GH20-2140-1

Program Product

Airline Control Program/ Transaction Processing Facility (ACP/TPF) General Information Manual

Program Number 5748-T11

Airline Control Program/Transaction Processing Facility (ACP/TPF) provides a reliable, highly responsive solution to realtime transaction-driven applications. The performance characteristics of ACP/TPF, which is capable of supporting large terminal networks with thousands of terminals, are largely due to specialized management techniques designed to optimize system efficiency in the area of data communications, data base, and systems resources.

This manual describes the functions and processing of the system. Included are the hardware supported and the customer responsibilities for use of the system. This manual provides customer executives, system administrators, system analysts, system programmers, and application programmers with a fundamental understanding of the Airline Control Program/Transaction Processing Facility.

IBM

GH20-2140-1

Program Product

Airline Control Program/ Transaction Processing Facility (ACP/TPF) General Information Manual

IBM

Second Edition (May 1979)

This edition is a major revision obsoleting GH20-2140-0.

This edition applies to Version 1, Modification Level 0, of the program product Airline Control Program/Transaction Processing Facility (5748-T11) and to all subsequent versions and modifications until otherwise indicated in new editions or Technical Newsletters.

Changes are continually made to the information herein. Therefore, before using this publication, consult your System/370 Bibliography (GC20-0370) for the editions that are applicable and current.

Requests for copies of IBM publications should be made to your IBM representative or to the IBM branch office serving your locality.

A form for readers' comments has been provided at the back of this publication. If this form has been removed, address comments to: IBM Corporation, Technical Publications, Dept. 824, 1133 Westchester Avenue, White Plains, New York 10604. IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation whatever. You may, of course, continue to use the information you supply.

Contents

Introduction

A computer used in a batch data processing environment normally follows a repetitive cycle of events that may be planned and timed in detail by the programmer. In a realtime environment, this is seldom the case, since the sequence of operations is unpredictable. The volume and variety of messages received is such that several messages may be in the computer at any one time.

Obviously, then, different programming methods are required in a realtime environment. The control program must continuously schedule work, allocate storage, and assess priorities. It permits message processing on a computer- and component-sharing basis to maximize the use of the various system resources. These resources include main and file storage, input/output components, terminal equipment, and the processing performed by the central processing unit.

Much of the efficiency of the Airline Control Program/Transaction Processing Facility (ACP/TPF) is gained by providing only those functions that are necessary. In the communications area, a limited number of terminal types are supported. In the area of file storage, both the usage of a single-access technique and rigid formatting reduce control program requirements. Other areas with concepts that contribute to efficiency include the management of entries, working storage, and programs. The inherent complexity of a realtime system also necessitates development of special purpose test procedures, test tools, and support functions.

ACP/TPF and its support programs reside in different environments. The primary realtime environment, under control of ACP/TPF servicing the application function, is often referred to as the online system. Most of the secondary programs and facilities whose purpose is to support the online system operate in a batch-oriented environment, under control of an OS/VS system. This environment is referred to as the offline system.

General Description of System

Initially developed for the growing air passenger travel industry, ACP/TPF has been adopted for use in other, completely diverse businesses that required similar system attributes, for example, the automation of the clerical, recordkeeping function with no impact on the casual and natural dialogue between the reservations agent and the customer during the transaction. Terminal response, availability, reliability, and recoverability were the overriding systems design considerations.

ACP/TPF is applicable to any online transaction-oriented application that requires fast message response time from a large number of terminals. Some examples of applications are airline seat reservations ... hotel reservations ... credit authorization/verification ... car rental reservations/billing ... police car dispatching ... electronic funds transfer switching ... teller memo post ... message switching ... loan payment processing.

Today, such a system has 3000 terminals and performs a realtime inventory control application 24 hours per day at a rate of more than 100 messages per second against a 5 billion-byte data base.

Reliability

ACP/TPF systems are characterized by thousands of terminals dispersed over a large geographic area, where each location may have from one to several hundred terminals. This environment dictates the need for extremely high availability and rapid, consistent responses. Fallback, restart, and recovery functions must be fast and must be accomplished with little or no awareness by the user, and with limited impact to the performance of the system. Availability in excess of 99% can be achieved at ACP/TPF installations. This is the result of a well managed operation and the use of the many support tools provided:

- Over 300 operator commands
- one- to three-minute system restart
- Online data base load/dump/copy
- Online debugging aids and offline test tools
- Dynamic system performance monitoring
- CPU, line, and I/O error recovery and recording

ACP/TPF Capabilities

ACP/TPF is a performance-oriented facility. As such, its capabilities can best be described in terms of a message processing rate and the response time for a message. Both of these measurements are dependent on application design; therefore, values stated are based on a known design, the Programmed Airline Reservation System (PARS). An average message in PARS requires approximately 14,600 instruction executions and ten file storage references (accesses). Other applications, such as retail banking or point-of-sale credit authorization require 10-12,000 instruction executions and six to eight file accesses. However, this path length can vary significantly, depending on the complexity of the application and the requirements for message switching, encryption, and message recovery.

ACP/TPF was designed to achieve an average response of one second, with 90% of message responses within three seconds. In a different environment, such factors as application design, communication techniques, etc., affect the response times.

ACP/TPF system support ranges from a System/370 Model 135 to a Processor Model 3033.

The number of terminals required to support a given message rate is dependent on the usage of the terminals. Even with a response time to a terminal of one second, the agent at the terminal might input a new message only once per minute. Time is required for the agent to react to the previous message and/or to input the next one. Existing systems operate with up to and in excess of 5000 terminals.

The number of instructions executed when processing an average PARS message has already been noted. The fact that this path length is low is attributed to the efficiency of ACP/TPF in controlling the message processing and to the efficient code inherent in the application when coding under ACP/TPF restrictions. The application instructions executed per message account for approximately one-third of the total number of instructions executed per message.

The number of file devices required is determined by the access and volume requirements for file storage. An access is defined as a control program action that initiates a physical read or write to file storage. If levels of indexing are used, each level may require an access. The access requirements include any overhead accesses that should be allocated to the message. (For example, output of a message response may require temporary file storage until the response can be sent.)

Security and Auditability

The constant availability of the online data increases the exposure of file and main storage to the effects of software and/or hardware malfunctions. This exposure can be minimized by ensuring that critical data can be replaced if necessary. ACP/TPF support programs are provided that help protect against permanent loss of data and help assist in maintaining system file storage integrity. These facilities are described in the section entitled *File Storage Support.*

ACP/TPF also provides a generalized message logging and tracking mechanism that can be used to increase message security. As an example, before transmitting a request to a remote computer, the application can request that ACP/TPF log the output with any associated data and activate a timeout. When the data reply is received, the application can retrieve the log record and deactivate the timeout. ACP/TPF will provide for the integrity of the log over a system interruption. Message security and integrity facilities will differ somewhat according to the line protocol and the nature of the remote devices. As an example, the SNA sequence number of an inbound message is available to both the host and the 3600 cluster controller. This number can be used for audit trails and message reply correlation at the cluster controller. Facilities related to message integrity and security are discussed later in this document under *Communications* and *Communications Program Support.*

Customer Responsibilities

Implementation of an operating system, including the development of application programs, can be a complex and difficult project that demands careful planning, control, and organization. ACP/TPF is typical of transaction-oriented operating systems in that it requires significant resources and commitment by the customer. The document *ACP Installation Overview - GE20-0597* describes the techniques for installing an ACP/TPF system.

To successfully install and use ACP/TPF the customer responsibility includes installing at least the minimum required machine configurations, communication equipment, and appropriate communication lines. In addition, the customer must have installed the appropriate operating system as required by the ACP/TPF support function. The customer must do the following:

- Have a thorough knowledge of the ACP/TPF application
- Train systems analysts, programmers, and operators in ACP/TPF
- Develop an implementation plan
- Design and create a data base
- Design and create a communication network
- Design terminal formats
- Design and implement application programs using ACP/TPF macro instructions and the Basic Assembler Language
- Develop procedures to assure adequate security for data on the system
- Develop appropriate backup procedures for the application
- Develop conversion procedures and schedules.
- Develop a procedural plan for monitoring the performance and tuning the system

The installation of a system such as that using ACP/TPF entails the matching of that program's variables to the environment in which it will reside. Terms that are associated with this process include *system generation*, *installation*, and *implementation*. As a rule, system generation refers to the control program, whereas installation and implementation refer to both control and application areas. Installation support provided with ACP/TPF includes the system initialization package and a portion of ACP/TPF documentation labled *Airline Control Program Systems Guide.*

System Initialization Package (SIP)

This package consists of documentation, macros, and programs that are used to generate an operational control program. The generator of a system using

SIP assigns variables, using terms that are common to the environment. SIP converts these terms to a control language that can be interpreted by the system compilers. SIP is designed to ensure that the assignment of variables is complete and that the variables are compatible with one another.

ACP Systems Guide (ACPSG)

This is a portion of the ACP/TPF documentation that describes in detail the program parameters required for installation of an ACP/TPF system.

Processing Description

ACP/TPF is a reliable, highly responsive, performance-oriented operating system for realtime transaction-driven applications.

ACP/TPF systems are characterized by thousands of terminals dispersed over a large geographic area, where each location may have from one to several hundred terminals. ACP/TPF provides realtime inquiry and update to a large centralized data base, where message length is relatively short in both directions, and response time is generally less than three seconds.

The functions performed by ACP/TPF include the following:

- Control of incoming and outgoing messages. Receives and transmits messages over low- and medium-speed communication lines.
- Main and file storage management. Controls allocation and release of main and file storage as requested by application programs.
- Queuing of work to be done. Maintains lists of entries waiting for event completion or further processing.
- Priority of processing. Reactivates entries on a system priority basis.
- Input/output control. Services all input/output operations, usually upon request of the application programs.
- Error checking and error recovery. Identifies, logs, and resolves (where possible) all permanent and transient equipment errors.
- Operator communications. Communicates with the operator and provides pertinent information considered necessary by the control system or requested by the operator.
- Restart and switchover. Provides a means of starting active operations in a computer system or restarting operations in the standby system.

All of the foregoing functions are performed in one manner or another by most operating systems. The differences between the performance of each can be attributed in large part to the concepts employed to accomplish each function. This section points out the concepts employed in the design of ACP/TPF to achieve the primary objective of performance.

Messages, Entries, Tasks, Jobs

All of these terms have been used at one time or another to describe units of work in an operating system. The terms used in ACP/TPF are *message* and *entry.*

ACP/TPF is a system in which units of work are initiated by messages; it is a conversational system in that a single response must be provided for each message. The term *message* applies to the input characters that trigger a work unit.

When ACP/TPF receives a message, it assigns a portion of storage called an entry control block. The term *entry*, derived from this, refers to the processing associated with an entry control block. Once activated, each entry is processed with equal priority. The life of an entry is measured from the creation of the entry control block to the deletion of the block.

Task and *job* have no meaning as such in ACP/TPF; however, if used, they are usually equated to the term *entry*.

Storage

Storage is divided, for purposes of this discussion, into three classes (see Figure 1). The first class, referred to as main storage, is used by a processing unit to retrieve and execute instructions. This storage, usually part of or directly associated with a CPU, is also referred to as main storage. All examples given in the following section refer to the PARS application. Values are typical and do not relate to any particular system.

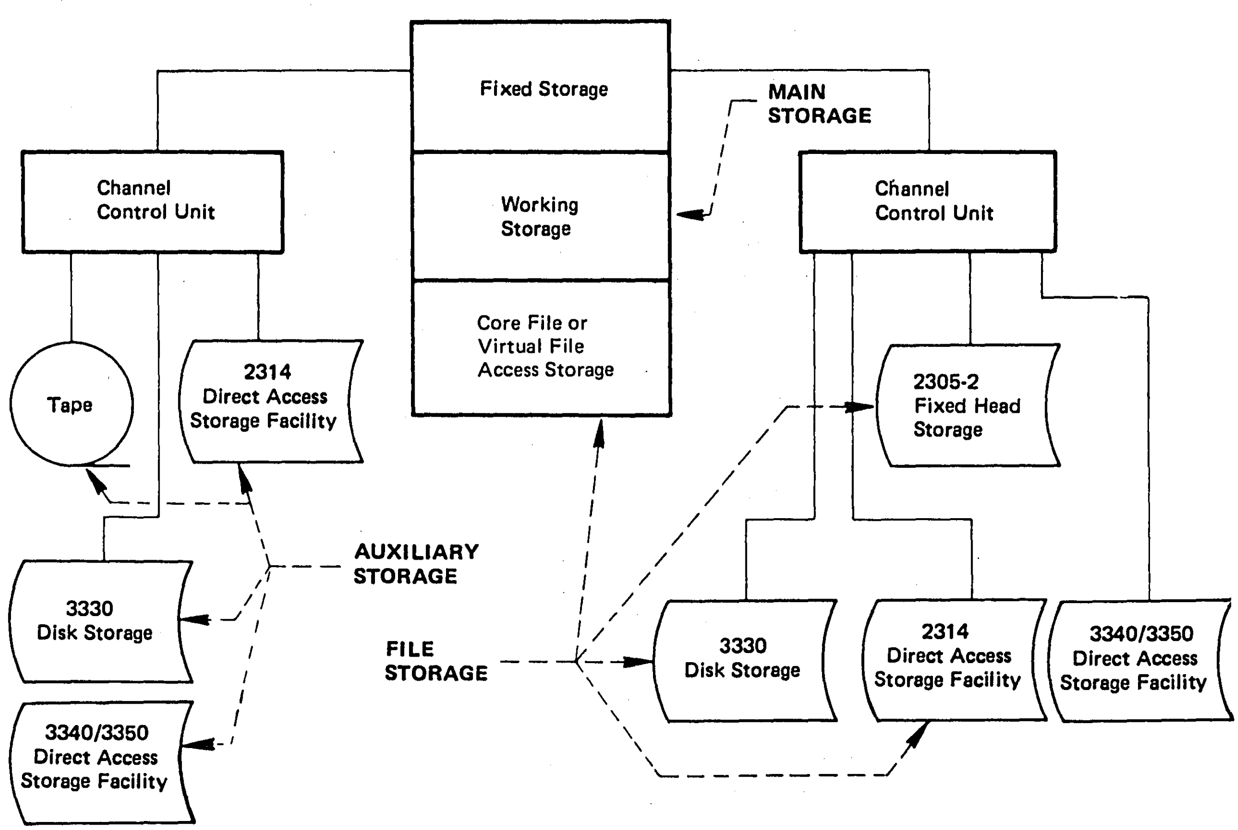


Figure 1. Three classes of storage

A second class of storage, used to hold programs and data, is called file storage. File storage must be readily available (in the order of milliseconds); however, its data must be transferred to main storage for processing. A portion of main storage set aside for use as file storage is an exception. Although this storage (core file) is treated by applications in the same way as any other file storage, it is part of main storage, and, as such, the data or programs can be operated on in place.

The third class, called auxiliary storage, is used to hold data and programs that have limited retrieval requirements during the life of a message. The data is usually placed in such storage for processing at a noncritical time or date.

Main Storage

Since main storage is required for program and data manipulation, its organization is the key to the design of a high-performance system. Part of main storage contains data and programs that are common to all entries and which remain in main storage. This portion is called fixed storage. Another portion, called working storage, is allotted to entries as required. When required, data and/or programs are transferred from file storage to working storage.

Each entry may have unique requirements for working storage, in terms of both a maximum and an average amount. A system's theoretical storage capacity for concurrent entries is calculated by dividing the amount of working storage available by the average amount of working storage required by an average entry. Since storage requirements of an entry vary during its life, and several entries may require maximum storage at any particular time, systems are designed to accommodate the condition in which many entries require a maximum of storage. This provides a practical design level for the number of concurrent entries.

The life of an entry in the system determines the message processing capabilities of that system. Again, from a theoretical point of view, if each entry had a life of one second, the message-per-second rate would equal the number of concurrent entries. In reality, variations in message life necessitate designing to a percentage of available time. Factors affecting the life of an entry include the CPU processing capability, the number of instructions that must be executed both in the control and in the application areas, and the amount of time spent waiting for the completion of transfers to and from file storage.

Less than 2% of this period is actual instruction execution by the processor. After an entry is given control of the processor, it will return control to ACP/TPF only when it cannot process any further without an I/O completion or when it has completed its function and wishes to terminate. ACP/TPF assumes that if an entry has control of the processor for an excessive length of time, the entry has malfunctioned, and ACP/TPF will abort the entry.

Although many factors influence the performance of a system, main storage is the place where entries are buffered during processing; thus, external factors cannot increase performance beyond the limits imposed by main storage. In fact, the first symptom of file storage performance deterioration is a lack of sufficient main storage. This occurs as the life of each entry is lengthened because of increased time required to service file requests.

The following sections discuss main storage and the techniques used in ACP/TPF to optimize its use (see Figure 2).

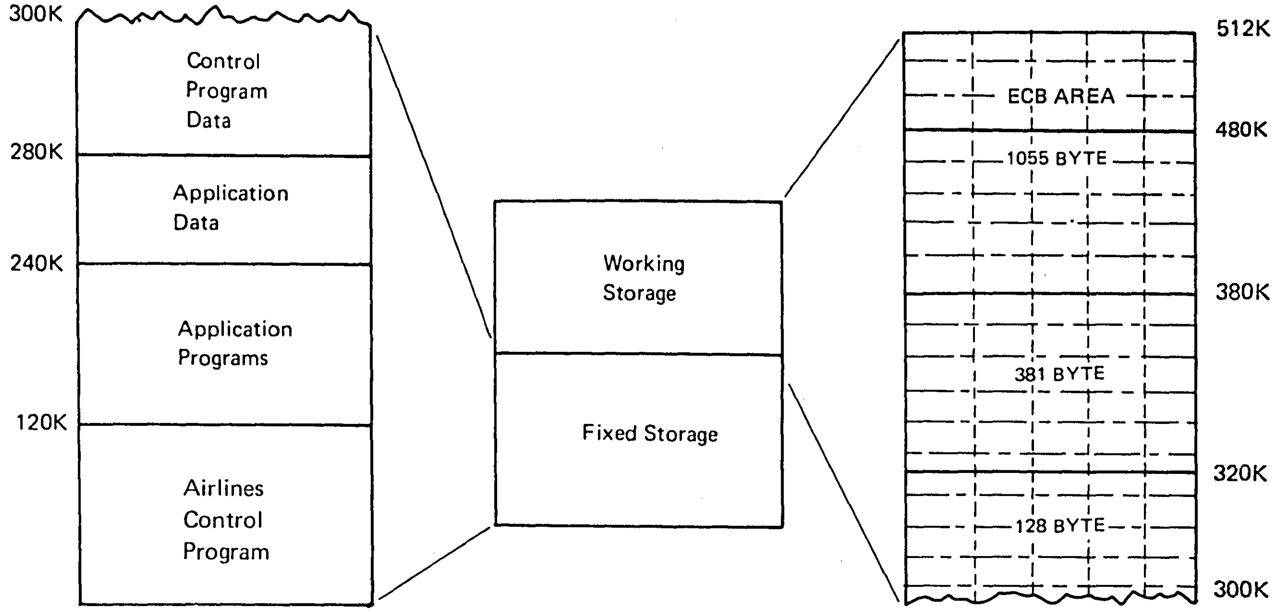


Figure 2. Main storage (typical values for a 512K system)

Fixed Storage

An amount of main storage is required to hold data and programs that control the basic operating system. Control includes scheduling the entry control blocks, handling the communication networks, and controlling the transfer of data between main and other storage. Other control functions may also reside in fixed storage or may be called into working storage as required.

Virtually all application programs (those that provide the function defined by the desired application) can reside in file storage and be called in to main storage when required. Some programs, however, have a high frequency of usage. Calling these from file storage adds time to the life of the entry and increases demands on working storage. If these are kept in fixed storage, the time to transfer programs is eliminated, decreasing the life of the entry and effectively decreasing the amount of working storage required. This decrease in working storage must be weighed against the increase of fixed storage when optimizing the system. In practice, it is found that efficiencies can be realized by placing a number of application programs in fixed storage; thus, an application program area of fixed storage is provided.

As in the case of programs, certain application data is best kept in fixed storage because of a high demand on such data. This application data area is called the global area.

Included in the fixed area, then, are a control program and its associated data records, an application program area, and an application data area. The sizes of these areas are dependent on the devices to be supported by the control program and the amount of data and number of application programs placed in main storage to optimize performance.

Working Storage

Working storage is allocated to entries during their system life. The elements allocated are entry control blocks (one per entry), programs required from file storage, and data blocks either created in main storage or transferred from file storage. The aim of ACP/TPF is to utilize working storage as efficiently as possible. In some systems, available areas of main storage may become temporarily unusable because they are too small to meet the requirements of entries waiting for main storage. This condition is called fragmentation. The problem is compounded if entries require contiguous storage for programs and/or data records.

ACP/TPF seeks to minimize these problems by dividing working storage into four areas, each area with blocks of equal size, and by not requiring contiguous storage. Successive blocks could contain a program, and a data record, each for a different entry. Two of the areas relate to records or blocks in file storage. In the small record area, the block size is 384 bytes; in the large area, it is 1056 bytes. The third area (128-byte blocks) is used for data records not stored in file storage. The fourth area contains entry control blocks ECB's. The ECB block size is variable and user-specified with a maximum size of 4095 bytes.

Protection

Storage protection is used for fixed storage, inhibiting an application from illegally modifying the control program or main storage-resident application programs and/or critical data records. Protection is not provided between entries in working storage. The strict requirements in application design and the extremely dynamic nature of working storage usage reduce the exposure of nonprotected main storage to the extent that this has not been a problem, and it is not anticipated that it will become one.

File Storage

The performance of a system is dependent on the number of file storage requests and the time required to transfer the data. The number is largely determined by the application design. The request time includes the amount of time taken by the control program instructions, the time required to find the requested block on a physical device, the time in queue for the device, and the transfer time. Since each device can handle only one request at a time, multiple requests for a particular device must be queued.

The control program instruction time will be discussed in another section. The transfer time and seek time are based on the device characteristics. The queuing time, however, is dependent on the file organization. A first step in reducing the queue for any device can be to design a system that approaches equal queue sizes for each device. The average length of the queue on a device can then be reduced by increasing the number of devices.

Two factors governing file design are the capacity of the files in relation to the data to be contained and the ability of the files to handle requests at a rate consistent with the requirements of system performance. As queuing increases, the data-request time increases, and the life of an entry increases; thus, a larger demand is made on working storage. This results in either a reduced message rate or a system failure because of a lack of working storage. Core file storage (CFS) is effectively an extension of file storage, except that the data resides in main memory: an area in main memory set aside for specific records so that an access to these records is relieved of the file access system overhead. Application programs access data residing in core file storage with the standard ACP/TPF macros, and the location of the data is transparent to the application program.

Virtual file access (VFA) is a facility that alleviates the potential accessing of application records from the file subsystem, thereby reducing the accessing requirements. The allocation of a set of records with high reference frequencies to VFA has the potential for reducing the number of I/O requests and system path length; this improves response, as well as throughput. The main storage into which these records will be placed during application processing is managed to eliminate redundant file accessing. The system uses a record sharing table to keep track of the records that are currently in main storage. This is transparent to the user application. VFA may also be used to improve resource usage for certain direct access storage devices. Improvement in the access-to-volume ratio is possible for large-capacity DASD devices that are I/O-bound. The VFA and CFS features are mutually exclusive.

Record Sizes

File storage is organized for two record sizes. These sizes (381 and 1055 bytes) are common to all file storage media and are consistent with the core block size in working storage. The difference in size between these and core blocks is due to core doubleword boundary requirements in main storage. Application software is designed to record sizes and need not be concerned with the storage media.

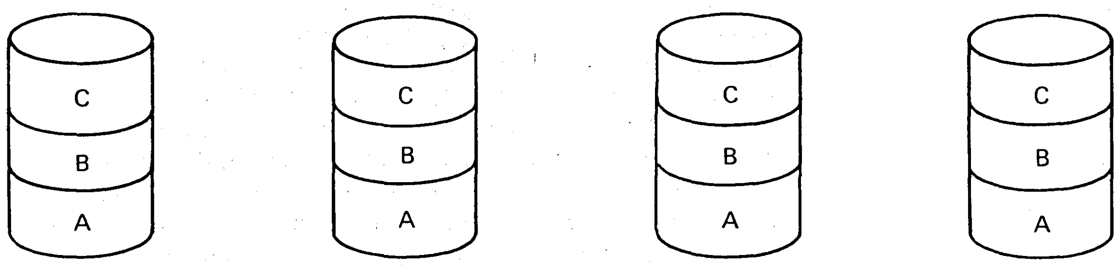
Record Organization-General

The organization of a file storage system is intended to distribute the records associated with a set of data, called a record type, over physical file storage to reduce queuing time at the devices. Thus, when an application accesses successive records within a record type, each record is obtained from a different physical component.

A record type contains data records that are related by application or by usage. Different record types may be files of programs, inventory records, customer accounts, payroll histories, etc. Within applications supported by ACP/TPF, there may exist many record types.

Traditionally, record types are called logical files and are customarily assigned to one or more storage media units. This has an advantage both in isolating particular files for protection and processing and in allowing an installation to reduce the number of media units when the file does not have a requirement for continuous availability. This latter advantage is lost in a realtime environment when all logical files must be available at all times. The main disadvantage of such an organization arises when the applications on a particular logical file require a high activity. This results in a queuing problem. Trying to balance logical files on storage media is very difficult and fails when the application logical file profile changes during operation.

ACP/TPF minimizes the problem of uneven queuing on DASD drives by distributing the logical files over all units of the particular device type (see Figure 3). This distribution is mandatory and automatic for all DASD devices with each type having its own distribution. The organization might be envisioned as a group of disk packs, each identical in format and type of contents. The logical files are layers on the packs, each pack with an equal percentage of the total logical file.



These represent either four 3330, four 3340, or four 3350 drives (cannot be intermixed). Residing on these are three logical files (A, B, and C). Each logical file shares an equal portion of each physical device.

Figure 3. Logical/physical device relationship

Record Organization–Detail

The distribution technique in organizing fixed file storage is to place the first record of a logical file on the first physical device, the second record on the second device, and so on to the Nth device, after which the devices are repeated (that is, logical records 1 and N+1 are on the first physical device). (See Figure 4, part 1.)

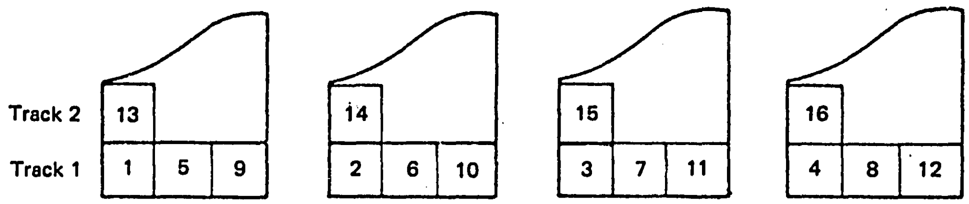
The purpose of this organization is to allow a larger number of concurrent accesses to any particular logical file than would otherwise be possible, thereby reducing the chance of excessive queuing. An additional advantage is that it frees the application design from many of the physical device performance considerations.

Core file and 2305 organization differ from other DASD devices in that distribution of logical files is not automatic. Logical files are consecutively added to the physical files. (See Figure 4, part 2.) Distribution can be accomplished by dividing a logical file into smaller logical files and positioning these on separate devices. (See Figure 4, part 3.)

Each of the physical device types has its own organization. Logical files are allocated on each type according to the rules of that type. Logical files can be split, with each portion placed on a different device type. (See Figure 4, part 4.)

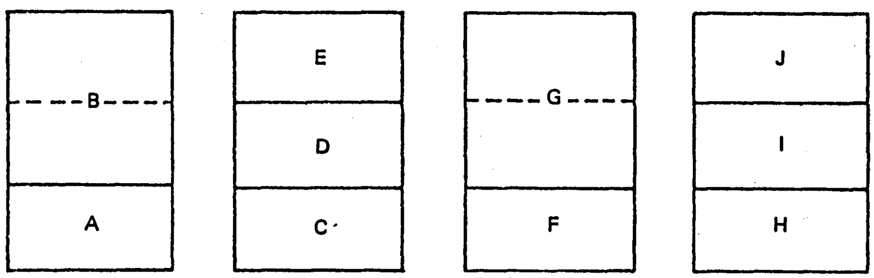
Note that all records allocated to a particular device type are contiguous.

Part 1 Record allocation (DASD)



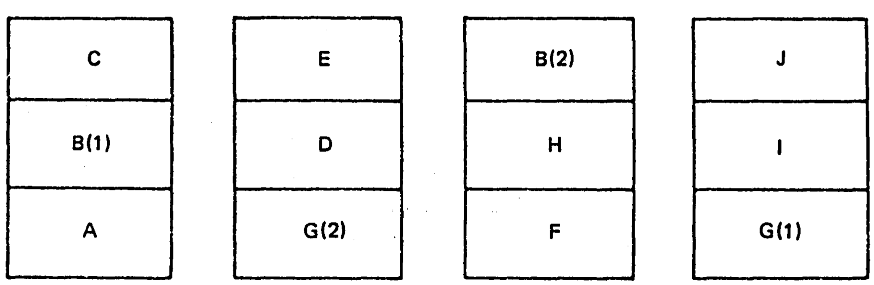
The above is an expansion of the logical files shown in Figure 3. As can be seen, consecutive records are placed on consecutive devices.

Part 2 Logical/physical relationship (core file, 2305)



Logical files A through J located on four core files or 2305s. Record organization is sequential on a device. Note that, like the DASD organization, all physical devices have the same format.

Part 3 Core file, 2305, logical file distribution



The same logical files as in part 2, except that they have been redistributed, dividing logical files B and G between devices. Records within files B and G are consecutive on each device.

Figure 4. Record organization detail

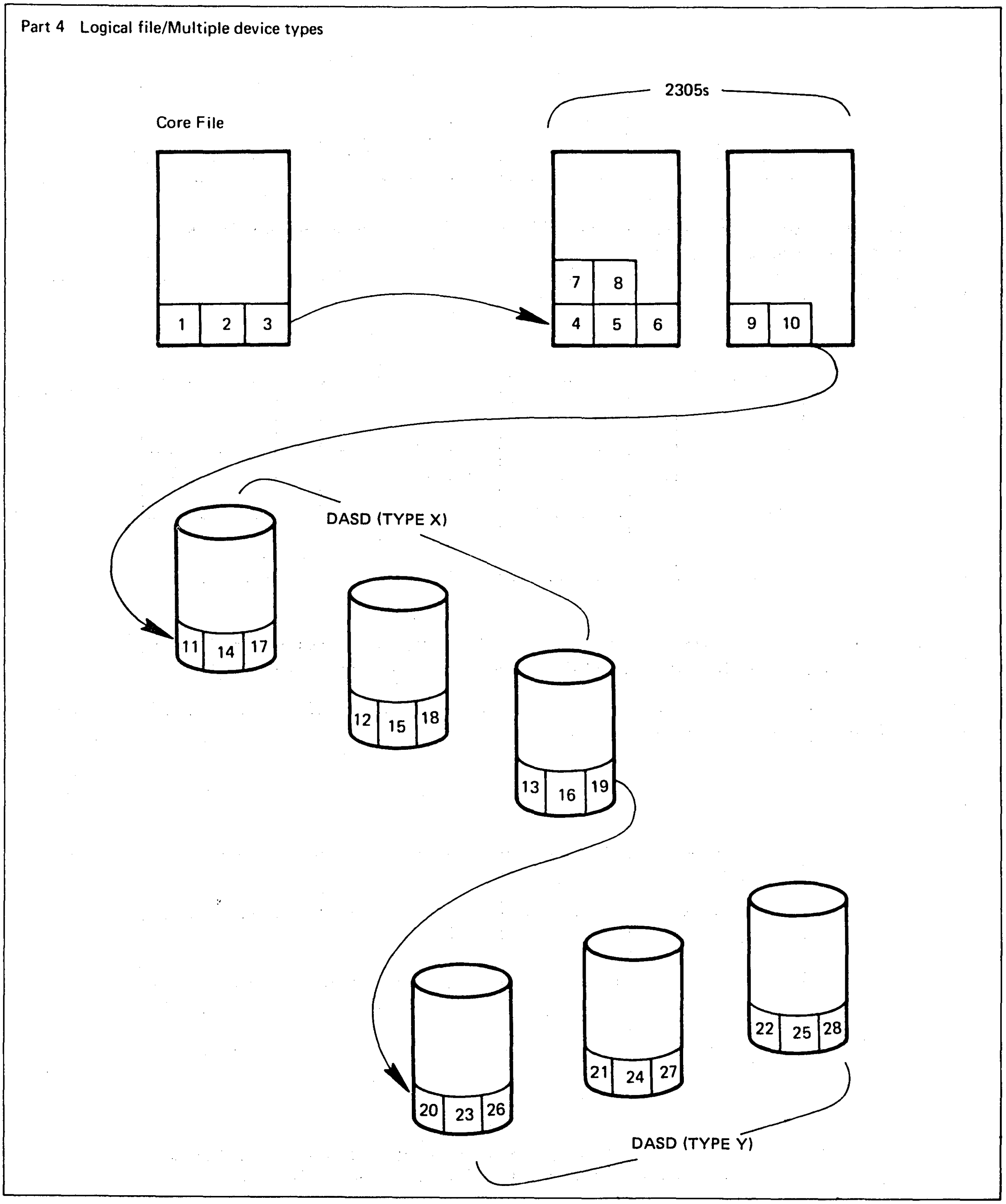


Figure 4. Record organization detail (continued)

Record Addressing

The file address used by an application program is sometimes called symbolic to emphasize that the translation required to generate an address directly utilized by the hardware is always completed by online ACP/TPF system programs.

The techniques used by the application to obtain addresses relate to the record definition on the files. Records on file storage are classified as either fixed or pool records, based on how their addresses were derived.

The fixed record area is an area of file storage that contains records whose symbolic addresses can be calculated using a record type and an ordinal number. The record type identifies a set of data within the fixed record area, and the ordinal number identifies a specific record within the record type. (As an example, consider an inventory file for an airline. A record type for flights can be identified, and a record can be set aside for each possible flight number on each date for which inventory is kept. Using the flight number and date, the sequence number of the appropriate record can be determined. Today's flight 300 would be on the three-hundredth inventory record, tomorrow's on the 300 + Nth record, where N is the maximum value of a flight number.) Obviously, this technique is wasteful if the number of records set up far exceeds the requirements of usage. When fixed records are set up for logical files, file storage is allocated and maintained for the exclusive use of each possible record in that logical file.

A different method of obtaining an address must be employed when either the number of records used is far less than those required on a fixed record scheme or the address cannot be calculated. (Although it would be possible in a logical file of names to allot a record to every possible name, it would be unreasonable in terms of the file storage required.) To solve this problem, a number of records are set up in file storage whose addresses are managed by ACP/TPF. The records set up for this purpose are called pool records. There are pools for each size record. When an application needs a new record, ACP/TPF gives it the symbolic address of one of these records in the pool. When the application is through with the record, its address is returned to ACP/TPF for recycling. ACP/TPF ensures that an address is not given to more than one requester before its release.

To retrieve a pool record, the application must be able to obtain its address. The common technique used by applications is to form data structures that use fixed records as indexes to pool records. For example, the reservation application uses a fixed record to locate passenger name records in the pool. The index is retrieved, using the flight number and date to calculate its symbolic address. The index is scanned for the passenger's name, and, when it is found, the associated pool address is used to locate the passenger name record. This allows the selection of one of a vast number of pool records with a minimum of file accesses.

Record Duplication

Duplication is insurance against loss of critical data because of hardware failure. If a record is duplicated, two copies will be updated whenever file storage is updated. Duplicates are kept on separate devices. When requested, either copy can be used, subject to the device queues and/or in the event of the inability to retrieve one of the copies. ACP/TPF supports the duplication of any or all logical files in both the fixed and the pool areas.

Areas on the DASD devices may be reserved to maintain a copy of the core file and/or 2305 files. Operator commands will affect the preservation or restoration of core file or 2305 contents.

Auxiliary Storage

A number of support functions process data apart from the online system. Some applications do not require immediate processing of data (the bank teller is given a response indicating the process request was recognized), and/or the processing techniques required may not be acceptable in an ACP/TPF environment (for example, a sort program that ties up considerable file and core resources). To accommodate these cases, provisions are made in ACP/TPF to transfer data to and from auxiliary storage. The devices used for auxiliary storage are disk files and tapes. This document is concerned with the online interface with auxiliary storage. Offline processing techniques must be consistent with the system used for that processing.

Disks used for auxiliary storage are referred to as general files or general data sets. The application interfaces with ACP/TPF, using a direct address file access method. The application, therefore, must either dictate pack formatting or be aware of the existing pack format. General files or general data sets do not have to be formatted the same as file storage; however, their record sizes must be consistent.

There are two classes of tape files. The tapes of the first class, called realtime tapes, are mandatory on a system and are used by ACP/TPF and applications both for output critical to system operation and for testing. The second class consists of general tapes. These tapes are treated as a sequential file and can be either written or read by the online system; however, the record (block) sizes must be consistent with the block sizes in main storage.

Communications

The term *communications* encompasses the devices and facilities that enable interaction with a CPU using communication line facilities. A user interfaces with the system by means of a device called a terminal. This device has an input and/or output facility. The input facility is usually a typewriter keyboard, and the output either a print mechanism (hard copy) or a video display (soft copy or CRT).

Figure 5 depicts types of devices that may be included in a communication network. Elements in this figure are:

- Communications Controller - Used to control the transmission and receipt of data over communication lines and to assist in transfers of data between itself and main storage.

- Remote Multiplexer - Used to concentrate a number of communication lines into one line for transmission to the host, buffering messages in the process. The communication techniques for individual lines may be different.

- Terminal Interchange - Used to provide sharing of terminal control facilities with multiple terminals. Sharing of functions reduces the prorated cost per terminal. A single device may act both as a terminal interchange and as a multiplexer.

- Cluster Controller or Terminal Controller - Used to manage the attached terminals and data transmission between the terminals and the central processing facility. Data formatting and local transaction processing are also performed by the terminal controller.

- Terminal Control Unit - Directly associated with or part of a terminal. This device provides some of the control, buffering, and/or storage required for terminal operation.

- Modem - Used to convert the signal on a communication line to a form acceptable by the devices used and vice versa.

Another purpose of Figure 5 is to show how the foregoing elements may be connected. Six paths are shown, the choice being dictated by such factors as the system cost, number of terminals at each facility, message loads, reliability, response time, and data security/privacy factors.

An inherent part of system design is the time it takes to respond to an input message (response time). This is measured as the time between user initiation of message transfer to the CPU and the receipt by the user of the first character of the response. Elements of response time are the life of the entry in the CPU and the time spent in the communication system. Factors that influence the time spent in the communication system include:

- Capacity of the communication line

- Message size

- Time taken by an intermediate device to handle a message (modem, multiplexer, controller, etc.)

- Data link protocol

- Polling frequency (for polled data links)

- Time spent by the CPU in handling the communication function

Another factor in the design of a communication system is the desire to keep line requirements low. As pointed out earlier, multiplexers and terminal interchanges help in this area. Other means of optimization make full use of a line's capacity by:

- Minimizing the number of bits required to define a character

- Minimizing the use of control characters required to operate the line, thereby leaving more room for data

- Operating lines in full duplex mode (simultaneous send and receive)

On any system, all of the foregoing factors and devices must be weighed and balanced to provide a system that is economical and satisfies performance and reliability requirements.

Before describing each of these specific line disciplines, the terms *start/stop* (or *asynchronous*) and *synchronous* should be defined. These are the two most important means of identifying characters transmitted. Since data on a communication line is a continuous string of bits, hardware must have the ability to gather bits into characters. Start/stop characters have start and stop bits to frame each character. This method adds to line load; however, there is no time dependency between characters. Synchronous transmission entails the synchronization of an entire message. The message starts with synchronizing characters (usually two), followed by other characters as a string of bits. The hardware divides these bits into characters to assemble the message. Although there is a saving of control bits, the entire message must be transmitted at a steady rate to allow synchronization.

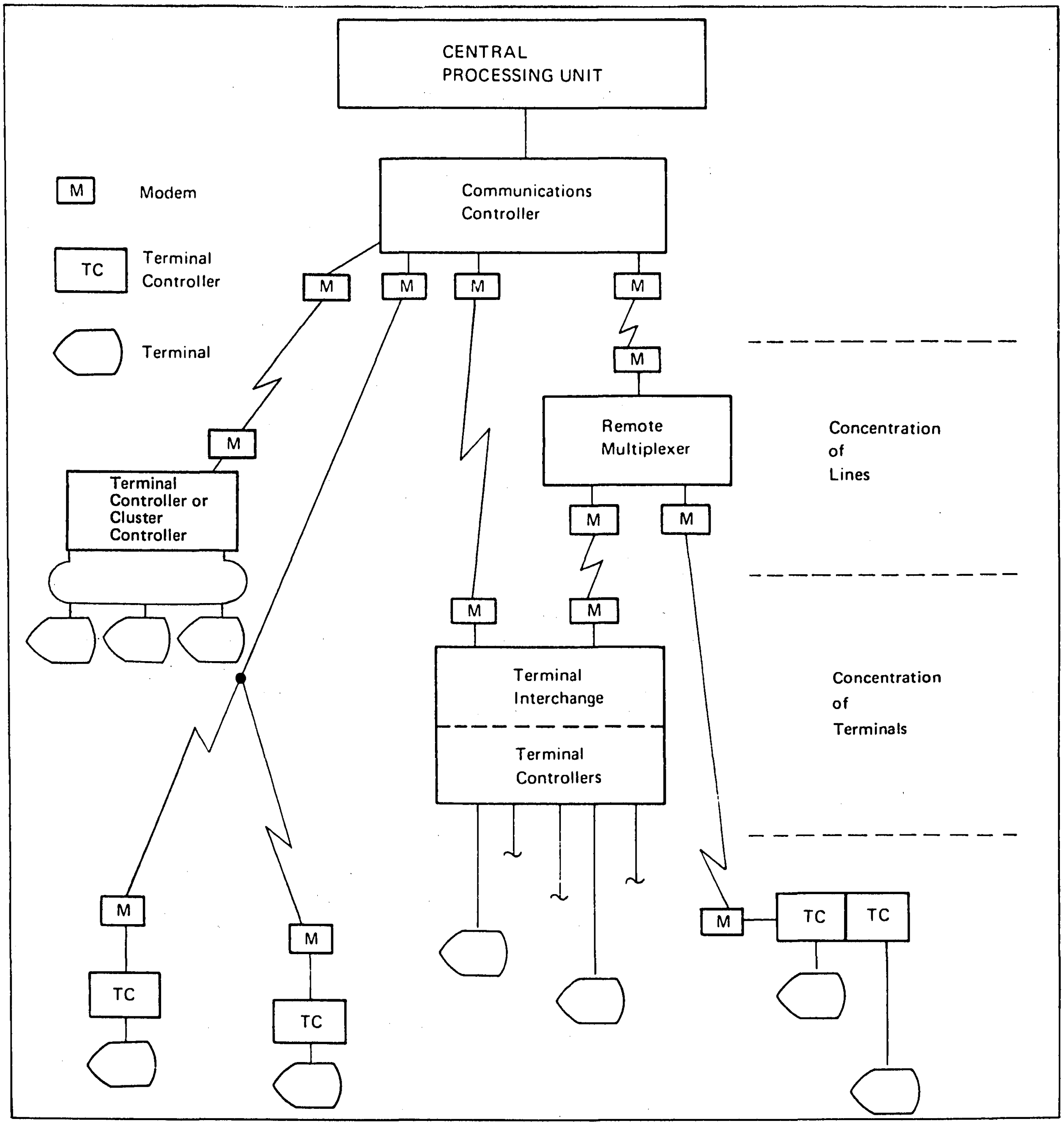


Figure 5. Communication network

Systems Network Architecture (SNA)

ACP/TPF provides systems network architecture (SNA) support for the 3600 Finance Communications System and for the 3270 Information Display Station. The following functions are provided with SNA support:

- Distributed function of remote intelligent controllers
- SNA and SDLC message integrity
- SDLC line control efficiency

ACP/TPF resides in the host System/370 and interfaces with the SNA network via the 3705 Network Control Program (NCP/VS). The message protocol used by the 3600 application programs generated via the Program Customizer (PC3600) is supported by ACP/TPF.

In addition to the one input and one output message protocol used by PC3600, ACP/TPF provides logical unit *pipeline* support similar to CICS, where multiple terminals at the intelligent controller share a single SNA communication path to the host.

In any telecommunications system, the time required to perform a restart after a system interruption must be short. In keeping with the objective of less than three-minute restart, the ACP SNA software does not automatically reinitialize the network on restart. Normally, the host and the cluster controllers will preserve message sequence numbers over a system interruption, and no network initialization is required. However, following a 3705 failure, ACP/TPF performs the network startup function. This is performed in a fully parallel fashion to minimize the time required for network restart.

Application Interface

The ACP/TPF/application message interface consists of a parameter list with message routing information and a main storage block containing the text of the message. This interface is common to all network components regardless of line discipline and terminal characteristics. However, each SNA network-addressable unit can optionally be addressed by a symbolic eight-character name.

Network Definition

Network definition is the procedure through which the user describes each logical unit in the network. The description includes the logical unit's node name, type, and modes of operation. The modes of operation are:

- Multithread or single-thread
- Recoverable or nonrecoverable logical units
- Unsolicited messages or no unsolicited messages
- Batch or interactive

Multithread operation is intended for logical units controlling many input terminals, such as banking-point-of-sale terminals. In this environment, the logical unit concurrently processes many input transactions. If a logical unit is defined as *multithread*, it may have up to 256 input transactions concurrently processing in ACP/TPF.

Single-thread operation is intended for logical units that send a single input transaction to the host and then await its reply. If a logical unit defined as *single-thread* sends another input transaction to ACP/TPF before receiving the reply to the preceding transaction, the input is rejected by the host returning a negative response.

For logical units defined as *nonrecoverable*, any host or transmission failure will cause output to be lost. If a logical unit is defined as *recoverable*, ACP/TPF guarantees that the input is processed to completion and the reply successfully returned to the logical unit. Optionally, an application can supply a routine to select which messages, from a *recoverable* logical unit, ACP/TPF should consider recoverable and which it should consider nonrecoverable.

An option allows logical units to be defined as permitted to receive or not permitted to receive unsolicited messages, that is, output for which there is no corresponding input. Unsolicited messages may be sent by the ACP/TPF operator, such as broadcast messages, or by host application programs.

Message Recovery

Message recovery is an optional feature of ACP/TPF that maintains information on every SNA input message currently being processed by ACP/TPF. The level of recovery is user-selected and extends from full recovery of all input and output messages to simply keeping track of each input message in process. For the host application programmer, the important aspect of message recovery is input recovery.

As each SNA input message is received, its origin and sequence number are recorded. The message is then passed to either an application or a system routine to determine its input timeout interval and recoverability. The input timeout interval is the time period ACP/TPF should wait for the application to respond. If no response is received before the time interval expires, the message is considered lost in the network, and is resubmitted to the application. The data resubmitted to the application is the original input if the input was recoverable or a canned message if the input was nonrecoverable. ACP/TPF assumes the application, on resubmission of input, will complete the transaction by either canceling the transaction or finishing its processing.

ACP/ACF Feature

The ACP/ACF feature provides the capability to link together multiple System/370s through the use of ACF/NCP/VS, and SNA advanced communication function (ACF) multisystem networking facilities (MSNF) supported by the ACP/TPF system. This allows for sharing of lines and terminals in an integrated network of computing systems. The architecture of the ACP/TPF system provides an economical vehicle to separate network management (that is, front-end processing) from systems designed to accommodate complex processing requirements. The ACP/TPF system provides the added advantage of transaction processing within a front-end processor.

ACP/TPF, as a cross-domain resource manager (CDRM), will communicate with ACF/VTAM, or ACF/TCAM, and other ACP/TPF CDRM's to control cross-domain session initialization, termination, and recovery. The ACP/ACF feature supports cross-system message routing, through which data may be transmitted across systems to its destination without host intervention, after initial session establishment. The ACP/ACF feature is the principal component in support of computer networks, discussed in a future section.

Bulk Data Transfer

ACP/TPF provides specialized SNA support to facilitate the transfer of bulk data to and from the remote cluster controllers. Transmission of negative credit files and off-host authorized transactions are two examples of bulk data. On interactive transaction messages, the application is shielded from the details of the SNA message protocol by ACP/TPF. However, during long transmission, the host application must establish meaningful checkpoints to minimize retransmission time in the event of a failure. For defined batch transfer logical units and application programs, ACP/TPF allows the application to receive, decode, and send SNA responses. The normal ACP SNA support is used in handling output for the batch logical unit.

Mapping/Paging

The application program achieves 3270 device independence by using the ACP/TPF screen mapping package to edit, construct, and format 3270 data streams. Input messages originating at the terminal contain field-oriented data consisting of device-dependent control characters and message text. ACP/TPF deletes the control characters and presents the text to the application as a set of variable-length data fields. This process is controlled by a predefined screen map. In the outbound direction, ACP/TPF constructs the terminal-dependent data stream from the application-provided data fields and the screen map.

When a display message is greater than the screen size, it is either handled as a continuous scroll or as pages in a book. The application provides the display message as a long output message. After the first page is displayed on the screen, the terminal operator controls the screen with page or scroll commands. Scroll commands reposition the screen image a line or group of lines at a time. The entire process is controlled by the screen map, which defines the page or scroll size. This feature facilitates split screen operation.

Operator Commands

These commands give the ACP/TPF computer operator the ability to control the SNA network. The status of an NCP, a line, a cluster controller, or an individual logical unit may be displayed at the console. The operator may also stop or start any portion of the SNA network and may alter the frequency for host selection of the NCP.

Online Load/Dump

The 3705 may be loaded or dumped online in parallel with realtime operation. The OS/VS-provided NCP generation process is used offline to create 3705 load modules. These load images are then stored in the online disk files and are transferred to the 3705 on command from the ACP/TPF operator. Dump images are temporarily accumulated online for later spooling to tape and offline printing in standard format.

The 3600 and 3614 may be loaded online in parallel with realtime operation. The OS/VS-provided subsystem support services package is used offline to create remote controller load images. The load images are stored in the online ACP/TPF disk files and are transferred to a 3600 or 3614 on command either from the ACP/TPF operator's console or from the remote cluster. Dump images are temporarily accumulated online for later spooling to tape for offline printing in standard format by subsystem support services.

Binary Synchronous Communications

ACP/TPF provides binary synchronous communications (BSC) support using standard point-to-point and multipoint data link protocols (GA27-3004-2). This facility is primarily directed at the computer networking requirement of large geographically dispersed networks. Because of the wide industry acceptance of BSC line protocol, ACP/TPF networking support allows communications with non-IBM as well as IBM systems.

Data Link Operation

The ACP/TPF support of BSC links is characterized by the following operational attributes:

Transmission code - To provide more general usability, ACP/TPF handles BSC lines operating with either EBCDIC or USASCII transmission code. EBCDIC transparency is optionally provided to allow the transmission of binary data between computers.

Blocked transmission - Long messages consisting of multiple message blocks can be received and sent by ACP/TPF. There is no limit to the total message size in either the inbound or the outbound direction.

Send limit control - To balance line utilization between input and output, ACP/TPF limits the number of output messages sent between input operations. The send limit is defined at network generation time and can be altered online by the ACP/TPF operator.

Master/slave contention priority - BSC point-to-point message protocol allows each end of the line to contend for transmission time. When both ends bid for the line simultaneously, the contention is resolved by a master/slave relationship. ACP/TPF may be either master or slave. This attribute is defined by line at network generation time and can be changed online by the ACP/TPF operator.

Control or tributary multipoint support - The control station on a BSC line either polls or selects the tributary stations. Polling invites the tributary to send data. Selection requests permission to send from the control station to the tributary station. ACP/TPF may be either the control or the tributary station.

ACP/TPF interfaces with the BSC lines via the 3705 Emulation Program (EP). When SDLC and BSC links are connected to the same 3705, ACP/TPF supports the Partitioned Emulation Program (PEP).

Where multiple BSC lines exist between two computers, ACP/TPF provides alternate line routing in the event that one or more of the lines fail. ACP/TPF also returns undeliverable messages to the originating application when all lines to a particular destination are inoperative.

When multiple lines exist between two computers, output messages are queued to the line with the smallest message queue. This balances the line usage and promotes fast delivery of messages.

Message Integrity

BSC line protocol requires positive acknowledgment on each message transmitted. In addition, ACP/TPF provides a generalized logging and tracking mechanism that can be used by the application programs. Before transmitting a request to a remote computer, the application can request that ACP/TPF log the output with any associated data and activate a timeout. ACP/TPF ensures the integrity of the log over a system interruption. In the event of a lost reply, ACP/TPF times out and initiates recovery action with the application.

Although 3614 encryption is provided with the SNA support, this algorithm can also be used to encipher and decipher messages sent and received over BSC lines.

Application Interface

The ACP/TPF/application message interface is common for all network components regardless of line discipline or terminal characteristics. The application can address the BSC network with either physical or symbolic addresses. The physical addressing is provided to ensure a short instruction path for message transmission.

Airline Line Control (ALC)

Airline line control was developed to satisfy the communication requirements of an airlines reservation system. The term *airline line control* (ALC) often encompasses the entire medium-speed communication system, rather than the particular line disciplines utilized.

The medium-speed communication system operates in a conversational mode: that is, for each input by an agent, there must be a response to that agent. The agent enters a message for transmission to the CPU. After a reasonable period, a response is returned to the agent, freeing the terminal keyboard for further input. If the response contains more than one screen of information, the agent may request the additional information with a short message. This is the normal sequence of events. Errors may cause either an error indicator to be activated on the terminal or the absence of a response. If the error indicator is activated, the agent, assuming the error occurred in responding, requests retransmission of the response. If a response is missing, the agent takes actions assuming that either the input message was lost or there was an error in processing. The action taken may be a repeat of the entry or some other action(s) appropriate to the application design. As can be seen, the agent is an integral part of the communication system, eliminating the need for extensive handshaking control messages such as *I got it.*

The foregoing description assumes a video terminal at which an agent is involved in the display of each screen of information. Multiple segment messages sent to hard-copy terminals are processed differently. A new segment cannot be accepted by a terminal until it has finished printing the previous segment. The terminal, therefore, indicates to ACP/TPF that it has finished printing by an answerback. ACP/TPF can then transmit the next segment. In this case, the equipment, not an agent, requests additional information.

Data Rates

Airline line control uses a synchronous line control, full-duplex, at rates of 2400, 4800, or 9600 baud, on dedicated communication lines. Each message contains two characters to synchronize the line/message, the data text, and a character to aid in determining the validity of the message. Each character is six bits in length. The advantages of six-bit characters are realized when determining the traffic that can be handled by a communication line. For instance, the theoretical capacity of a 2400-baud line using eight-bit characters is 300 characters per second, where one with six-bit characters is 400 characters per second, an increase of 33%.

Polling

ACP/TPF supports two polling methods, roll call and hub polling. When terminals or terminal interchanges (TI's) are in roll call mode, ACP/TPF treats each as if it were the only one on the line. When requesting data, ACP/TPF sends a poll message to the device asking for any data to be transmitted. The device always acknowledges receipt of the poll. Data precedes an acknowledgment. A TI transmits data from all attached terminals before acknowledging.

Hub polling is a technique used to increase line utilization and reduce the control program overhead for communications. In hub poll mode, all terminals and TI's on a line are treated as one line by ACP/TPF. ACP/TPF polls the most remote (in terms of line mileage) device. When this device acknowledges, it sends a poll message to the next-closest device; this procedure continues until all devices have been polled. Hub polling is used to reduce overhead when there are a large number of TI's (drops) on a line or to reduce propagation delays when the line is of such a length that the delays become significant. The disadvantage of hub polling is that it requires an extra modem, at all but the most remote TI, to enable the device to recognize poll messages on the input line to the CPU.

In summary, airline line control provides an efficient means of communication with ACP/TPF, in terms of both line capacity and performance. (Refer to Figure 6.)

Low-Speed Controlled Telegraph

Low-Speed Controlled Telegraph (LSCT) was developed in ACP/TPF to support existing communications facilities. This support was primarily required for domestic airlines that use 83B-type equipment for message switching functions.

LSCT supports multiple terminals on a line in a start/stop mode of communication. As this implies, each terminal can be addressed and must be polled for data. Half-duplex lines are supported, operating at rates up to 75 baud, or in other terms, up to 100 words per minute. The character size is five-bit baudot code used in telegraph communications.

At these rates, messages transmitted through LSCT have relatively long response times; therefore, in practice, these are low-priority messages not demanding immediate responses.

Low-Speed Free-Running Telegraph

Low-Speed Free-Running Telegraph (LSFR), like LSCT, supports an existing low-priority communication system. This technique (LSFR) is primarily used outside the United States, with the international airline carriers having the main exposure to ACP/TPF.

Communication in LSFR is point-to-point, start/stop transmission. Since the communication is point-to-point, there are no terminal addressing requirements. The line is free-running, eliminating poll messages and other control characters. The lines operate at rates up to 75 baud in full-duplex, half-duplex, or simplex modes. Characters are five-bit baudot.

When an operator wishes to transmit data, the buffered data is entered and sent down the communication line. The other end of the line must be ready to receive data at all times. Contention problems, when operating in half-duplex mode, must be resolved procedurally.

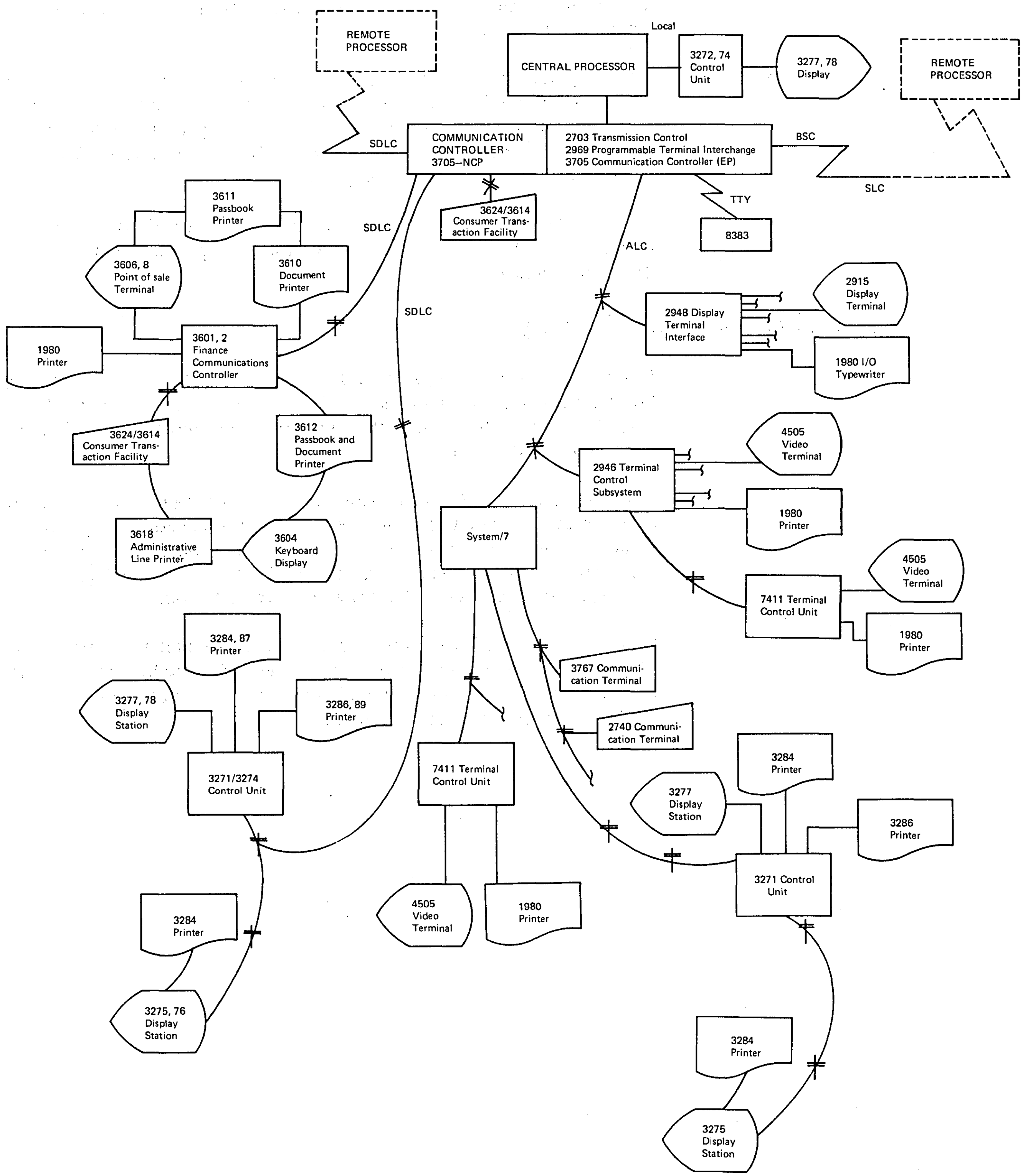


Figure 6. Composite ACP/TPF line configuration

ATA/IATA Link Controls

Two types of line control procedures, synchronous and asynchronous link control, have been specified by the Interline Communications Committee of the Air Transport Association of America and the International Air Transport Association, organizations representing both domestic and international airlines. These procedures were developed to meet the requirement for processor-to-processor communication between different airline systems. The functions provided by both types are similar, but the synchronous version has an advantage in terms of transmission speed and character set size. Each line control procedure is described below. Also, see the section titled *Computer Networks* for further discussion of the ACP/TPF support for these transmission techniques.

Synchronous Link Control (SLC)

SLC operates on a point-to-point basis over full-duplex, voice-grade lines at speeds up to 9600 bits per second. Up to seven lines can be combined to form a *link* between two CPU's. More than one link may be used to connect two CPU's. There is no polling associated with SLC lines. Instead, each CPU continuously listens to its receive lines and, when it has data to transmit, sends it via the transmit line. Whenever there are long periods of inactivity, dummy messages are transmitted telling the receiver that the line is still operational.

Data and control characters are eight bits long (seven data plus parity). Each message contains a block check character for error detection. In addition, all messages are sequenced, allowing each CPU to detect missing or spurious messages and automatically institute correction procedures.

Asynchronous Link Control

This procedure also uses full-duplex, point-to-point lines for intersystem communication. A five-bit character set is provided with no parity checking, and the lines usually operate at 75 bits per second. Message sequence numbering is an optional feature, but predefined acknowledgment messages are normally used to detect erroneous transmissions. While this line control is still used extensively today, the growth in volume of intersystem message traffic has made SLC (with its higher throughput capability, better error checking, and larger character set) a much more attractive alternative for many users.

Communications Program Support

The programming in ACP/TPF that supports the communications hardware devices and line control procedures described previously can be thought of as consisting of two elements. The first consists of those functions necessary to physically control the transmission and receipt of messages on the terminal network. The program components that fall into this category perform such functions as polling devices on a line, assembling and editing incoming messages, queuing and transmitting outgoing messages, and carrying out all the associated error detection, recovery, and logging. The specific functions performed by this class of programs are quite straightforward in that they must generally meet predefined requirements called for in the hardware or line control specifications.

The functions included in the second element of communication program support are not nearly as obvious. These deal with the interfaces between ACP/TPF and the application programs that allow processing of input messages and the initiation of output to terminals. Whereas the preceding paragraph addressed *physical control* of the communications network, the control program functions discussed here deal with *logical control* of the network.

When an input message is received, the control program puts it into a standard data format. It then determines which application should be given control for subsequent processing and places the message on a queue to await dispatching. (The term *application* indicates a class of related programs needed to accomplish a specific function, such as reservations, car rental, check guarantee, etc.) All messages are processed on a first-in/first-out basis within priority category.

Output messages, which may be responses to input transactions or unsolicited transmissions, are formatted by an application program assuming a general class of terminal rather than a specific type. Classes include teletype, video displays (3270, 4505, 2915), keyboard/printers (1977/2740), and receive-only printers. The 3270 represents a special case in that it can be used in either native or 4505 emulation mode.

Application Transparency

As shown in the preceding figures, 3270 devices can be physically connected through a System/7, locally attached via a 3272 control unit on the byte multiplexer channel or directly attached to an SDLC line through the 3705 with NCP; the 3270 in emulation mode can be used to access applications written to use the 4505 display. Program support allows the mode of attachment to be transparent to the application program. Thus, applications written to use the 4505 display will work unchanged on a 3270 used in emulation mode.

Mapping/Paging

An input and output data mapping service is available to the application program that will translate incoming 3270 messages to a field-addressable data record and perform the reverse function on output. This relieves the application programmer of the task of interpreting or creating the complex data streams that are required by the 3270. It also makes possible the redesign of screen formats for either input or output without affecting application programs. Mapping and paging services support a screen size up to 1920 characters.

The addition of the 3600 feature allows ACP/TPF-based applications to interact with the various terminal devices comprising the 3600 Finance Communications System.

Log On

The system will manage the receipt of input messages from terminals and the transmission of response messages in much the same way as it does for supported terminals. The major changes, aside from unique data formats, involve the interface to the 3705 Communication Controller using the network control program and the addition of message sequence numbering to improve message integrity. A logical unit (LU) may be permanently logged onto an application at network generation time.

Support is included for handling multisegment long messages both to and from the terminal controllers. The extensions to this support, provided by the 3600 feature, allow more generalized multiple segment message transfer and can be used for such things as data transfer from the terminal controller's local data base to the central system.

Unsolicited Messages

Facilities are provided for delivering unsolicited messages (an output message with no corresponding input). Usability of this function is improved by the ability to identify 3600 terminals by their node name. This provides a degree of independence between the application and the physical configuration of the network.

Cluster controllers in the SNA network may be loaded (or dumped) online. As an option, ACP/TPF provides message recovery. This option assures that once an input message is passed to an application, the input is processed to completion and the response successfully transmitted to the LU.

Restart takes advantage of the fact that the 3705 with NCP and the terminal controllers are able to maintain status information across a host system interruption. This allows a *warm start* to be performed with recovery of messages that were in transit between the terminals and the host CPU.

Computer Networks

Generally, the term *computer network* refers to a group of data processing systems that can communicate with one another in some fashion but are not dependent on any one system for overall control. This definition allows networking to be differentiated from multiprocessing, in which one control system is required to assign work to and monitor the activities of the other processing units. Another distinction of networks is that the systems comprising the network need be compatible only in respect to the technique used to exchange data between processors and may, in fact, use totally unique hardware and software.

ACP/TPF supports a variety of network functions that provide a wide choice in the way the user may configure multiple systems into a network. Since ACP/TPF is a transaction processing system, these functions are aimed at the interchange of relatively short messages on a realtime basis. However, the transfer of large volumes of data base information can be accommodated as well.

The ACP/TPF message router package (support of BSC and SLC links) and ACP/ACF (support for SDLC links), both system-supplied support for establishing dynamic interprocessor connections via *LOGIN* message, are mutually exclusive. The ACP/TPF support of computer networks includes the use of the following data link protocols:

- Synchronous Data Link Control SDLC
- Binary Synchronous Communication BSC
- Synchronous Link Control SLC

The objective of the ACP/TPF support of computer networks is to provide an ACP/TPF installation with the ability to:

- Forward a message received from a terminal via an ACP/TPF-supported data link protocol to an application program that may be executing in another domain. The other CPU may be utilizing an operating system other than the ACP/TPF system.
- Receive a message that may be from another domain and forward the message to a designated terminal or application within the receiving domain.

ACP/ACF

The SDLC data link protocol provides the capability to link together multiple System/370s through the use of ACF/NCP/VS, and SNA advanced communications function ACF multisystem networking facilities supported by the ACP/TPF system. The SNA multisystem network support is simply called ACP/ACF.

ACP/TPF as a cross-domain resource manager CDRM will communicate with ACF/VTAM, ACF/TCAM, and other ACP CDRM's to control cross-domain sessions utilization, terminations, and recovery. The ACP/ACF feature supports cross-system message routing, through which data may be transmitted across systems to its destination without host intervention, after initial session establishment.

For an illustration of ACP/ACF facilities, see Figure 7.

- Terminal A may enter into session with application A or application B but *not* with application A *and* application B.
- SNA/ACF in support of multisystem networking means that if terminal A enters into session with application B, then normal data flow, after the session is established, will not require any assistance on the part of CPU-A; all messages will be routed to CPU-B by the NCP in the communications controllers. However, CPU-A is required to establish the session, since terminal A is in the domain of system A.

BSC/SLC

The ACP/TPF programming package that provides the computer network support for both binary synchronous communication and synchronous link control data link protocols is the message router. The logging facility (or LOGI) is that portion of the ACP/TPF system which is responsible for establishing connections between terminals and application programs in the ACP/TPF system or in another domain and for controlling the flow of messages between them. Application to application connections are also made possible by the ACP/TPF routing facilities. The routing program allows the transfer of data from an origin point to a destination point anywhere in the supported network through the use of BSC and SLC data links.

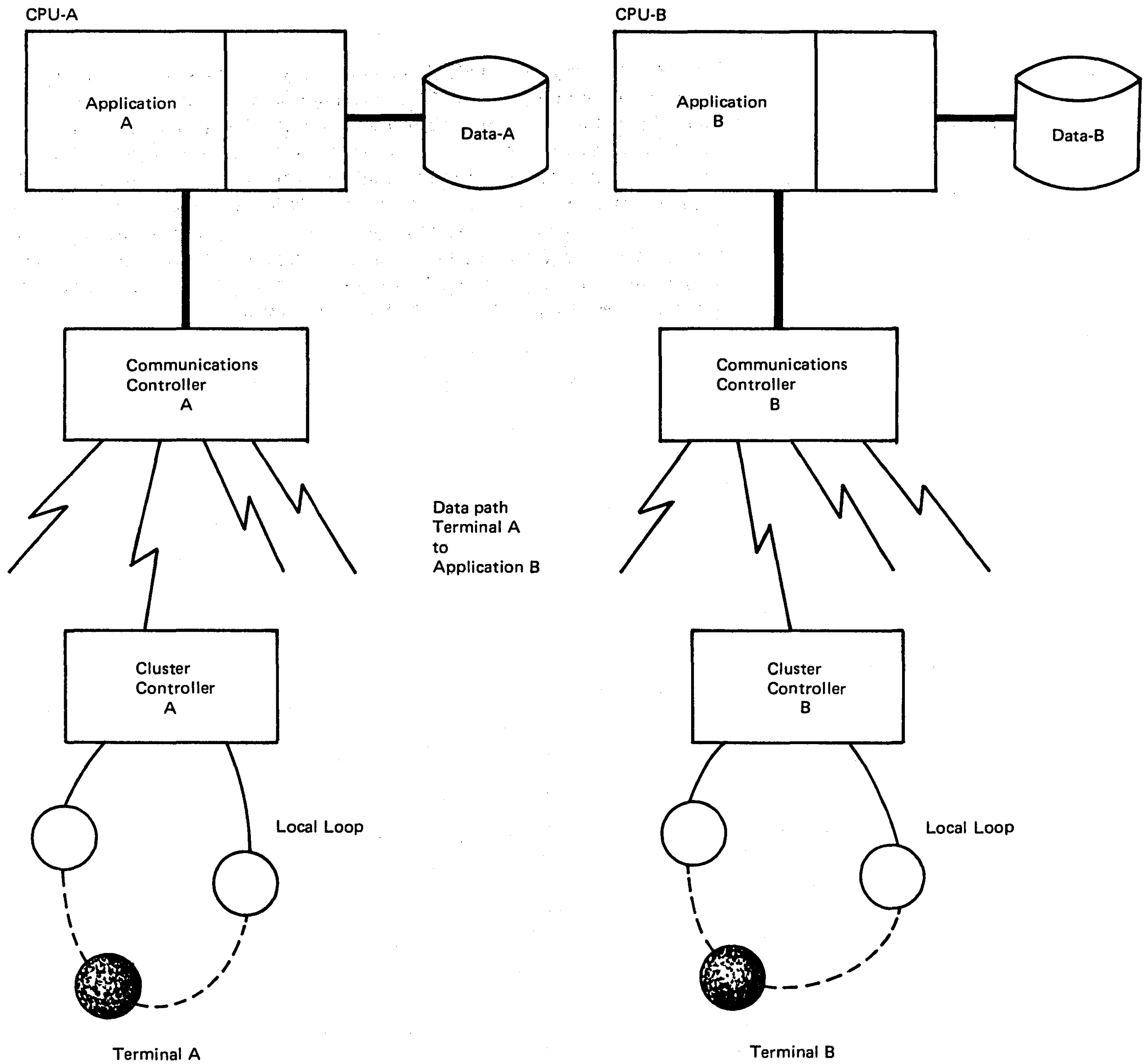


Figure 7. ACF network

Cohabitation

CPU selection for a realtime system is usually based on an estimate of the peak processing requirement several years in the future. Since these systems are generally expected to grow in terms of message volumes and new applications, the peak load experienced in the first few years of operation will be below that which the CPU is capable of handling. Also, the anticipated peak period often lasts for only a few days each year and, even within the day, the processing load can vary considerably. For example, the system may be operated on a 24-hour basis, but the number of users and the amount of transactions created by them often drop sharply over the evening and nighttime hours.

All this adds up to a considerable amount of CPU power that is unused when a system is dedicated to online processing. One way of utilizing this unused CPU power is to provide a facility that allows multiple operating environments to exist in a single CPU (cohabitation). Thus, when one environment is not demanding system resources, another can make use of them. (See Figure 8.) Cohabitation of ACP/TPF with other operating environments is effected using a facility of ACP/TPF known as the hypervisor. A similar method of cohabitation for other operating environments is called Virtual Machine Facility/370 (VM/370). The hypervisor differs from VM/370 in that it is dedicated to maintaining the high performance characteristics of ACP/TPF.

Hypervisor

Cohabitation of two distinctly unique operating environments (ACP/TPF and OS/VS or a native ACP/TPF and a test ACP/TPF) is achieved by use of a software feature, the hypervisor, in conjunction with System/370 relocate hardware. The operation of the native ACP/TPF portion of the system is unchanged, but to allow concurrent execution, the hypervisor must simulate a System/370 machine for use by OS/VS or a test ACP/TPF. The hypervisor intercepts all system interrupts and, depending on whether the native or simulated system is in control, either passes the information on to the proper interrupt handling program or queues it for later processing.

The hypervisor also allows sharing of the system channels and main storage. Individual control units and their associated input/output devices can be assigned to either the ACP/TPF or HYPERVISED system, and this assignment may be changed in the course of system operation by the console operator. Main storage, on the other hand, must be partitioned into two fixed-size areas, one for each system, when the processor is initialized. However, it is possible to maintain multiple copies of OS/VS and to initialize the system using any one of them. This allows the user to reconfigure the main storage partitions and to have available a number of different OS/VS systems (that is, different release levels of the same control program and/or both OS/VS1 and OS/VS2 programs).

In addition to providing more effective usage of the online processor, the hypervisor can also increase the productivity of systems used for program testing. It has often been necessary for ACP/TPF users to dedicate large amounts of time on an offline system for testing and debugging new

ACP/TPF-based applications. This testing requires a sizable configuration to perform a relatively small amount of processing. By using the hypervisor, the user can test ACP/TPF-based applications on a system that is the effective online ACP/TPF system. The hypervisor favors the native ACP/TPF system and simulates either OS/VS1, OS/VS2 (SVS), OS/VS2 (MVS), or an ACP/TPF system. Use may also be made of the virtual machine assist (VMA) feature to improve performance, if it is available on the CPU.

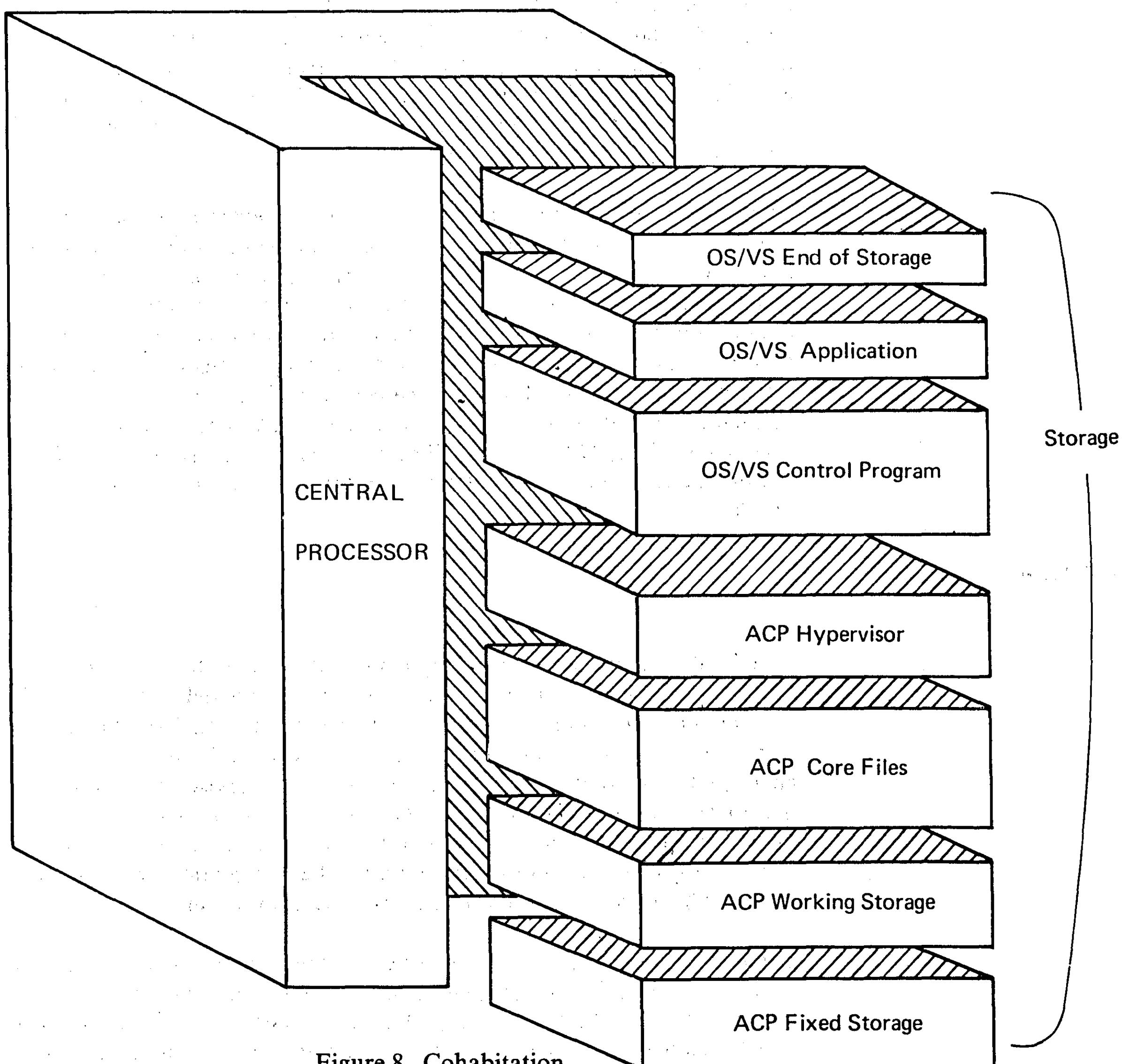


Figure 8. Cohabitation

Other Control Program Functions

Restart/Switchover

ACP/TPF consists of the instructions and data required to control and service application programs. This section outlines the methods of storing and restoring ACP/TPF code from file to main storage. The majority of ACP/TPF, used at a high frequency, is kept in the fixed portion of main storage. The remainder of ACP/TPF is called into working storage when requested. A copy of the fixed area of main storage, a restart program, and a routine called IPL (Initial Program Load) are kept on file storage for the purpose of restart.

Assume that main storage does not contain any meaningful data and a restart of the system is desired. An operator initiates the IPL routine. The restart program is brought into main storage. This program transfers control programs and data from file storage. At this point, ACP/TPF begins to take control. Application programs and data for fixed storage are retrieved from files, and those applications that must be continued (maintenance types of applications not requiring responses to an agent) will be restarted. The last step in restart is to notify the operator that the system may be put in normal status.

The foregoing operation, called restart, may be initiated automatically by ACP/TPF when it determines that such action is required or by an operator taking a manual action. The length of time to restart is approximately one minute. ACP/TPF systems are usually duplexed to minimize exposure to equipment failure. The switching of equipment other than CPU's may entail some physical switching, an operator action, or a restart. The switching of CPU's, however, is given a special term, *switchover*. Using ACP/TPF, switchover is accomplished by the operator taking a manual restart action at the other CPU.

Keypointing

The proper operation of a realtime system demands that critical data be saved over an interruption of activity. ACP/TPF meets this demand by updating the file storage copy of critical data whenever that data is modified in fixed storage. Among items that could be considered critical are file status and file storage references. The data considered critical is grouped into special records called keypoint records. The data fields are called keypoints, and the process of filing the associated keypoint record whenever a keypoint is updated is called keypointing. A subsequent restart, therefore, contains the most recent data.

In some cases, application progress must be preserved over an interruption. The applications with this requirement use the same techniques as ACP/TPF uses in keypointing. The records with critical application data are called global records, and the data fields are called globals; however, the process is still called keypointing. ACP/TPF handles the keypointing of global records.

Entry Management

ACP/TPF is an interrupt-driven system. An interrupt is an action that causes the CPU to leave its normal instruction sequence. Interrupts that can occur include the input of a message into the system, the completion of a file request, and the execution of a macro of the supervisor call type.

When ACP/TPF finishes an entry or leaves it to process another, it uses a control routine to find the next entry for processing. If there is no work to do, the system leaves the control routine and enters the WAIT state. There are four levels to be examined for entries. Each level is examined in turn and must be empty before the next is examined. A priority system is thus established; however, the priority is generally assigned for system considerations rather than application function.

Table 1 shows these four levels, with the criteria used to place an entry in each level. The objective is to give priority to entries currently in process before starting new entries. All entries are given equal consideration, allowing efficient usage of system resources. The fast response of all individual entries is attributable in part to this technique.

Table 1. Priority Levels

Level	Name	Reason Entry Placed on Level
1	I/O list	Used by control program to assign top priority for certain critical I/O list manipulation and processing
2	Ready list	Completion of I/O request (file request, etc.). Internally created entries
3	Input list	Input from communication lines. Internally created entries
4	Deferred list	Application requests deferred processing. Internally created entries

As can be seen, the primary levels are concerned with entries already in the system. Assigning high priority to these levels helps to reduce response time and helps prevent multiple inputs from overloading the system. Applications may defer processing or assign low priority to an entry. Once an entry is activated from any level, all subsequent placement will be high priority unless the application takes an overt action to reduce priority. To reiterate, the primary thrust is to speed an entry through the system. Entries are taken from a level on a first-in, first-out basis.

Working Storage Management

As stated in the discussion of working storage, this area of main storage is divided into working blocks. ACP/TPF controls the distribution of these blocks to entries as requested. Normal operation entails a somewhat simple bookkeeping type of function. As activity increases near system capacity, however, working storage may become scarce. This circumstance, normally of a short duration, is called peaking.

If nothing is done to slow down the system during peaking, storage overflow may occur, and the system will be interrupted. ACP/TPF helps alleviate this condition by controlling the input of new entries, based on the number of working storage blocks available. To accomplish this, ACP/TPF controls the frequency of polling terminals for new inputs. Reducing input and giving high priority to active entries levels storage use. Another technique is to prevent the activation of low-priority entries when storage is highly utilized, even though there is little CPU activity. This situation occurs when there is abnormally high activity on the files.

Program Management

Programs may reside in either fixed storage or file storage. In either case, they must be segmented to fit within a large file record (1055 bytes) or be split into sections that will fit. (Small programs may be placed in 381-byte blocks.) Communication between segments of a program or between programs is accomplished via macros. The interface allows one program segment to transfer control to another (program A goes to B, which returns to A, etc.). Application programs are not aware of a segment's residence (file or fixed storage).

If an application requests to transfer control to a program and the program resides in main storage, ACP/TPF transfers control to that program. If it resides in file storage, ACP/TPF checks to see whether the program is temporarily residing in working storage (it may have been previously requested by this or another entry), and, if so, ACP/TPF transfers control. If the program must be retrieved from files, the retrieval mechanism is used, and the transfer is not effected until the retrieval is complete. On completion, an item is placed in the ready list (defined in Table 1), and, subsequently, control is transferred to the program just retrieved. If a return to the entering program was expected, ACP/TPF anticipates the return and effects it when requested.

Programming

General

The ACP/TPF package consists of macros, programs, and data records required to perform the functions of the system. Components that are part of the online operation are referred to as the realtime or online components. Programs that are part of the online system are also termed operational or application programs. The remainder of the software programs are referred to as offline programs.

This section outlines the interfaces between application online programs and ACP/TPF. The interfaces are effected through use of a storage block called an entry control block (ECB) and through macro interfaces.

Entry Control Block (ECB)

The ECB, required to perform the online functions, is a block in working storage associated with each entry in the central processor for the in-computer life of that entry. It is held in the system until processing of the subject entry is completed.

The ECB is used by ACP/TPF and the application programs for communication with one another. It is used by the application programs as scratch pad storage for interprogram communication.

The ECB is crucial to the manner in which applications are written. It represents a data control block for an application program and is the mechanism that permits all application programs to be reentrant. This means that the identical executable portion of an application program may be invoked for processing different messages simultaneously. All references to system storage and all dynamic manipulation are accomplished through the ECB. This is equivalent to calling for a new copy of code without using any additional storage for the code.

Macros

There are, in general, two types of macros, the executive and the declarative.

An executive macro generates either code or data that is incorporated into the program being assembled. One type of executive macro, called control program macros, provides a linkage to the ACP/TPF service routines. Generally, this implies the generation of a supervisor call type of instruction with parameters specifying the desired service. A second type of executive macro is that which is termed application macros. This type of macro, which may generate a sequence of machine instructions, corresponds more closely to the operational definition than the control program macros.

A declarative macro produces information used by the assembly process in the generation of code. In this class are data macros and equate macros. The data macros generate definitions for commonly used data records. The equate macros generate statements that equate numerical and special alphabetic values to symbolic parameter names.

Control Program Macros

These macros are issued by application programs to the control program for services.

- Enter/Back Macros

  Application programs transfer control to other application programs through the use of enter/back macros. A request may be made to enter a program with an expected return or to enter a program with no expected return.

- Main Storage Allocation Macros

  Included as a part of the ACP/TPF system is a portion of main storage used by application programs for entries that require additional processing space. When an application program executes a macro to get storage, the control program assigns one storage block to the entry. Another macro provides the application programs with the ability to cause release of main storage blocks when they are no longer needed.

- File Storage Allocation Macros

  The file storage medium has a pool of record addresses available for use by application programs. An available file address may be obtained by executing a macro. File addresses are returned by executing a release-type macro.

- Find/File Macros

  A series of macros have been provided to allow the application program to reference and update data stored on files. The file reference macros allow the application program to initiate a read and write of a record. A wait macro is used to ensure completion of the request.

- Rout Macro

  Application programs request the transmission of output messages via a macro. Before transmission to the terminal, ACP/TPF translates the message from the internal system code to the appropriate terminal code.

- Create Macros

  An application program may create new and independent entries in the system and assign priorities for initiating the processing of the entries. The create deferred macro is used to create an entry and defer its processing to a nonbusy period. The create immediate macro is similar, except that the created entry is given the highest priority. The create time initiate macro creates an entry by which the control program transfers control to the specified program at the indicated time.

- Defer/Delay Macros

  The defer macro is used to defer processing of an entry until the amount of activity in the system is sufficiently low to allow for completion of this low-priority task. The delay macro is used to temporarily delay the processing of an entry and allow processing of other entries.

- Tape Macros

  The ACP/TPF system has two kinds of tape files. Realtime tapes are write-only tapes available to all entries in the system. General tapes are separated into distinct sets, each set exclusively associated with a particular function. General tapes can be read or written.

- System Error Macro

  An application program issues a system error macro when it encounters an error to be recorded. This macro specifies the data to be output if this error is encountered. The data will be output only on the first occurrence if this option is selected. Subsequent occurrences will indicate only that an error occurred. Output is to a realtime tape for offline processing.

Application Macros

An application macro may represent a set of executable instructions to perform a repetitive function unique to a specific application. Application macros may, in turn, use declarative-type macros. Two macros required by every application are the begin and finis macros.

- Begin Macro

  The begin macro provides the necessary formatting for the program header items. The information formatted by the macro is utilized to load the programs onto the online system from an offline OS library as well as to identify standard system names. All online application programs are coded with a begin macro as their first statement.

- Finis Macro

  Finis provides the necessary assembler information to complete assembly of a program. All online application programs are coded with a finis macro as their last statement.

Test Facilities

It has been said that, in relation to realtime systems, programs are tested individually, tested with other programs, tested with a large data base, and tested in a live environment, and when finally installed are not fully tested. In fact, all that can be stated is that all known errors have been removed.

Many factors contribute to the truth of this statement. One of these is the large number of permutations of any particular entry. The data base is constantly changing. Agent messages have many variations. Unique sequences of events occur; thus, it is impractical to test all possible conditions.

Another factor is that errors may go undetected for considerable lengths of time because of the delayed appearance of symptoms. A program may erroneously update a data record that is not examined for days or months. At the time of error discovery, the cause of error is difficult to determine, the program creating the error may have been modified, and the circumstances under which the error occurred may be impossible to determine and duplicate.

An error may propagate itself through the system before a symptom appears. In these cases, it is difficult to determine the original culprit, and in the process, many data records may be corrupted.

Finally, a new program or modified program may be tested without error; however, it or its results may cause another program or function to fail. This possibility dictates the need to reverify all existing functions in those cases where a modification may affect other programs.

All of this is intended to point out the complexity of realtime systems and their need for extensive test facilities. The test facilities of ACP/TPF, designed with the foregoing factors in mind, do not test all cases and possibilities; however, their objective is to test a program to the point where there is reasonable assurance that the program will not fail and to provide a facility for detection, analysis, and recovery from failures. These facilities, however, can be no better than the procedures used for their administration and control.

Update/Compilation/Loading

As a program is developed, it undergoes several iterations because of design changes and errors in compilation and test. After installation, additional improvements will effect further changes. ACP/TPF test facilities use the OS library functions in conjunction with rigid procedures to maintain program libraries, ensuring that certain libraries have restricted usage and that the individual program code is controlled at the library rather than the programmer's desk.

Compilation in ACP/TPF consists of compiling Assembler language programs. This process is controlled by cataloged JCL procedures that ensure that the object code is in a form compatible with the online system. The system loader consists of programs that transfer object modules and data to an ACP/TPF system from an OS library. The loader has the facility to transfer selected versions of programs and to modify (patch) a program in the process of transfer.

Test Environment

Whether testing a program by itself or in conjunction with other programs, an environment must be created for each particular test case. This environment may consist of a test vehicle, input messages, a *driver* type of program, other programs, and/or data records. The creation of the environment must be simple in order to eliminate programmer time in creation and eliminate the need to test the vehicle used to test the program. The programmer must have the ability to call on the efforts of prior users and on masses of data.

The system test compiler (STC) provides the means of amassing those elements (with the exception of the test vehicle) required by a programmer to provide a test environment. Its function is to collect all the messages, programs, and data requested by the programmer and place them on a sequential data file for input to the test vehicle. The programmer has the facility to:

- Call selected programs, with the option of modifying each
- Specify unique messages to be used in the test
- Specify unique data records, either in absolute or symbolic form. (The same labels used in defining data record fields in coding can be used to define fields for test data.)
- Specify one or more messages or data records located on a library of messages and data records. (This library expands during the life of a system as new data is added because of increased function, etc.)
- Specify messages and data records on the library, modifying each for the particular test case

(Note: An environment is created for initializing a system. This particular environment is located on a tape (sequential file), often referred to as a pilot tape.)

Test Vehicle

The vehicle used to test a program must simulate real conditions as much as possible, allowing some flexibility for test procedures. The vehicle must also provide for the isolation of each test case on an optional basis. The vehicle used with ACP/TPF is the program test vehicle (PTV).

PTV can be best thought of as an additional program loaded with an ACP/TPF system. This program, when activated, controls the execution of test cases. Although a system can operate normally with PTV loaded and inactive, it is not recommended for live systems because PTV does occupy a considerable amount of fixed main storage and thus affects performance.

As can be seen, the first criterion of the test vehicle is met by using the actual control program and a special program (PTV) to provide flexibility. The satisfaction of the second criterion, that of isolation, will be apparent as the operation of PTV is outlined in the following paragraphs.

Each test case is handled individually. The first task of PTV is to load programs and data collected by STC for the environment. The messages, specified in the environment, are readied for simulated input as if by an agent. PTV feeds these messages to the system for processing. Options allow input of either a single message at a time or multiple messages running simultaneously. The original copies of data records, modified during load or test execution, are copied to a data file for possible restore after completion of the test, thus maintaining an unmodified system for the next user. Special output for programmer analysis can be requested, providing data on the progress of test cases.

System/Regression Testing

System testing requires a large amount of data and messages. Although these can be specified in a test environment, the specification of data records can become rather cumbersome. To alleviate this situation, miniature copies of the *live* system are created. (The live system is that system actually performing the application function.) These copies have a small number of physical files and reduced-size logical files. A data base is created by loading an environment and/or processing messages until a basic system is present. From this point the test environment need only specify data that differs from the base system. In reality, once a base system is created, it is used for all phases of test. System test entails the running of a large number of messages against the base.

As new or modified programs are added to a system, the system must be checked for the programs' effect on current message processing. This is accomplished by running the new or modified programs and their associated messages against a background that is the previous system test. This is called regression testing. When the new items have tested successfully, they and their environment are added to the system test and its environment.

Online Aids

These aids allow a programmer to analyze a program operation during its progress. This is accomplished by providing a printout of storage and/or register contents whenever a linkage type of macro is executed (TRACE). A trace can be requested for a particular program or for entries from a particular terminal. Linkage macros traced are specified by the request. A trace can be requested during live as well as test operation. All traces and printouts of main storage under ACP/TPF control are formatted for ease of analysis. Areas with symbolic names are labeled. Data blocks associated with each entry are grouped with that entry.

Throughput and Performance Considerations

Factors that govern the capability of the system to process a particular message rate include the model of CPU, communications techniques, amount and organization of main storage, and number, organization, and type of file storage devices (in terms of volume and accesses). All these must be balanced to optimize a system. The balance is achieved by understanding the concepts of the system and using the data collection program to *tune* the system.

System Performance

System performance functions (data collection/reduction) are designed to provide operational data on all significant activities involved in processing messages. By analyzing the reports generated by this package, the user can determine how efficiently the installation is running, discover where the bottlenecks are, and find clues as to what changes in system allocation (main storage, file, lines, terminals) could improve system performance.

The major goals in view throughout the design phase of this program were to:

- Provide a tool that can be used during the installation and postcutover phase to tune the system to peak efficiency

- Provide a means by which periodic monitoring of system performance can be readily achieved

- Provide sufficient statistics so that long-term trends can be observed from the runs, thus providing the base for predicting growth in system load and justification for future expansion

Data Collection

In any data collection program involving realtime operations, the prime factor to be considered is the load and interference the collection programs themselves will impose on the system being measured. Minimum impact on normal processing operations is essential. Since this impact cannot be zero, every effort must be made to know the extent of the influence data collection has on the key parameters to be analyzed.

Collectors are run in a sampling mode, allowing multiple types of data to be collected while avoiding significant interference with message processing. In this mode, each collector can be run for a short interval during a larger period in a sequential manner. There is also a continuous mode of operation available, where any one collector can be run with no time spacing between the interval samples. All collection programs write the data gathered to an online tape. No attempt is made to reduce any of the data online, since this would defeat the objective of causing a minimum impact on the system statistics being measured.

The three basic techniques used in collecting data in ACP/TPF are those of:

- Reading out counters that are imbedded in, and updated by, ACP/TPF

- Intercepting specific events, such as file macros and program enters, when those collector programs are active

- Dynamically sampling parameters that fluctuate with time, such as I/O device queues and core blocks in use

Data collection can be run to provide a history file of key system parameters, such as milliseconds per message, file accesses per message, core usage per message, enters per message, message rate, and message length. Any changes in system configuration, programs, core allocation, etc., should be carefully documented and dated so that the performance aspects of the system obtained for data collection can be correlated with these changes. The foregoing information, plus the history and trend of passengers boarded, can be used very effectively to predict the need for more memory, channels, files, lines, or terminals, or the need for more CPU capacity.

Data Reduction

All data reduction associated with the system performance package is done under control of an OS/VS system. The delay involved in reducing the tape offline after the data has been gathered can really be considered an advantage over a technique that would provide *instant reply* or immediate printout of results online. The analysis phase cannot be approached with haste. Snap judgments made from too little data usually cause more problems than leaving the system status quo. For instance, ACP/TPF has many carefully designed cutoff levels as protection against overload conditions, and any tampering with the adjustment of these levels must be weighed in light of the many interacting factors involved.

The primary objective of the initial analysis phase for any ACP/TPF system is to establish the normal state limits for each of the key factors affecting performance. Once these factors are set and agreed to be realistic, then a periodic check on the system becomes routine.

The data reduction reports were designed to be used by an analyst familiar with ACP/TPF, but not necessarily a statistician. However, for the convenience of those so inclined, frequency distribution reports, including means, standard deviations, variances, etc., of each parameter, are available.

The initial analysis of any collection should always start with summary reports. The summary reports provide the key performance data required for history and trend analysis. When investigating a problem area, the more detailed plot and distribution reports or the specialized reports of the file and message reduction programs are used. The plot reports, showing the value of each parameter sample in chronological order, can be very effective for cross correlation of the cause-and-effect relationship between parameters.

Control, Audit, and Reconstruction Procedures

File Storage Support

An inherent characteristic of the ACP/TPF environment is the relatively large volume of application data records that must be readily available and organized to enhance performance. These factors dictate the need for special support programs. Included in support programs are those that help ensure against permanent loss of data records, facilitate the expansion of file storage, and assist in maintaining system file storage integrity.

Capture/Restore

The constant availability of file storage exposes it to the effects of software and/or hardware malfunctions. This exposure could be removed by eliminating all errors (a somewhat improbable if not impossible task) or reduced by ensuring that critical data lost because of these factors can be replaced.

A basic insurance against loss of critical data is the maintenance of file storage copies on auxiliary storage media. These copies, taken periodically, are called captures; the process of copying is called capture; and the process of restoring the capture to file storage is called restore.

A full restore restores all file storage, whereas a partial restore restores the data on one or more devices, but not all. (Remember that a physical device contains portions of many logical files.) The period of capture for airline reservation systems varies from one per day to one per week, depending on the user's requirements and/or confidence in the system. A capture is often used to ensure against destruction of file storage because of the introduction of new programs.

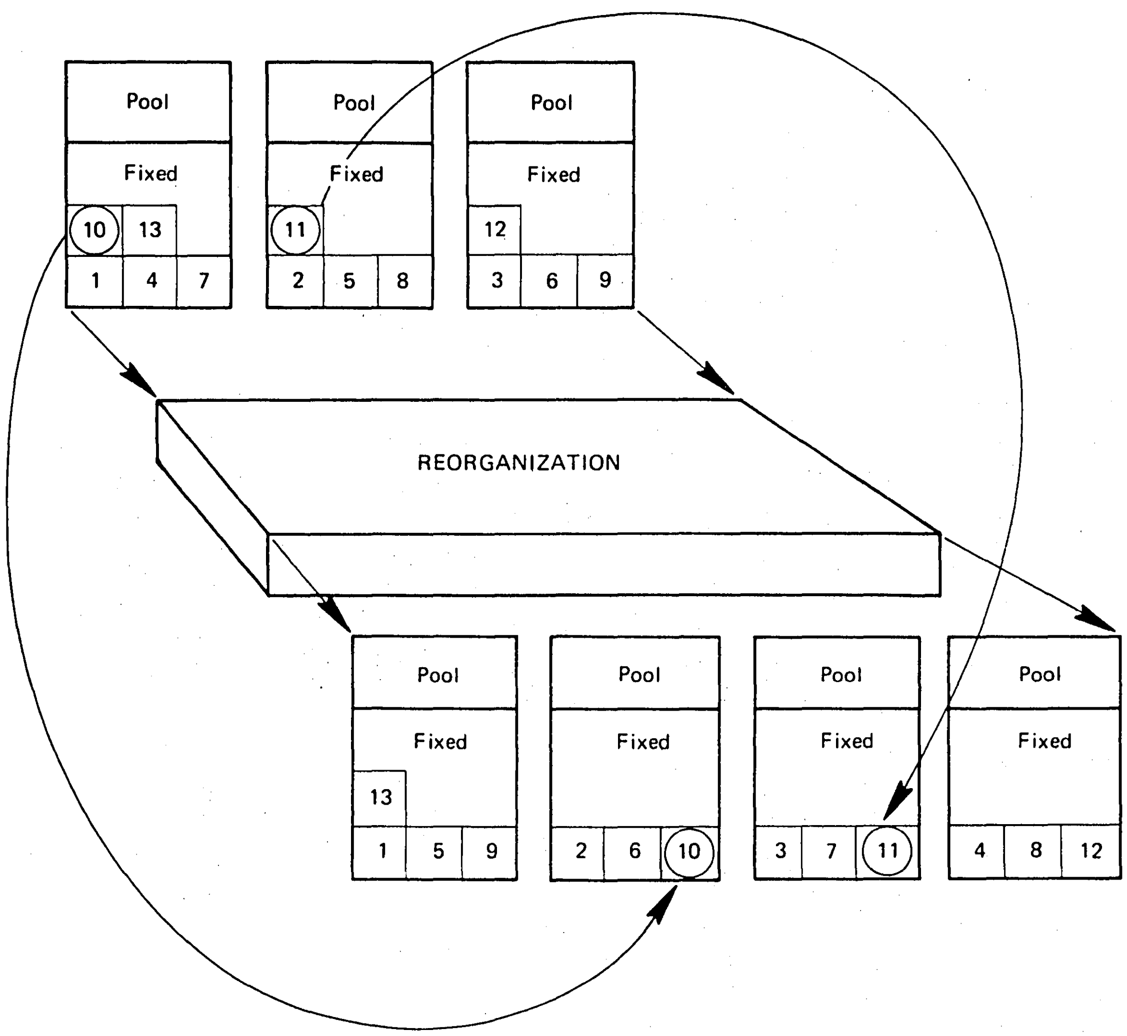
One means of capturing file storage is to stop the system and then, using standard offline utilities, copy to disk or tape. This procedure prevents any use of the system during capture. A second method is to use the online capture supplied with the system. This program captures the files during normal system operation. It is run during periods of low activity. The data on each file storage device is copied to magnetic tape. Simultaneously, a separate tape, called an exception tape, collects a copy of all records modified during the capture process.

Restoration of a full system restores the system to the time and date of the completion of the capture. Further programs or procedures are required to reconstruct the files that resulted from activity between capture and time of restore. These programs usually fall into the application area, since the data considered critical is highly dependent on the application.

Fixed File Reorganization

Normal system growth usually dictates an increase in the number of direct access file storage devices. As stated earlier, file storage may be divided into areas of pool records and fixed records. The addition of new devices provides an increase in both areas. This does not present a problem in the pool areas, since the addresses of the new records need only be given to ACP/TPF for handling. The fixed area presents a different problem. Since a logical file uses all devices, the records must be distributed over all devices to take advantage of the added files.

A program provided to accomplish this is the fixed file reorganization program. This program collects all records in the fixed area, using their symbolic address with the old system definition. The records are written back using the same symbolic address with, however, the new system definition (including additional devices). Since the application programs use symbolic addressing to reference fixed-file storage records, the reorganization is transparent to the application. (See Figure 9.)



This file set is being reorganized because of the addition of a device (total now 4). Note the change of each record's residence, retaining the numerical sequence of each. Records 10 and 11 are highlighted.

Figure 9. Fixed file reorganization

Recoup

During the operation of an ACP/TPF system, a pool record's address may be lost by not being returned to ACP/TPF for further use when the application is through with it. In this case, the amount of available pool records is diminished. Correction of application errors lessens the possibility; however, whenever the system *goes down* because of error, some entries may be partially processed. The operators associated with these entries will correct the problem functionally, but pool addresses may be lost in the process.

The recoup program is provided to alleviate this situation. This program examines all pool records and determines which are both valid and active. The remainder are made available to the system for further use. Obviously, the recoup program must interface closely with the application program to determine which records are valid. A by-product is information about program errors that might have led to a loss of addresses.

Diagnostics/Maintenance

To effectively perform corrective maintenance on a nonoperational device, information is required about the nature and environment of the failure. A portion of this information is in the form of a history of intermittent failures. Another portion of information is the output of special programs designed to diagnose equipment and isolate failures by exercising the failing device with specific routines. These programs are called diagnostics.

Central Site

A history is kept as to intermittent failures of central site equipment. These are logged to a realtime tape for processing on the offline system. A customer engineer, by analyzing this history, can predict future problems and take measures before they occur. All diagnostic routines are run from the offline system. This prevents overloading of the online system and reduces the exposure of their interference with the online data base. Since installations are duplexed at the central site, the online function can continue during the period of diagnosis and repair.

Remote Equipment

Remote equipment history and diagnostics are maintained through the online system. The load to the system associated with this function is negligible. The customer engineer can request counts of line and adapter errors for lines attached to either a 2946 or a System/7 (RM). These counts are available at any terminal on the system.

The PARS online remote terminal diagnostics consist of file-resident routines designed to test remote equipment by exercising individually specified terminals. They may be used by a customer engineer to determine whether all valid characters can be displayed, all valid commands are accepted, the proper function is performed, line stability is maintained, etc. The output messages generated act as visual aids in identifying terminal malfunctions.

Normally, output will consist of a test pattern sent to the terminal being tested. When necessary, error messages will also be sent to the terminal advising the user of the error condition (for example, faulty input message format).

Programming Systems

ACP/TPF provides an operating environment and therefore does not require an operating system for its execution. However, ACP/TPF requires OS/VS1 or OS/VS2 for offline utility functions. ACP/TPF requires those versions of OS/VS1 or OS/VS2 which support subsystem support services (SSS) if support for the IBM 3600 Finance Communication System is used. See the appropriate SRL's for the required configuration needed to support OS/VS1 or OS/VS2.

ACP/TPF has been developed and source code distributed in Basic Assembler Language (BAL). A number of ACP/TPF support programs are written using PL/I and require the PL/I optimizer for compilation.

ACP/ACF function of ACP/TPF requires that NCP/ACF feature reside within the 3705 communications controller supporting SDLC lines.

System Requirements

ACP/TPF is designed to operate on all System/370 machines from System/370 Model 135 to 3033 Processor, with a requirement for a minimum of 512K real memory. The minimum hardware system configuration required to execute ACP/TPF:

- System/370 Model 135–512K Memory
- System Console (3215 or equivalent)
- Two 3330/3340/3350 Drives
- Four 3420 Tape Drives
- One 3705 Communications Controller

See Figure 10 for a configuration of the minimum system. The shaded areas represent backup equipment.

Hardware Supported

- CPU's

  IBM System/370
  Model 135, 135-3, 138
  Model 145, 145-3, 148
  Model 158, 158-3
  Model 168, 168-3
  Model 195 Processor
  Model 3031, 3032, 3033 Processors
  Model 4331, 4341 Processors

- Consoles

IBM	1052-7	Printer-Keyboard
	2150	Console (with 1052-7)
	3210-1	Console Printer-Keyboard
	3215	Console Printer-Keyboard
	7412	Console (with 3215)
	3277-2	Display Station (with 3286 Printer)

- Channels

IBM	2860-1,2,3	Selector Channel
	2870-1	Multiplexer Channel
	2880-1,2	Block Multiplexer Channel

- DASD

IBM	2305-2	Fixed Head Storage
	2314-A1,B1	Direct Access Storage Facility (with Airline Buffer RPQ)
	2319	Disk Storage

	2835-2	Storage Control
	3330-1	Disk Storage
	3333-1	Disk Storage
	3340	Direct Access Storage Facility
	3350	Direct Access Storage Facility (Native Mode)
	3830-1,2	Storage Control
	Integrated File Adapter for Model 135, 138	
	Integrated Storage Control for Models 145, 148, 158, and 168	

- Tape

IBM	240X	Tape
	2803	Tape Control
	3420	Magnetic Tape Unit
	3803	Tape Control

- Unit Record

IBM	3211	Printer
	3505	Card Reader
	3525	Card Punch

- Transmission Control Unit

IBM	2703	Transmission Control
	2969	Programmable Terminal Interchange
	3705I,II	Communications Controller (Locally attached)

- Terminal Interchange/Control Units

IBM	1971	Terminal Control Unit
	2946-4	Terminal Control Subsystem
	2948	Display Terminal Interface
	S/7 (RMX/7)	Remote Multiplexer

IBM	3271	Control Unit Models 11, 12
	3272	Local Control Unit Models 1, 2
	3274	Control Unit Models 1B, 1C (SDLC/SNA)
	3276	Control Unit Models 11, 12
	3601	Finance Communication Controller Models 1, 2A, 2B, 3A, 3B
	3602	Finance Communication Controller Models 1A, 1B
	3603	Terminal Attachment Unit
	7411	Terminal Control Unit
	7441	Terminal Control Unit

- Terminals

IBM	1977-1	Terminal Unit
	1980-9	I/O Typewriter
	1980-21/24	Printer
	2740-2	Communication Terminal
	2915-3	Display Terminal

3275	Display Station
3277	Display Station
3278	Display Station Models 1-4
3284, 86, 88	Printers
3287, 89	Printers Models 1, 2
3604	Keyboard Display Models 1-6
3606	Financial Services Terminal Models 1, 2
3608	Printing Financial Services Terminal Models 1, 2
3610	Document Printer Models 1-4
3611	Passbook Printer Models 1, 2
3612	Passbook and Document Printer Models 1, 2, 3
3614, 24	Consumer Transaction Facility Models 1, 2, 11, 12
3618	Administrative Line Printer Model 1
3767	Communication Terminal (2740 emulation)
4505-21	Video Display

Figure 11 shows a typical Model 168 configuration. The shaded areas represent backup equipment.

Features Supported

- Communications

  Airline Line Control (ALC)
  Synchronous Data Link Control (SDLC)
  Low-Speed Controlled Telegraph (LSCT)
  Low-Speed Free-Running Telegram (LSFR)
  ATA/IATA Medium-Speed Link Control (SLC)
  ARINC Asynchronous Link Control (ARINC)
  Binary Synchronous Line Control (BSC)

- Cohabitation

  ACP/TPF with OS/VS1
  ACP/TPF with OS/VS2
  ACP/TPF with ACP/TPF
  VMA feature

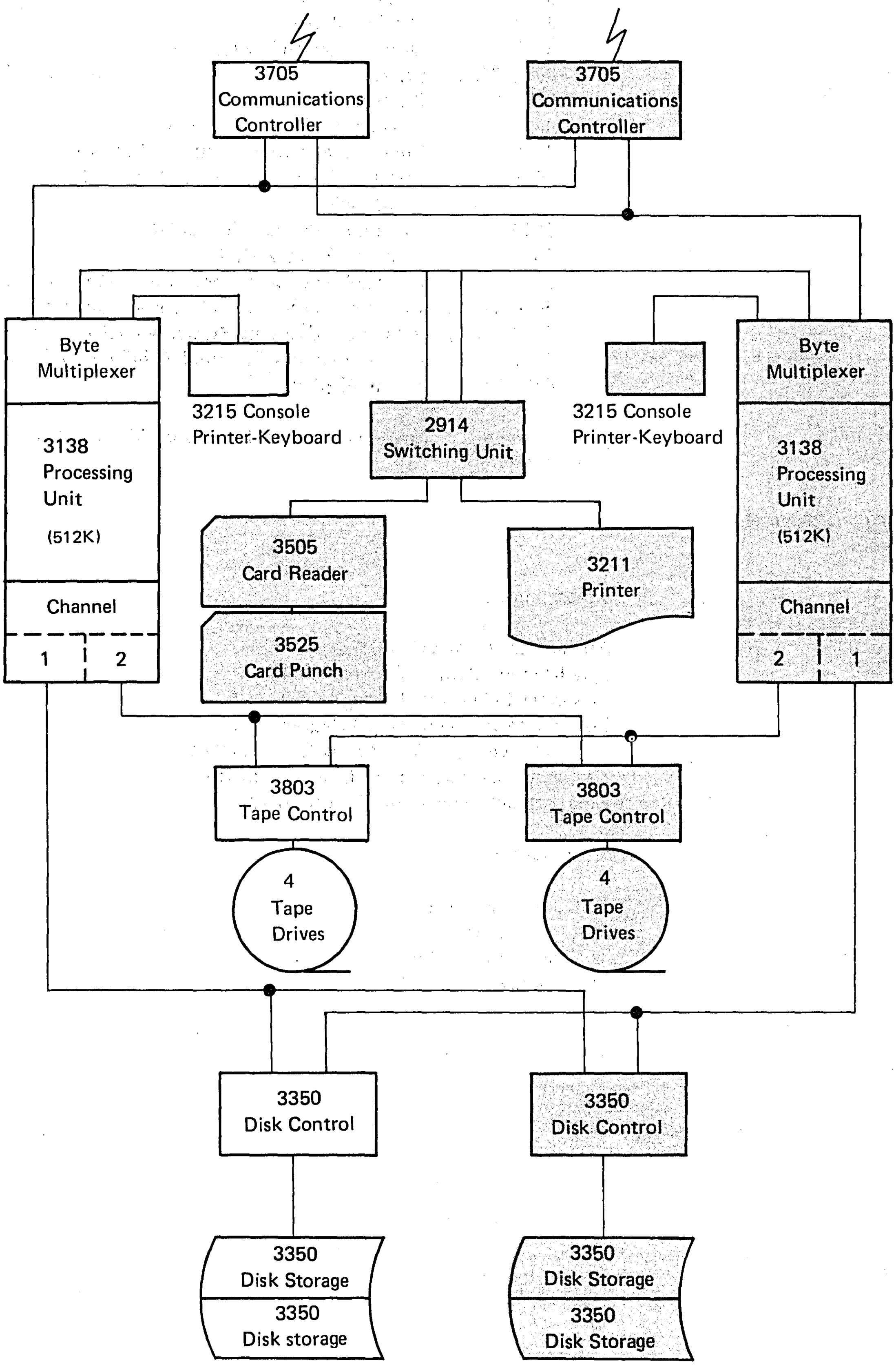


Figure 10. ACP/TPF entry configuration (System/370 Model 135)

Note: Staples can cause problems with automated mail sorting equipment.
Please use pressure sensitive or other gummed tape to seal this form.

Airline Control Program/Transaction Processing Facility (ACP/TPF)

General Information Manual

GH20-2140-1

READER'S COMMENT FORM

This form may be used to communicate your views about this publication. They will be sent to the author's department for whatever review and action, if any, is deemed appropriate. Comments may be written in your own language; use of English is not required.

IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation whatever. You may, of course, continue to use the information you supply.

Note: *Copies of IBM publications are not stocked at the location to which this form is addressed. Please direct any requests for copies of publications, or for assistance in using your IBM system, to your IBM representative or to the IBM branch office serving your locality.*

Possible topics for comment are:

Clarity Accuracy Completeness Organization Coding Retrieval Legibility

If you wish a reply, give your name and mailing address:

____

____

____

What is your occupation? ______________________________

Number of latest Newsletter associated with this publication: ____________________

Thank you for your cooperation. No postage stamp necessary if mailed in the U.S.A. (Elsewhere, an IBM office or representative will be happy to forward your comments or you may mail directly to the address in the Edition Notice on the back of the title page.)

GH20-2140-1

Reader's Comment Form

Fold and tape | Please Do Not Staple | Fold and tape

NO POSTAGE NECESSARY IF MAILED IN THE UNITED STATES

BUSINESS REPLY MAIL

FIRST CLASS PERMIT NO. 40 ARMONK, N.Y.

POSTAGE WILL BE PAID BY ADDRESSEE:

International Business Machines Corporation
Department 824
1133 Westchester Avenue
White Plains, New York 10604

Fold and tape | Please Do Not Staple | Fold and tape

IBM®

International Business Machines Corporation
Data Processing Division
1133 Westchester Avenue, White Plains, N.Y. 10604

IBM World Trade Americas/Far East Corporation
Town of Mount Pleasant, Route 9, North Tarrytown, N.Y., U.S.A. 10591

IBM World Trade Europe/Middle East/Africa Corporation
360 Hamilton Avenue, White Plains, N.Y., U.S.A. 10601

Cut or Fold Along Line

GH20-2140-1

Airline Control Program/Transaction Processing Facility (ACP/TPF) General Information Manual Printed in U.S.A. GH20-2140-1

IBM®

International Business Machines Corporation
Data Processing Division
1133 Westchester Avenue, White Plains, N.Y. 10604

IBM World Trade Americas/Far East Corporation
Town of Mount Pleasant, Route 9, North Tarrytown, N.Y., U.S.A. 10591

IBM World Trade Europe/Middle East/Africa Corporation
360 Hamilton Avenue, White Plains, N.Y., U.S.A. 10601