SOFTWARE SYSTEM INTRODUCTION
SOFTWARE DESCRIPTION
2-WIRE NO. 1 AND 1A "ESS*" SWITCHES

*Trademark

CONTENTS PAGE

CONTENTS PAGE

CONTENTS PAGE

CONTENTS PAGE

CONTENTS PAGE

CONTENTS PAGE

1. GENERAL

INTRODUCTION

1.01 The No. 1/1A ESS switch is an automatic common control type switching system operating under control of a processor directed by a stored program. This stored program is organized as a generic program plus a data base which uniquely associates the generic with a particular office. A generic program is a general purpose switching system control program for ESS switching equipment and is generic in the sense that the same basic program is used for all No. 1/1A ESS switching equipment installations. A data base, consisting of parameter and translation data, is created to uniquely define a particular ESS switch installation to its generic program. The generic program consists of all of the software and data necessary for routine and maintenance operations. For purposes of discussion, the generic program may be broken into many functionally dependent parts. This document provides an introduction to the No. 1/1A, generic program at the functional part level. Any information that is peculiar to 1A ESS switch is noted as such. Information that is peculiar to No. 1 ESS switch is not given.

1.02 This section is being reissued to include coverage of:

- Circuit Switch Digital Capability (CSDC)
- Supervision Modernization
- Attached Processor System (APS)
- Memory Expansion
- Peripheral Unit Controller
- Peripheral Unit Controller/Data Link
- Station Message Detail Recording (SMDR) and Expanded Message Detail Recording (XMDR)
- Automatic Call Distribution (ACD) Electronic Switching Systems Management Information System (AEMIS) Data Base (MSDU)
- Centrex Station Rearrangements (CSR).

Change arrows are used to indicate these significant changes.

1.03 This section is based on the 1E7 (No. 1 ESS switch) and 1AE7 (No. 1A ESS switch) versions of the generic program.

SOFTWARE DOCUMENTATION

1.04 The term software, as used in this document, refers to the series of programmed instructions and associated data used to direct the operation of a No. 1/1A central control. Supportive information such as maps, listings, and program documentation is included in this definition. Refer to Part 7 of this document for more detailed information.

PURPOSE OF THE SOFTWARE

1.05 The software, ie, generic program package, is the set of stored programs which controls the operations of the ESS switch. The ESS switch software directs all hardware activities for normal operations, including maintenance and interrupt environments.

SCOPE OF SECTION

1.06 This document provides a system level introduction to the No. 1/1A generic programs. Additional information describing software support information is provided. Some of the programs or pidents referenced may not be active for particular installations. Refer to the program listings for specific applications.

1.07 Part 9 of this document provides a defined list of the abbreviations and acronyms used in this section.

1.08 Section 231-045-000, titled "Introduction to Software Description," provides cross-reference lists of all programs covered in the software description series, which are:

- 254-280-XXX
- 231-045-XXX
- 231-310-XXX.

Note: The last three digits in each series further designate the functional positions of the document within the overall No. 1/1A ESS switch software coverage.

1.09 PG6A002 titled "ESS Switch Index of Generic Program Documents for Program Store and

Call Store Generic," contains a listing of program titles and pident numbers.

2. MAJOR FUNCTIONS DESCRIBED

CONTROL STRUCTURE

2.01 The generic program can be functionally arranged into three functional divisions as shown in Fig. 1.

(a) The operational software includes the base level or normal operations necessary to accomplish call processing and other routine operations.

(b) The maintenance software is responsible for the automatic and manually initiated maintenance and diagnostic programs used for ESS switch maintenance.

(c) The interrupt control software is predominantly responsible for controlling or obtaining control of the ESS switch during abnormal processing environments and for restoring the system to normal operations.

2.02 The ESS switch software documentation consists of this and other manuals or support information which provide ESS switching equipment operation and maintenance personnel with the required user information.

3. INTRODUCTION TO PROGRAM CONTROL

PROGRAM STRUCTURE

3.01 The ESS switch programs are controlled under a structure designed primarily to handle the call processing requirements of the office but, in addition, provide for other operational and maintenance functions. Most normal operations, including the bulk of call processing and routine maintenance, are carried out by the base level portion of the generic program. Base level refers to the hardware exe-

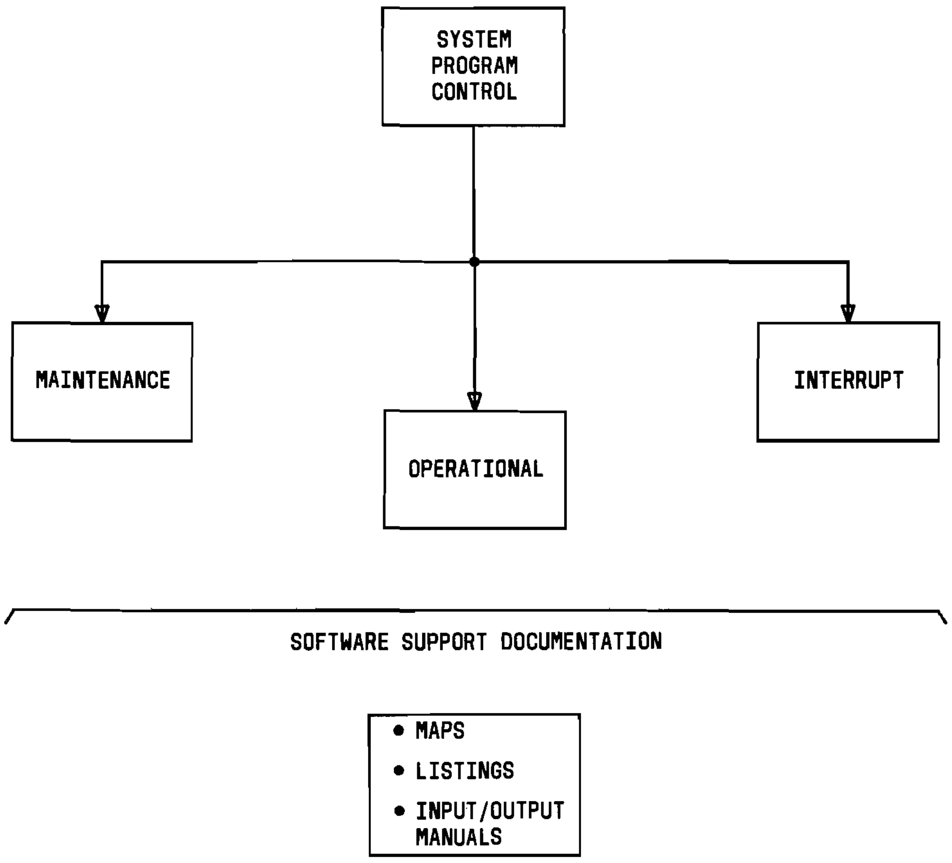


Fig. 1—ESS Switch Software and Documentation Breakdown

cution state existing when the interrupt handling mechanisms are not in control. All base level programs and routines collectively form the base level program which, although functionally divisible, operates as a group of endless loops, often referred to as the base level loop. The core of the base level program is the executive control main program.

3.02 The base level program execution sequence may be temporarily interrupted by programs which operate on interrupt level (Fig. 2). Provision is made for ten levels of interrupt priority: A through K, excluding I. An interrupt causes control to be taken from the instruction currently in control of the hardware and given to the interrupt handling programs. The highest priority level of interrupt is A-level.

CENTRAL CONTROL

3.03 The central control is the heart of the ESS switch processor system. The No. 1/1A processor, consisting of two identical central controls (central control 0 and central control 1 provide basic processing redundancy), process instructions and data stored in various memory devices. The major processing function is to administer the switching network for call processing purposes. The central control reads, decodes, and executes the coded machine instructions. Execution times vary according to instruction types and sequences.

MAIN MEMORY

3.04 Main memory is comprised of program stores and call stores. Both program store and call store are high speed random access stores operating under control of the central control. The primary function of the program store is to store the memory resident portion of the generic program. Call store serves primarily as a repository for call processing and other system related data. Program store holds a single copy of the resident generic program while call store may provide simplex (single copy) or duplex (redundant) storage.

AUXILIARY MEMORY STORAGE

3.05 In the No. 1A ESS switch, auxiliary storage is provided by file store and tape units. File store consists of a disk storage system which provides high-speed access (but slower than program store or call store) to bulk stored programs and data. The primary functions of file store are (1) to store the non-resident portion of the generic program, and (2) to provide backup copies of all program store and simplex call store information. The file store system also provides complete redundancy for itself.

3.06 Magnetic tape storage for the No. 1A ESS switch is provided by the auxiliary data system (ADS). The ADS provides a versatile, medium-speed, data storage and retrieval facility. Tape provides for further redundancy of system programs, data, etc.

BUS SYSTEMS

3.07 The four major bus systems of the processor provide 2-way communications between the central control and other processor units and the periphery. These four bus systems are:

- Call store bus
- Program store bus
- Peripheral unit bus
- Auxiliary unit bus (1A only).

Each bus system provides a write bus to send data from central control to a unit and a reply bus for central control to receive a reply from a unit.

3.08 Access to a particular unit on a particular bus is via an address. In order to address any particular unit, an address including an enable field (ie, a K-code) which specifies the unit being accessed is applied to the address bus for the particular bus system. The K-code plus the remaining address bits specify the location within the unit and the type of operation. For example, the call store address field includes:

(a) A 5-bit K-code identifying one call store

(b) An address specifying one within the store module.

Each bus system also provides for transfer of maintenance, control, and status information.

4. OPERATIONAL SOFTWARE FUNCTIONS

4.01 Despite functional subdivisions that are made for various purposes, the ESS switch pro-

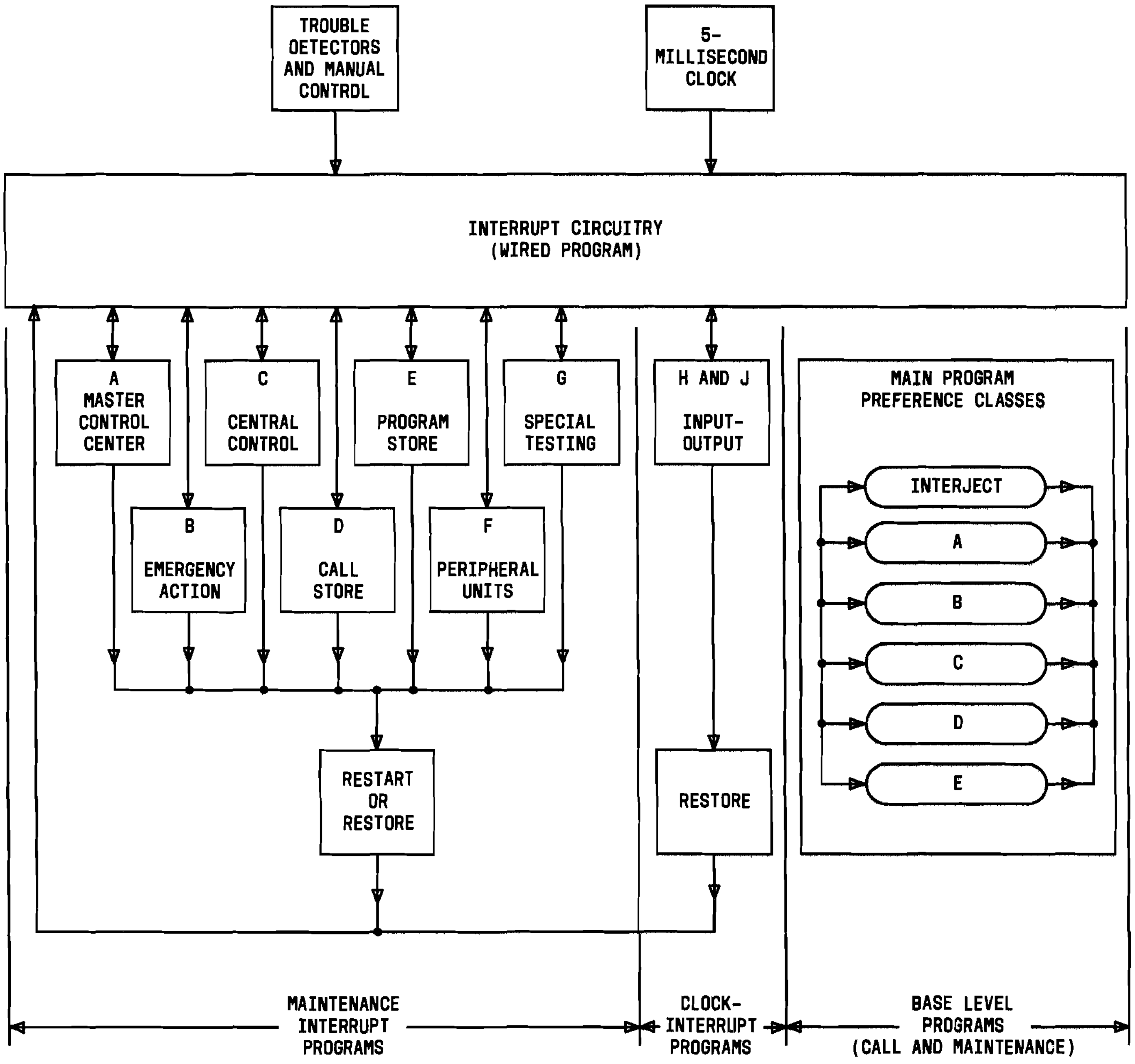


Fig. 2—Program Control Plan

grams are operationally a single program made up of endless loops. The operational software control structure controls the orderly scheduling and execution of programs used in call processing, input/output management, and administrative control, via the operational software listed in Table A. Each of these functional areas is described in subsequent paragraphs.

SOFTWARE CONTROL PROGRAMS

A. General

4.02 Programs associated with operational software at the major control level are as follows:

(a) Executive Control Input/Output Program

TABLE A

NO. 1/1A ESS SWITCH OPERATIONAL SOFTWARE

Operational Software Control
Call Processing
Scanning
Outpulsing
Peripheral Control
Operator Functions
Charging
Toll/Tandem Switching
Translations
Recent Changes
Queue and General Purpose
Centrex—Data Link
Special Services
Measurements
Network Management
File Store Administration*
Processor Input/Output
Auxiliary Data System*

* No. 1A only

(ECIO): The ECIO administers and schedules the execution of J- and H-level input/output programs and sets up the control structure for running base level jobs.

(b) Executive Control Main Program (ECMP): The ECMP administers and schedules the execution of all base level (L-level) programs.

(c) Automatic Overload Control Program (AOVD): The AOVD identifies overload conditions as they occur and initiates control strategies to reduce the demands on system resources.

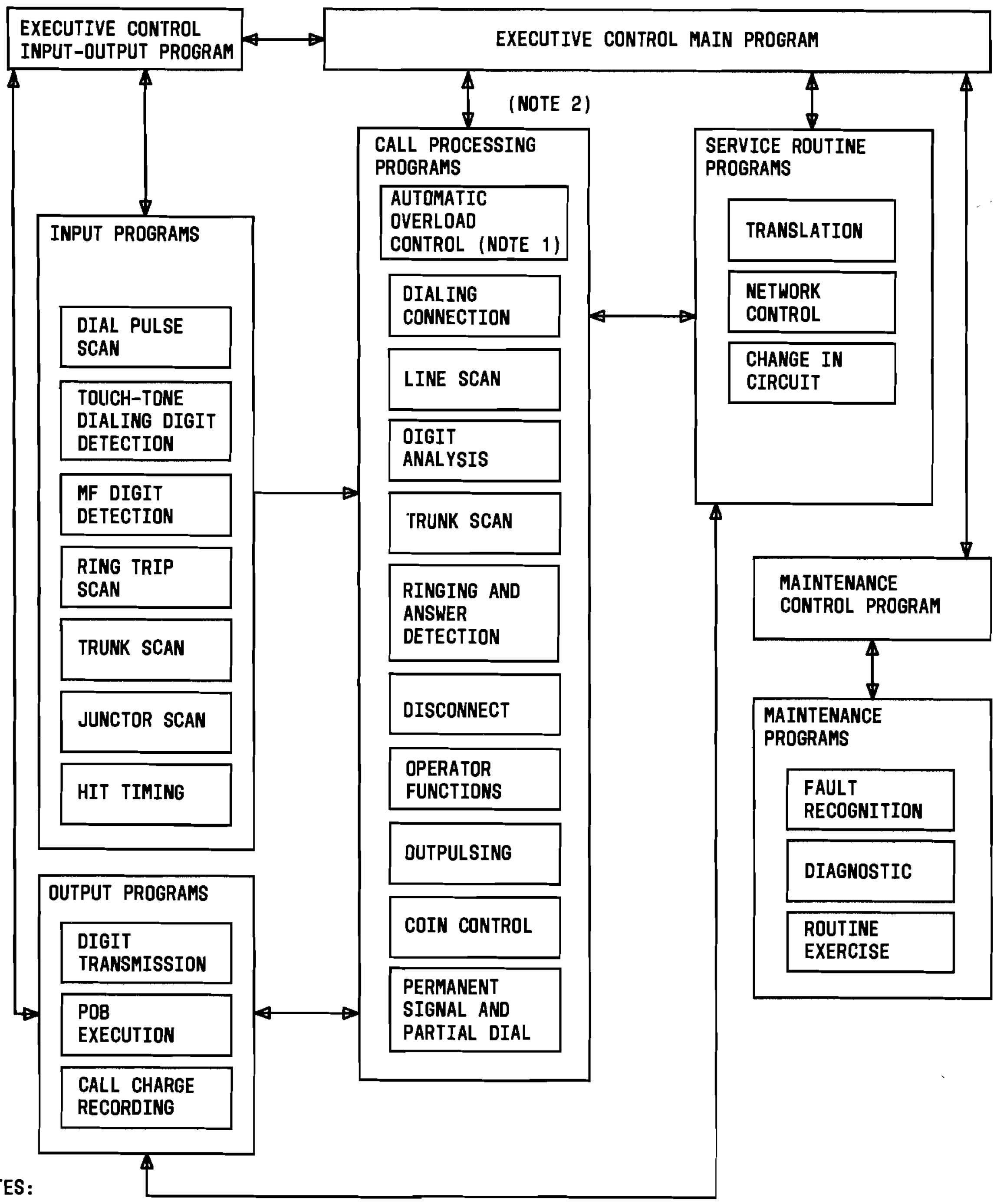
The basic operational control structure interface is shown in Fig. 3.

B. Input/Output Program Control

4.03 Every 5 milliseconds a system clock activates the J-level interrupt which gives control to the input/output programs via timetables administered by the executive control input/output main program.

4.04 All input/output programs are classified into high priority and low priority tasks according to the frequency and urgency with which these tasks must be performed. Low priority tasks can be delayed for a few milliseconds without an adverse effect on the operation of the system. In the event that the coincidence of input work under a peak traffic load causes the system to take more than 5 milliseconds to complete high and low priority tasks on J level, the H-level interrupt will occur and the low priority work will be interrupted. The high priority tasks will again be performed before returning to the low priority task that was interrupted. When the low priority tasks have been completed, return is made to the interrupted base level program. A general flow diagram of J- and H-level processing is shown in Fig. 4.

4.05 In order to perform all tasks promptly, the individual task must not take too long. Thus it is necessary to limit the amount of processing performed by the interrupt programs. The input programs are confined to scanning for and recognizing input signals and storing the input information in a call store hopper. Each hopper is inspected by the base level programs. When data is present in the hoppers, appropriate base level programs start or continue the processing of the call. Likewise, call store buffers are provided for use by the base level programs in storing output information. At an appropriate time, the call store buffers are unloaded by output programs which deliver the information to the peripheral equipment. The peripheral order buffers are used to store address and control information for peripheral equipment, such as network controllers and signal distributors. These buffers provide the means for communication between the scheduled input/output programs and the base level call processing programs.



NOTES:

1. THE AUTOMATIC OVERLOAD CONTROL PROGRAM CONTAINS BOTH CALL PROCESSING AND OVERLOAD CONTROL ROUTINES.
2. THE INTERCONNECTION LINES INDICATE THE GENERAL FLOW OF INFORMATION AMONG PROGRAMS. THESE LINES DO NOT DEPICT SPECIFIC PROGRAM RELATIONSHIPS OTHER THAN THE RELATIONSHIP TO THE EXECUTIVE CONTROL MAIN PROGRAM AND TO THE TYPES OF PROGRAMS. THE EXECUTIVE CONTROL MAIN PROGRAM SCHEDULES THE WORK OF ALL CALL PROCESSING PROGRAMS.

➧Fig. 3—Operational Control Structure—Functional Grouping➧

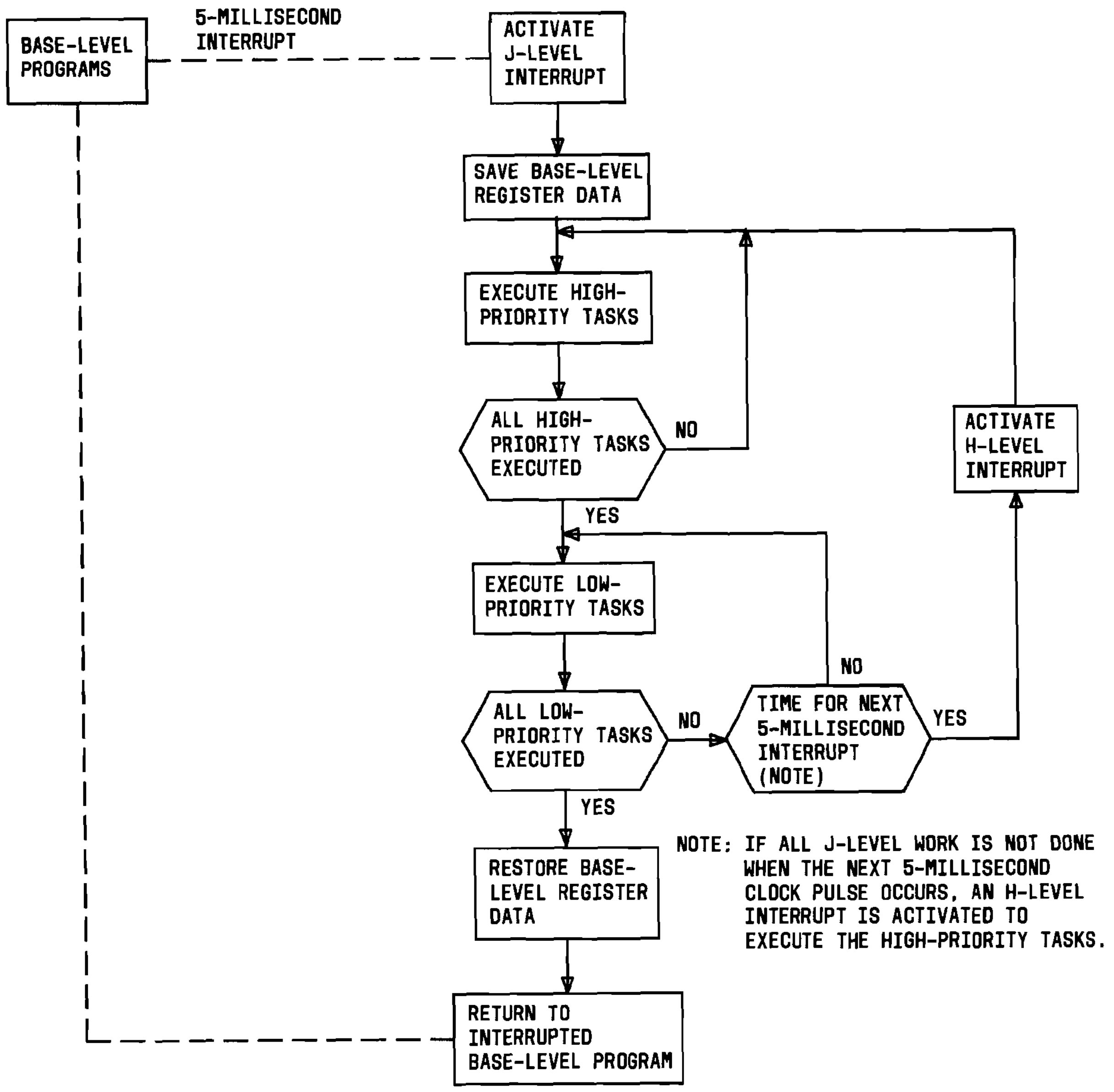

BASE-LEVEL PROGRAMS
5-MILLISECOND INTERRUPT
ACTIVATE J-LEVEL INTERRUPT
SAVE BASE-LEVEL REGISTER DATA
EXECUTE HIGH-PRIORITY TASKS
ALL HIGH-PRIORITY TASKS EXECUTED
NO
YES
ACTIVATE H-LEVEL INTERRUPT
EXECUTE LOW-PRIORITY TASKS
ALL LOW-PRIORITY TASKS EXECUTED
NO
TIME FOR NEXT 5-MILLISECOND INTERRUPT (NOTE)
NO
YES
YES
RESTORE BASE-LEVEL REGISTER DATA
RETURN TO INTERRUPTED BASE-LEVEL PROGRAM
NOTE: IF ALL J-LEVEL WORK IS NOT DONE WHEN THE NEXT 5-MILLISECOND CLOCK PULSE OCCURS, AN H-LEVEL INTERRUPT IS ACTIVATED TO EXECUTE THE HIGH-PRIORITY TASKS.

Fig. 4—J- and H-Level Interrupt Processing—Flow Diagram

C. Base Level Program Control

Frequency Class and Interject Administration

4.06 The bulk of the programs, both call processing and maintenance, are executed on the base level while no interrupts are in effect. All base level work can be deferred to some extent, but the amount of delay each program can tolerate varies widely. It is for this reason that a priority system implemented by a control program is used within the base level.

4.07 All base level programs are controlled by a single program called the executive control main program, ECMP. The main program performs its control function on the base level with the use of six priority classes of programs. The highest priority class is interject. The other five classes are A, B, C, D, and E, in descending order of frequency of examination. Among other tasks included in the base level scheduling universe (external to frequency classes A through E) are supervisory trunk and line scanning and routine maintenance.

4.08 Within each frequency class there is a fixed sequence of major program units called task dispensers. The majority of these are for call processing and administration. In general, they dispense program control to one or more task programs a consecutive number of times, depending on the number of tasks that the task dispenser program finds waiting.

4.09 Although interject work is not regularly scheduled in the class sequence, it has priority over the work in frequency classes A through E. After execution of each task program within a class, a check is made for the existence of an interject request. If interject work is required, it is performed before returning to the task dispenser presently in control.

System Overload Control Administration

4.10 System overload occurs when offered traffic produces excessive demands on any of the available system resources (hardware, software, and real time). The objective of automatic overload control is to identify overload conditions as they occur and then to initiate appropriate control strategies.

4.11 Overload conditions are generally classified as hardware, software, or real time, according to the source of the problem. Hardware overloads can occur as a result of all network paths being blocked or when the total demand exceeds the amount of available service circuits, outgoing trunk circuits, or 2-way trunk circuits. Software overloads occur when the demand for memory resources exceeds the supply, for example, when no register of a particular type is available or when a hopper overflows. Real-time overloads occur when the system work load requires more time than is generally acceptable to maintain efficient system operations. The basic strategies employed for hardware and software overloads are (1) try another way, (2) queue, (3) try again later, and (4) do not serve the call. For real-time overloads, the strategies are to delay and/or eliminate work.

4.12 During an extreme situation such as an emergency or a disaster, traffic may increase to the point that the automatic overload controls cannot alleviate the overload condition; consequently, service can be degraded. ▶Service during these periods is improved with the use of a last in first out (LIFO) buffer line service request algorithm. This algorithm assures that the most recent originations will receive dial tone first. This reduces the probability of partial dials and/or misdirected calls which would place further resource constraints on the system. The LIFO is augmented by an automatic Dynamic Service Protection mechanism which provides some degree of priority treatment during overload to essential lines.◀

CALL PROCESSING SOFTWARE

A. General

4.13 The ESS switch interconnects telephone customers by centralizing the decision-making and memory required to process telephone calls in an electronic data processor. As a result, many aspects of call processing have been greatly simplified.

4.14 The purpose of call processing is to interconnect customer telephone lines. Calls between customers in the same central office (intraoffice) are handled a little differently than calls between customers in separate central offices (interoffice). However, there are three categories of software which become involved with both types of calls. These are: (1) programs which detect changes in the periphery and constitute inputs to the ESS switch and those which produce changes in the periphery and constitute system outputs (input/output programs); (2) call control programs which have only call related purposes and whose function is to advance a call to completion; (3) programs called service routines which perform frequently used functions. These can be used by call processing programs as well as other programs. Figure 5 shows the call processing interfaces.

B. Input/Output Programs

Input Programs

4.15 The programs which detect system input are designed to be relatively simple and highly efficient programs. They report changes or events to call control programs which analyze the report and perform any required actions. There is a large number of inputs (scan points) to be interrogated regularly, but the number of changes detected at any one time is expected to be quite small.

4.16 The program which detects line service requests interrogates all line scan points in the office approximately three times a second. The line scanners are arranged so that 16 line scan points are

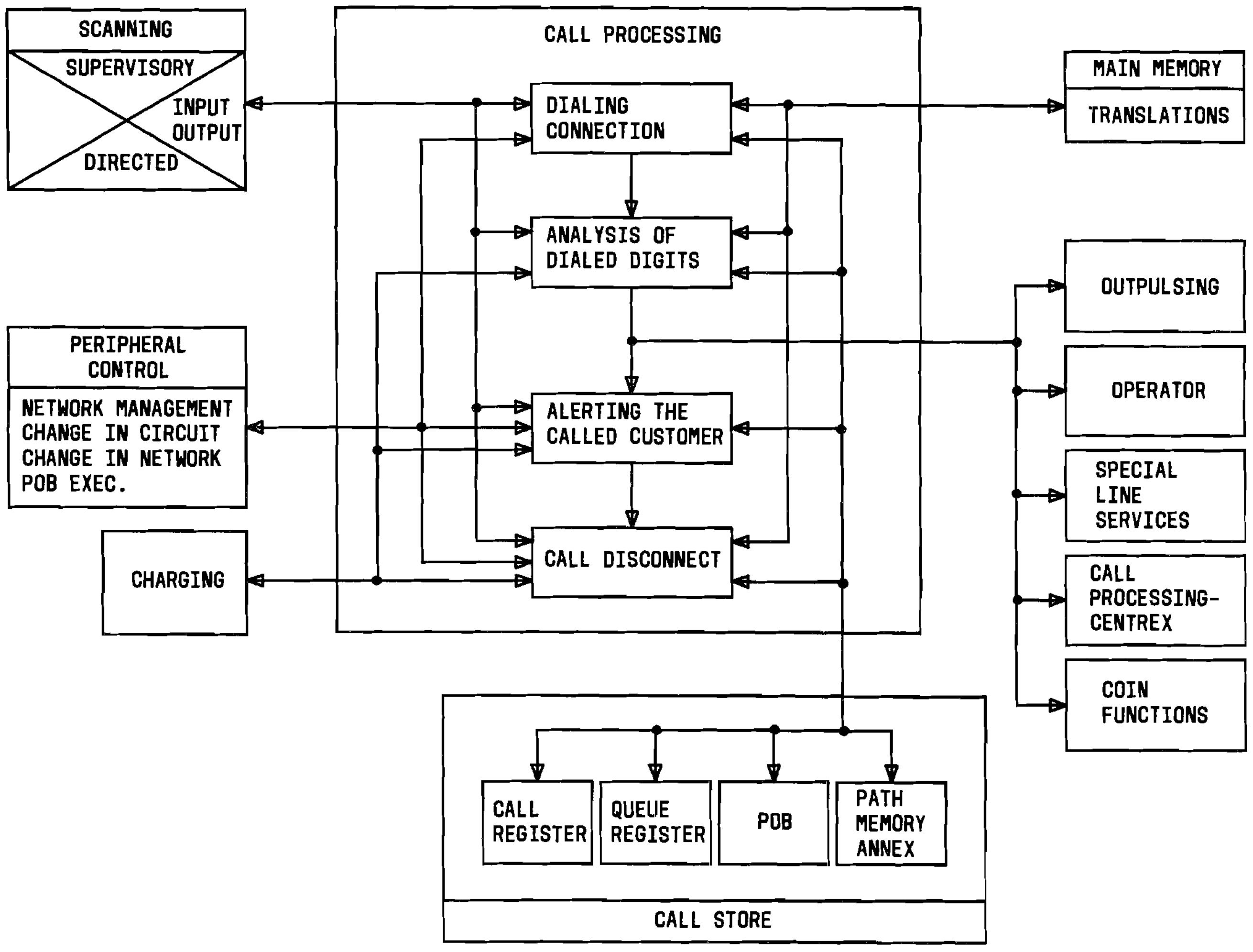


Fig. 5—Call Processing Interfaces

read simultaneously. The supervisory scan program reports the origination to a call control program ▶via a hopper entry in the line service request hopper◀ and continues its round of line scanner interrogation.

4.17 Another input program detects dialed digits and reports them to a call control program which will determine whether any action should be taken and if so what actions. The program also performs two other auxiliary functions for which reports are made. The first function is to report when the first digit is dialed so the call control program will know to remove dial tone. The second function is to perform permanent signal and partial dial timing. If no digits are received for an interval of time, the program reports this to the call control program, which will handle the call from that point on.

4.18 A third input program scans the relays controlling the ringing circuits. When the called customer answers, the input program reports to a call control program which removes the ringing connection and establishes a talking connection.

4.19 Another scanning program looks for changes from off-hook to on-hook. A change of state from off-hook to on-hook may be a momentary hit on the line or an inadvertent switchhook jiggle. Also, the customer may be flashing to initiate a special service request. Therefore, the scanning program reports the change to a hit-timing program which times the length of the on-hook signal to discriminate between hits, flashes, and true disconnects. The results are reported to the call control programs, which decide the appropriate action to be taken.

4.20 The trunk scan program detects a number of signals in incoming and outgoing trunks. The program can deduce some information from the changes it is designed to detect. For example, a change from off-hook to on-hook indicates the start of a disconnect or a flash, and is reported to the hit-timing program. A change from on-hook to off-hook on an incoming trunk constitutes a request for service; but on an outgoing trunk, it could indicate that the call was answered at a distant office. ➧The scanning programs do not know if the on-hook to off-hook is an answer or request for service. The program loads the information of the events into the trunk seizure and answer hopper (TSAH). Then the base level determines the type of report.➧

Output Programs

4.21 When the processor needs to output data to initiate action in the periphery, it encounters an entirely different time scale. The generic program processes parts of a call at a very high rate of speed. To operate relays and other hardware associated with completing calls, a relatively large amount of time is required. The processing of other call elements cannot be delayed to wait for these actions.

4.22 ➧The call processing program determines which periphery needs to be operated and loads peripheral control orders into a peripheral order buffer

- Close or open a relay in a trunk circuit
- Close a network switch
- Scan a scan point in a trunk circuit.

After loading all orders into the peripheral order buffer, the call processing program suspends further processing of that call and passes the peripheral order buffer to an output program.

4.23 The output programs are executed once every 25 milliseconds. Each execution is called a cycle. On every cycle, the output program visits a peripheral order buffer. One order is unloaded from the peripheral order buffer and transmitted to a peripheral circuit. While this peripheral circuit is slowly performing the transmitted order, the output program does not wait but proceeds to call another peripheral order buffer. The output program unloads one order from that peripheral order buffer and transmits that order to another peripheral circuit. The output stops executing the current cycle when it visits all peripheral order buffers passed to it by the call processing program and returns control to the main executive program. Then the main executive program processes other calls or performs maintenance function. On the next cycle, the output program recalls each peripheral order buffer, unloads the next order from the visited peripheral order buffer, and transmits the order to the appropriate periphery circuit. The output program repeats this sequence of events during each cycle. When the output program unloads all orders from a given peripheral order buffer, the output program passes that peripheral order buffer back to the call processing program to resume processing the call until the point of interruption.

> ***Note:*** Interweaving output functions with call processing functions in this manner, allows the central control to process calls and perform maintenance functions. This allows periphery to operate at a slow speed.➧

C. Call Control Programs

4.24 Each call control program performs a specific function, usually related to a stage in the progress on a call. This separation of responsibilities permits each program to be of manageable size and perform a defined function well. It also makes the addition of new features relatively easy.

4.25 On a normal intraoffice call, many call control programs are brought into play and are responsible for handling the call through various stages. These are:

- Dialing connection
- Digit analysis
- Ringing and answer detection
- Call disconnect.

Dialing Connection Programs

4.26 When a customer requests service, a report is made by the line scan program. To serve the request, there are several things that the dialing connection programs must do. They must find a block of temporary memory in which to store data regarding the calling line and the number that the customer will dial. Also, they must acquire some information about the calling line such as whether it is dial pulse (DP) or TOUCH-TONE® service or if it has any special features.

4.27 When this information has been obtained, the programs select an idle customer digit receiver which is compatible with the telephone being used. A network connection is made from the calling line to the selected digit receiver. After the connection has been established and the line cutoff contacts opened, the digit receiver applies a power-cross detection test to the line and reads a scan point to determine the results of the test. After all the necessary tests have been performed and passed, the programs cause a relay in the digit receiver to operate and connect a supervisory relay and dial tone to the calling line.

4.28 A similar procedure is followed to establish a dialing connection for trunks. When an incoming trunk is seized, the dialing connection for the trunks program is given the trunk network number of the trunk and additional information, such as supervisory information and what type of digit receiver is needed.

4.29 An idle incoming register is seized and a path is established between the trunk and an idle receiver of the appropriate type (MF, DP, revertive or TOUCH-TONE service). Supervision is then supplied and a start pulsing signal is sent if needed. A receiver junior register is seized for digit scanning. Received digits are loaded into a digit hopper along with abandon reports. Control is then passed to the digit analysis programs.

4.30 It should be noted that the dialing connection programs merely request the actions; they do not actually perform these actions. Instead, the services of the network control programs are called upon to find an idle path from the line to the digit receiver. The network control programs load the network controller addresses and instructions in a peripheral order buffer. The dialing connection programs then call on the circuit control programs to load the desired relay and scan actions in the peripheral order buffer. After the buffer loading is complete, and the buffer is activated, the peripheral order buffer execution programs remove the instructions from the buffer one at a time until it is emptied. The peripheral order buffer execution programs then report back to the dialing connection programs that the execution was completed successfully. If trouble develops during execution, the rest of the actions will be skipped and a failure report returned to the dialing connection programs.

Digit Analysis Programs

4.31 These programs are responsible for recording and interpreting the customer's digits as they are dialed. As the digits are received, they are counted and stored. ➧At predetermined digit counts (usually after the first, third and the seventh digit), all digits dialed up to that point are analyzed by translation programs to determine their meaning. The translations program returns:

(a) A call type

(b) Charging condition

(c) How many more digits are expected

(d) When the next digit analysis is needed.➧

If the digits dialed so far represent a prefix or special code requiring special actions, translation routines will return the information necessary to pass control to the proper programs.

4.32 Call processing actions vary solely as a function of the dialed digits as interpreted by the translation programs. For example, the translation programs may return the following call type information:

- Service code dialed
- Invalid code
- Interoffice code
- Intraoffice code
- Charge or free call.

4.33 If the call type indicates a valid service, program control is transferred to the operator functions subsystem. If the digits are invalid, the call is routed to an intercept operator. When the digits dialed indicate an area code, the translations program further determines if the call is to a trunk code or zone code. If a trunk code is indicated, several digits are absorbed and the program continues to collect further digits until all the expected digits are received. If a zone code is indicated, the program continues to collect digits until all expected digits are received. In either case, the call is now termed an interoffice call. The translation program also indicates a charge in both cases.

4.34 When the digits dialed indicate a trunk code, the translation programs will determine if the call is to another local central office or to a customer served by the same office. If the call is to another local central office, the program continues to collect digits until all expected digits are received.

4.35 If the call is to a customer served by the same central office, the call is termed an intraoffice call, and the program continues to collect digits until all expected digits are received. When all digits are received, this program then shuts off the customer digit receiver. The translation subsystem is again used to convert the dialed directory number to a called line equipment number and terminating class information. This information consists of the following:

- Busy or idle line
- Invalid number
- Special treatment line
- Temporary transfer activated
- Trunk group found.

4.36 If the called number is invalid, program control is transferred to an intercept operator, tone circuit, announcement circuit, or automatic intercept system depending upon local option. Otherwise, digit analysis checks to see if the called line was found busy or idle. If the line is busy and does not have special treatment, a peripheral order buffer is seized, loaded, and executed (by using the peripheral control subsystem) for establishment of a network connection between the calling line and a busy tone circuit. If the line is busy and has special treatment (eg, call wait), program control is transferred to the special line services subsystem. If the line is found to be idle, the digit analysis program will then seize and initialize a ringing register, if available. If all ringing registers are busy, queuing for a ringing register is done. When a ringing register becomes available, program control will be transferred to the ringing and answer detection programs.

Ringing and Answer Detection Programs

4.37 The ringing and answer detection programs, as their name implies, control all system actions on intraoffice or incoming calls from completion of dialing until answer. Their basic job is to connect ringing to the called line, connect audible ringing tone to the calling line or trunk, establish a talking connection if the call is answered, and remove all connections if the call is abandoned before answer.

4.38 The network locations of lines and the type of ringing to be applied to the called line are needed to set up the ringing connection. The information regarding the calling and called lines (intraoffice calls) is passed to the ringing and answer detection programs by the digit analysis programs. With this information, the ringing and answer detection programs call upon the services of the network control programs asking these programs to:

(1) Find idle circuits connected to the active phases of ringing and audible ringing

(2) Find and idle path from the calling line to the selected audible ringing circuit

(3) Find an idle path from the called line to the ringing circuit

(4) Reserve a path from the calling line to the called line

(5) Load the instructions for making these connections into a peripheral order buffer.

4.39 The ringing and answer detection programs then call upon the circuit control programs to load further instructions in the peripheral order buffer among which are:

- Power Cross test
- A pretrip test
- Continuity.

The circuit control programs also load instructions to operate circuits which will apply ringing voltage to the called line and audible ringing tone to the calling line.

4.40 An answer is detected by the ringing trip scan program and is reported to the ringing and answer detection programs. Upon receipt of the report, ringing and answer detection programs call upon network control and circuit control programs to release the ringing and audible ringing circuits, idle the two network paths, and to set up ➧the previous reserved network path between two lines. A junctor circuit is used to connect the two lines together.➧

Disconnect Programs

4.41 Unless a special service call is in progress, no other call control programs are called into play until one of the customers hangs up. At this time, a scanning program reports this event to the disconnect programs. Hit timing and optional flash timing will already have been performed by the scanning program.

4.42 The functions of the disconnect programs are:

- To provide calling line control of the call
- To signal disconnect to a distant office over an incoming or outgoing trunk
- To remove the talking connection at disconnect
- To restore to idle any lines or trunks involved in the call.

4.43 To determine the treatment for disconnecting a call, the programs must find out some information about the line reporting a disconnect. This information includes which party has disconnected, whether the line has any special services, and whether there is an incoming or outgoing trunk involved. Different actions are called for, depending on this information.

4.44 ➧For an intraoffice call, if the disconnect is from the calling line, the calling line is idled and the called line is placed on a 10-second false origination protection timing. When either the timing expires or the called line disconnects, the called line is idled. The false origination protection timing allows the called customer to determine that the call has ended and disconnects. If the called line disconnects first, the connection between the two lines is split and the call is placed on a 10-second timed release disconnect timing. When either the timing expires or the calling line disconnects, both lines are idled. If the called line returns off-hook within 10 seconds, the connection between the two lines is unsplit and the call is allowed to resume.

4.45 Disconnect for an interoffice call is handled in a similar manner with one additional step when the called line disconnects first. Before placing the call on the 10-second timed release disconnect timing, an on-hook signal is transmitted over the incoming trunk to the calling office notifying that office that the called line went on-hook. Likewise, when the called line returns off-hook within 10 seconds, an off-hook is transmitted over the incoming trunk to the calling office. The calling office needs to know the status of the called line because the calling office also performs a timed release disconnect timing simultaneously with the called office and disconnects the calling line. This occurs when the calling office does not receive an off-hook from the called office within 10 seconds.➧

D. Service Routines

4.46 Most call control programs use a number of service routines while controlling the progress of a call. Examples of service routine usage are: to request a change in a network configuration, to request the operation or release of a relay in a trunk or service circuit, and to obtain translation information. These service routines are used not only by the call control programs but also by the maintenance and diagnostic programs.

Network Control Programs

4.47 The primary function of the network control program is to hunt for idle network paths, to administer the network map and path memory, and to load instructions in peripheral order buffers which are used to close network paths. In the process of performing these functions, the network control program is provided with the ability to find an idle trunk in a group, to make a second trial (or third trial for line to trunk paths) if all the paths to the first selected trunk are busy, and to consult the translation program to find an alternate route if all trunks are found busy in the first-choice route. Since a record of the busy or idle condition of all links in the switching network is kept in temporary memory (network link map), the network control program can reserve a path from one terminal to another terminal for expected use at a later stage of a call. Similarly, the information regarding an established or reserved connection in the switching network is kept in temporary memory associated with network terminals (path memory). The network control program records pertinent information about a path at the time a connection is reserved, established, or removed.

Circuit Control Programs

4.48 When a call control program determines that a change of state in a trunk or service circuit is required to make a test or cut through a talking path, the call control program calls upon the services of the change in circuits program. The client informs the change in circuits program of the type of circuit to be used and the function to be performed. Then the change in circuits program loads the peripheral order buffer with the signal distributor operations necessary to implement the change and any scanner actions needed to check that the operation was performed successfully.

> ➧*Note:* The change in circuits program does not perform these functions. It is the clients responsibility to perform these actions as necessary.➧

CENTREX CALL PROCESSING

A. General

4.49 A Centralized Telephone Communication Exchange (centrex) processes calls for a business customer group. Special switching equipment may be installed on the customer's premises or the lines may be tied directly to the ESS switch. Centrex calls are processed by four centrex programs in addition to the normal call processing programs. These centrex programs perform the following:

(a) Analyze and translate digits dialed from a centrex station.

(b) Analyze and translate digits from a distant office for a call to a centrex station.

(c) Provide disconnect on calls to and from centrex stations.

(d) Seize and release registers needed for processing centrex calls. Figure 6 shows the centrex call processing interface.

B. Centrex Digit Analysis Program

4.50 Whenever the dialing connection program determines by translation that a line requesting service is a centrex line, it passes control to the centrex digit analysis program (CXOR). The CXOR receives the dialed digits, then uses a table lookup technique to determine what type of service is being requested. The digits may be the number of another centrex station, the centrex attendant, a line outside the centrex, an invalid number, or a request for special centrex services. The CXOR performs the actions necessary to route the call or provide the requested special service.

C. Incoming Digit Analysis for Centrex

4.51 The incoming digit analysis for centrex program (CXIC) performs the same kind of functions as CXOR, but for incoming calls to centrex. It is entered from the digit analysis for trunks program and performs any special actions needed to complete centrex calls. This may include routing to a centrex station, the centrex attendant, or a recorded announcement. Additional processing may be done if the called station has special features such as call forwarding.

D. Disconnect for Centrex

4.52 The disconnect for centrex program (CXDS) assures correct disconnect of centrex calls. It works together with the normal disconnect program (DISC). Centrex features such as ADD-ON, which require flashing, are also handled by CXDS. If a flash is detected, a transfer is made to the appropriate feature routine. For normal disconnect, disconnect timing is done and the connection is disconnected.

E. Centrex Register Seize and Release

4.53 The centrex programs use several types of call registers for storage of information needed temporarily for call processing. The centrex register seize and release program (CXYH) maintains a list of idle centrex call registers. Whenever one of the other centrex routines needs a register, it requests one from CXYN. The CXYN allocates a register and removes that register from the idle list. When the register is no longer needed, CXYH is called to put it back onto the idle list. The CXYH program handles four types of registers:

- Customer Facility Group Registers

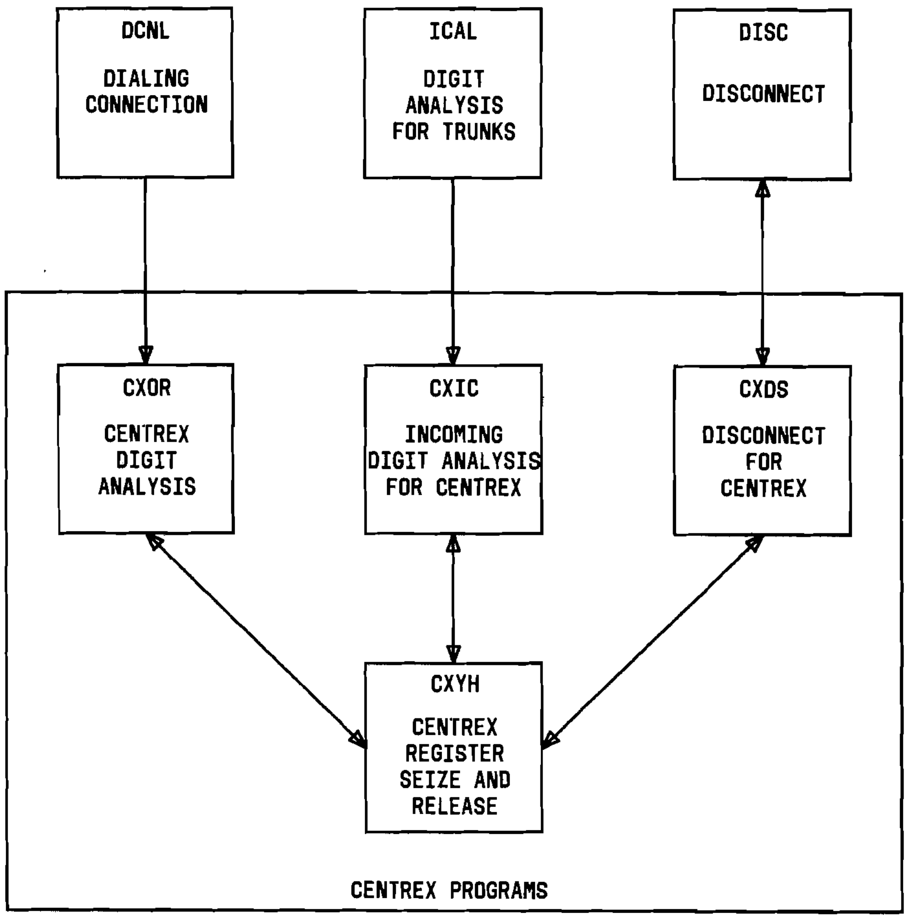


Fig. 6—Centrex Programs Interface

- Centrex Loop Call Registers
- 6-Port Conference Registers
- Simulated Facilities Registers.

➧ CIRCUIT SWITCHED DIGITAL CAPABILITY (CSDC)

A. General

4.54 The CSDC feature provides end-to-end circuit switched transmission capability for 56 kilobits per second (kb/s) data signals. This feature employs an alternate voice-data (AVD) capability which allows the customer to alternate between direct distance dialing (DDD) like voice communication and 56kb/s digital communication on the same circuit path any number of times during any call.

Note: The CSDC feature was formerly known as the Public Switched Digital Capability (PSDC) feature. It is important to note that these terms are used synonymously and both terms are used in this document due to pident references. The CSDC term is used in reference to the feature or for customer identification and explanation.

B. Feature Operation

4.55 An off-hook origination initiates the seizure of an originating register. A line translation is performed and the 2-bit PSDC indicator from the line equipment number class 4 (LENCL-4) word is stored in the originating register. A TOUCH-TONE service receiver is connected to the line and dial tone is applied. Upon receiving a digit, dial tone is removed. If the digit received is #, digit collection is set up to collect two more digits. Upon receipt of two more digits (99) the prefix access code translator is indexed via the dialed digits to determine what service has been requested. A check is made to determine if the line is allowed the CSDC feature. If the line does not have the CSDC feature, special service announcement treatment is given. The "AB" service indicating digits (88) are collected next. Once these digits are received, the associated service index and chart column (CCOL) are retrieved. If the "AB" digits dialed are valid, a check is made to determine if the line has the type of CSDC service requested via the "AB" digits. If the CSDC access code (#99AB) is successfully dialed, the original CCOL in the originating register is overwritten with the CSDC CCOL. A CSDC originating call traffic count is pegged at this time. The CSDC service index is stored in the originating register. A bit is set in the originating register to indicate that the call being processed is a CSDC call. The call is now reinitialized to appear as though no digits have been collected. Dial tone is reapplied and digit collection proceeds until the proper number of digits (7 or 10 with an optional 1 prefix) have been received. Dialing is not considered complete until a final # is received. When the final # is received, it is not stored as a digit. The call is then routed according to the dialed digits on a CSDC outgoing trunk. Answer timing (2 to 3 seconds) is done and an automatic message account (AMA) register is seized. Upon answer, the answer time is recorded in the AMA register. Line-to-line CSDC connections are only allowed via a digital carrier trunk (DCT) digroup that is looped in the office or a DCT trunk to a tandem office that is routed back to the originating office.

4.56 An outpulsing annex is seized and a peripheral order buffer is hunted. An outgoing trunk and a transmitter (if needed) is found. Information on the trunk is stored and a transfer to the appropriate outpulsing routine (either MF or Common Channel Interoffice Signaling [CCIS]) is accomplished. A verification is made to ensure that the originating line or incoming call (for the tandem case) and the outgoing trunk are both CSDC. If this verification fails, retry is inhibited and the call is routed to tone or announcement.

4.57 For MF outpulsing, a junior register is seized, the outgoing trunk is put into the proper states, and start pulsing signal detection is done followed by digit outpulsing. For CCIS, call processing is the same as for a non-CSDC call except for the following actions. The CCIS continuity check (if one is done) is performed in the on-hook state. For line-to-CCIS outgoing trunk calls, the trunk is kept in the on-hook state until answer.

4.58 For an incoming call, the PSDC bit from the trunk class code (TCC) word is stored in the incoming register and a CSDC incoming traffic count is pegged. Digit collection is performed in the same way as a message telecommunications service (MTS) call. Once it has been determined that the call is to terminate locally and the proper number of digits have been collected, a terminating directory number translation is performed. If the terminating line is idle, a check is made to determine if the line is allowed the CSDC feature. If the line is not allowed CSDC, the call is routed to a service announcement. Provided that both the line and trunk have CSDC, a ringing register is seized and the PSDC bit is set in the ringing register. Normal call processing actions are done to provide ringing and audible ringing. Upon ring trip, a talking path is established. For all cases except line-to-CCIS outgoing trunk connections, answer processing is the same as for MTS calls. Upon answer, the incoming trunk is put in the off-hook state to pass answer to the next office.

4.59 Standard disconnect and trunk guard timing is performed on CSDC calls when the called or calling party goes on-hook after a talking path has been established.◆

TANDEM CONNECTIONS PROGRAM

4.60 The tandem connections program (TAND) processes calls requiring a trunk-to-trunk connection through the switching office. Trunk-to-trunk connections may be made

(a) when the ESS switch is used as a local tandem office or as a toll office,

(b) when an incoming call is to a line with temporary transfer to a line outside the office,

(c) to connect a pair of centrex tie trunks, or

(d) to connect an incoming call to a centrex station.

4.61 When TAND is entered, it receives the trunk network number of the incoming trunk and an outgoing trunk route index. It uses the route index to hunt for an outgoing trunk. It restores the incoming register to the idle list and prepares information for the outpulsing program. It then passes control to the outpulsing program. If the outpulsing is successful, TAND restores the transmitter to the idle list and supplies answer and disconnect supervision.

4.62 If outpulsing fails, TAND takes appropriate actions. If the selected transmitter is preempted, the registers associated with the call are released and the incoming trunk is connected to overflow tone. A transmitter may be preempted if no start pulsing signal is received within 4 seconds after it is seized. If the incoming trunk abandons, the registers are released and the path is idled. If there are interoffice signaling problems, an attempt is made to route the call over a different trunk. If a busy condition is encountered, an attempt is made to preempt a transmitter. If this fails or a blocked condition exists, the incoming trunk is connected to overflow tone. If there is a hardware failure, the registers are released and the failure is reported.

SCANNING SOFTWARE

A. General

4.63 The scanning programs supervise the call processing functions by detecting and reporting changes of call states, such as off-hook, on-hook, flash, etc. The scanning group of programs performs scans on lines, junctor circuits, trunk circuits, and selected supervisory points to detect service requests, disconnects, and call-related information.

B. System Scan Control Interface

4.64 All line scanning and nearly all of the trunk scanning functions are performed on base level which is external to the system controlled A, B, C, D, and E frequency classes. Figure 7 illustrates the interfaces of the scanning programs with the ESS switch control programs. Most scanning control flags are set at lower priorities than the frequency class flags in order to decrease scanning rates at certain traffic levels. In the case of trunk scanning, the scan control flags are set at higher priorities than the frequency flags. Trunk scanning is also performed on J level to provide a minimum scanning rate during the time in which long audits are running on the base level. A relative minimum scanning rate for lines is accomplished by performing some line scanning during the class E frequency system jobs. This intervisit time is controlled during system overload conditions.

4.65 System control flags are used to maintain a 50-ms scan rate for inband operator trunks and a 200-ms rate for all other supervisory trunks. A supervisory scan program segments the trunk scanning function so that approximately 1/20 of the scanning work is done per each scanning routine entrance. Every 10 ms, the system main program controls a segment scan of the inband trunks. The two scan rate controls for inband trunks provide the control for distributing the five scanning segments evenly over a 50-ms interval.

4.66 The system also utilizes a control flag for scanning of lines. This flag is set twice every 200 ms and can provide up to 20 additional program scans each 200 ms, the rate depending upon traffic levels.

4.67 The low priority flags are administered in order to decrease scanning rates of both line and trunks simultaneously as system performance varies. The use of the next lowest priority flag also minimizes the probability of performing any maintenance during the time when trunks and lines are not being scanned at their maximum rates.

4.68 System control flags are also used for line scan supervision. These control flags provide for varying the scan rate during traffic overload conditions. As traffic increases, the lowest priority control flag is affected. The decrease in the rate of setting this flag causes the trunk and line scanning rates to decrease simultaneously. As traffic continues to increase, the next lowest priority flag setting rate decreases causing the line scan rate to further decrease. In this case, the trunk scanning rate will remain high because of a high priority control flag.

C. System Scan Hardware Interface

4.69 Input information to the scanning programs, such as on-hook, off-hook, dial pulser, and electrical states of various points within the system during diagnostics and administrative procedures, are furnished by scanners. Central control addresses a scanner via a central pulse distributor. The scanner, in turn, interrogates a group of sensors called ferrods. Their electrical state is then returned to the central control.

D. System Memory Interface

4.70 The scan results are loaded into blocks of memory called hoppers. Registers contain information which identify the call or maintenance procedure.

E. Scan Programs

4.71 Two types of scan functions are performed, namely: (a) supervisory and (b) directed. The

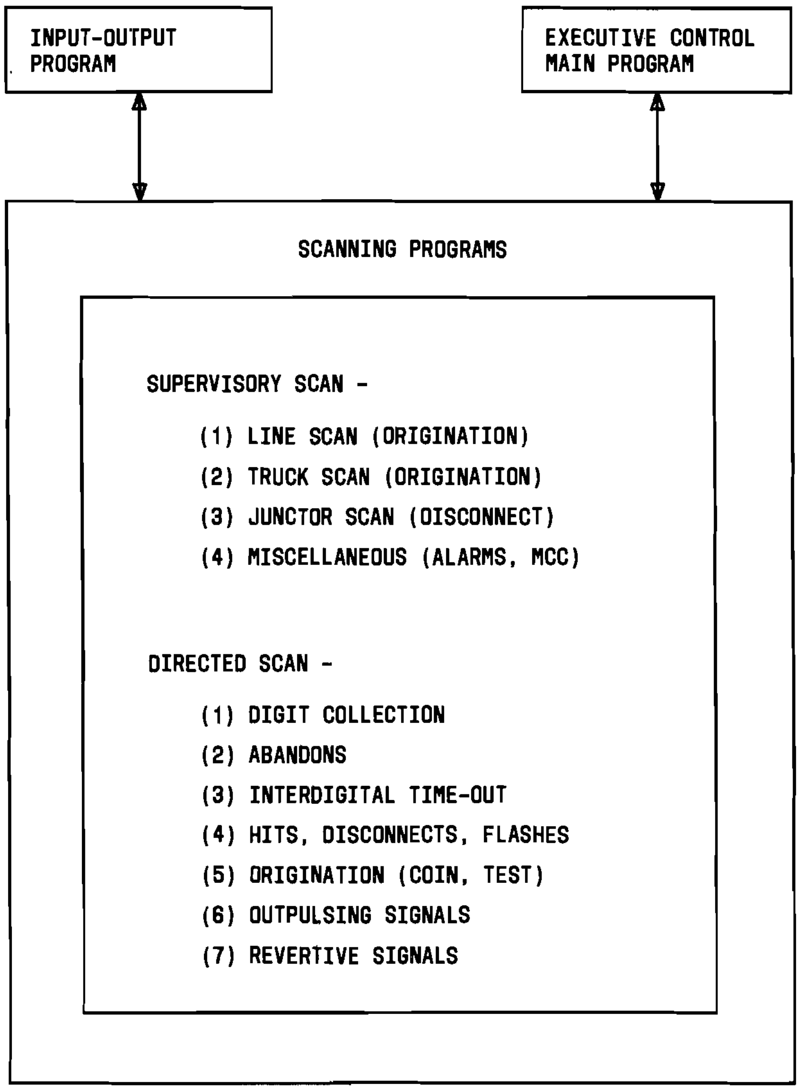

INPUT-OUTPUT PROGRAM
EXECUTIVE CONTROL MAIN PROGRAM
SCANNING PROGRAMS
SUPERVISORY SCAN -
(1) LINE SCAN (ORIGINATION)
(2) TRUCK SCAN (ORIGINATION)
(3) JUNCTOR SCAN (DISCONNECT)
(4) MISCELLANEOUS (ALARMS, MCC)
DIRECTED SCAN -
(1) DIGIT COLLECTION
(2) ABANDONS
(3) INTERDIGITAL TIME-OUT
(4) HITS, DISCONNECTS, FLASHES
(5) ORIGINATION (COIN, TEST)
(6) OUTPULSING SIGNALS
(7) REVERTIVE SIGNALS

Fig. 7—Scanning Program Interface

supervisory scans are those performed on lines, junctor circuits, trunk circuits, and selected miscellaneous points to detect changes of call states. Directed scan routines collect information about the state of a line or trunk during the call process or maintenance procedure.

Supervisory Programs

4.72 Incoming lines are scanned for originations (on-hook to off-hook) at nominal intervals of 200 ms via a line scanner. The ferrods of a scanner row are interrogated simultaneously approximately five times a second. The electrical state of each ferrod is compared with the state of the previous scan. If the change of state indicates an origination (on-hook to off-hook), a report is made to the dialing connection program to connect the line to a receiver.

4.73 Every 100 ms the ringing circuits are scanned to determine whether the called line has answered the call. Whenever the answer (on-hook to off-hook) is detected, the report is recorded in a memory hopper.

4.74 The junctor circuit is scanned to detect disconnect (off-hook to on-hook) on either end of the call connection. The on-hook occurrence is placed in a register to determine if the change is a true disconnect or a hit.

4.75 Universal and inband operator trunks are scanned to detect an off-hook change every 50 ms. The on-hook condition is reported via the trunk seizure and answer hopper. In the case of an on-hook change, program control is given to a routine which determines if the on-hook is a hit.

Directed Programs

4.76 The directed scan programs collect information about the state of a line or trunk at various stages of a call or maintenance procedure. The scan routines include:

- Digit collection
- Detection of abandons
- Detection of interdigital time-out
- Detection of hits, flashes, disconnect
- Origination (coin, test)
- Detection of signals for outpulsing
- Detection of revertive pulse signaling.

OUTPULSING SOFTWARE

A. General

4.77 The purpose of the outpulsing function is to control and coordinate the transmission of signals representing digits required to complete a call between two customers, served by different central offices.

4.78 The outgoing call control programs serve as the main controlling programs for the outpulsing process. These programs act as the interface between the client programs requiring outpulsing and the programs which perform the outpulsing functions.

B. Outpulsing Control

4.79 To communicate with other types of central offices, the ESS switch central office must be able to transmit digit information to each office in a form which the distant office is prepared to accept. Since each type of office is different, several forms of signaling have been developed and used in the central office. The two most common signaling methods are MF signaling and DP signaling. Each of these outpulsing control programs perform the following functions:

(a) Sets the outgoing trunk and transceiver into proper states for outpulsing

(b) Tests the transmission path for continuity and polarity

(c) Transmits trunk seizure signals

(d) Detects start-pulsing signal

(e) Causes the called number (and in some cases the calling number) to be transmitted.

The outpulsing function interfaces with five subsystems in order to handle all aspects of an interoffice call. Scanning, peripheral control, call processing, and translation subsystem interface are essential in completing any type call, while diagnostic programs are called upon in case of any malfunction. This interface is summarized in Fig. 8.

C. Outpulsing Description

4.80 Entry to the outpulsing subsystem is always made by the outgoing call control programs and may come from any one of the following client programs:

- Digit analysis programs
- Add-on and dial conference programs
- Coin control programs
- Trunk and line test panel programs
- Trunk and service circuit diagnostic programs
- Tandem connection program.

4.81 When outpulsing is requested by one of the above client programs, the client program initializes an outpulsing control register (OPCL). This register is used to store outgoing call control infor-

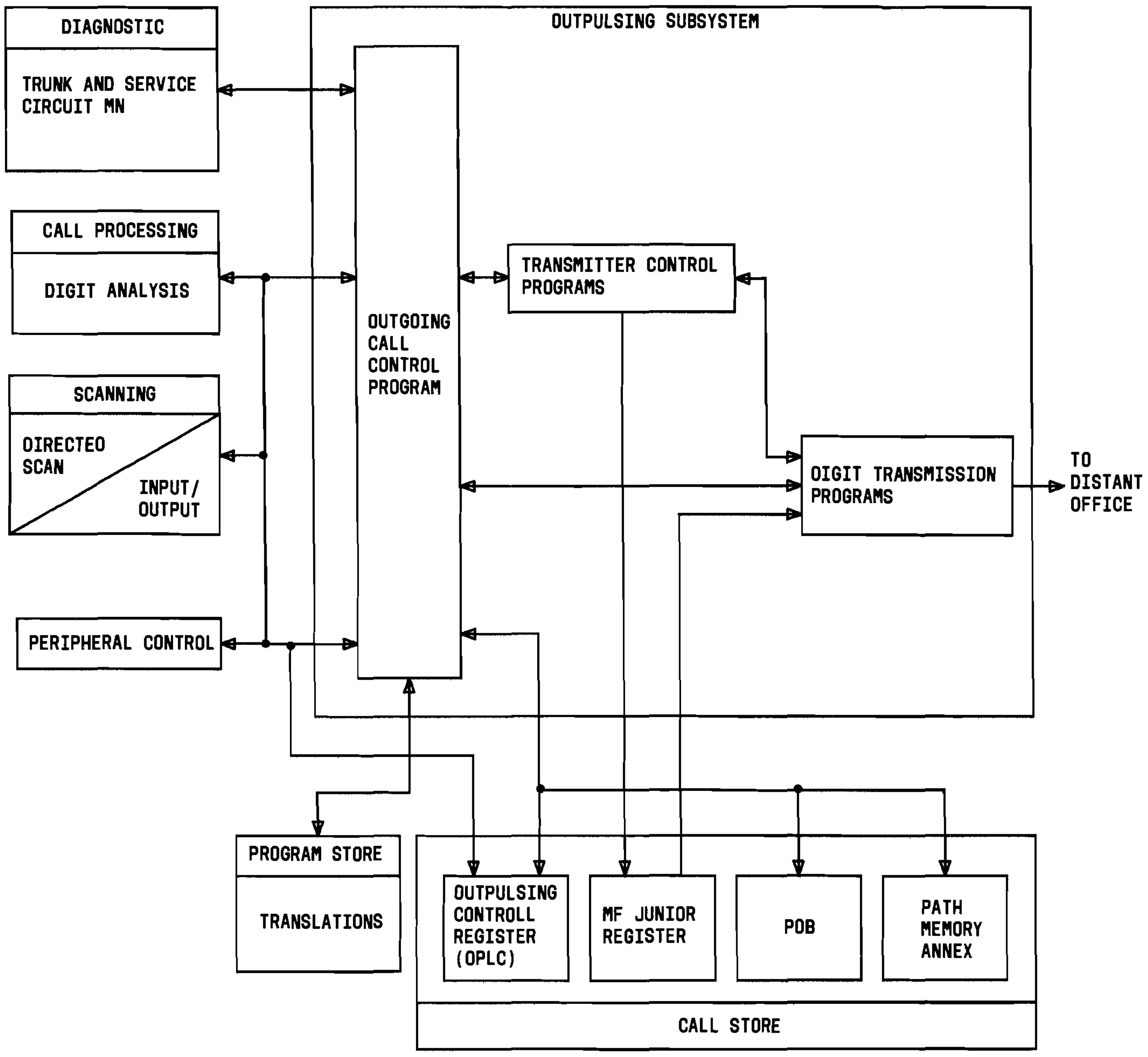


Fig. 8—Outpulsing Interfaces

mation needed during outpulsing operations. The OPCL contains the following input information:

- Trunk hunt control information
- Path memory information
- Digit collected or derived from translations
- ▶Outpulse prefix and delete digit information◀
- Receiver preemption control information
- Return information.

Trunk Hunt Control Information

4.82 The client program furnishes trunk hunt control information through the use of network control programs. This information includes selection of an idle outgoing trunk and transmitter. The selection of a transmitter depends on the type of receiver at the distant office (MF or DP). The network control programs are provided with the ability to find an idle trunk in a group and to make a second trial if all paths to the first selected trunk are busy. The trunk and transceiver locations and the type of outpulsing (type transmitter) are stored in the OPCL.

Path Memory Information

4.83 The network control programs are used to select paths through the network. The busy/idle record of all links in the network is kept in the network map; the network control programs can reserve a path from one terminal to another for use in the call. The paths selected are reserved by changing the busy-idle bits associated with the paths to busy. The path information is stored in the OPCL. The instructions to establish the paths are loaded into the peripheral order buffer.

Digits Collected or Derived from Translations

4.84 The digits collected or derived from translations are the digits to be outpulsed. The number of digits outpulsed and the stage when the outpulsing function will begin are determined by the translation programs. The translation programs may indicate overlap outpulsing, a method of speeding up call completion by starting to outpulse digits before all dialed digits are received. When overlap outpulsing is performed, the client program transfers control to the outpulsing control program prior to receiving all the digits to be outpulsed. This information, as well as the remaining digits to be outpulsed is stored in the outpulsing control register when received by the client.

Return Information

4.85 The OPCL contains the appropriate return addresses and coding to reflect the state of a call at the completion of outpulsing. This information indicates one of two conditions: outpulsing success or outpulsing failure.

D. MF Outpulsing

4.86 The MF outpulsing is a high-speed, long-range method of digit transmission using short-duration bursts of voiceband tones. Speed is gained over other pulsing methods because each digit is assigned a unique pair of tones and need not be built up in time by a series of like pulses, as is done in dial pulsing. Range is extended because voiceband signals can use any amplifying equipment in the trunk over which they travel without special bypassing or relaying circuits.

4.87 Before the MF outpulsing control program is entered, the client program to outpulsing has collected and analyzed the digits which identify the terminating office. The outgoing call control program has performed the following:

(a) Seized an outpulsing annex register

(b) Seized a peripheral order buffer

(c) Called the peripheral order buffer loading programs for changes in network to hunt and busy an outgoing trunk, transmitter, network path, and to load the peripheral order buffer with orders to connect the path from the outgoing trunk to the transceiver.

4.88 An MF junior register (MFJR) is seized and initialized. The peripheral order buffer is loaded with orders to close cut-throughs to the transmitter, putting the outgoing trunk and transmitter into the proper state for outpulsing. The peripheral order buffer is activated, and the MFJR is initialized for digit transmission using the MF outpulsing control program.

4.89 After the peripheral order buffer is successfully executed, preparations are made for digit transmission. If an abandon occurs, the outpulsing process is terminated and control is returned to the client program to take down the connection.

E. Dial Pulse (DP) Outpulsing

4.90 The DP method of digit transmission consists of transmitting trains of direct current pulses representing the called number of the terminating customer line. Each train represents a digit. The numerical value of each digit is represented by the number of on-hook intervals in the train of pulses.

The on-hook intervals of each pulse are separated by short off-hook intervals, while the digits themselves are separated by relatively long off-hook intervals called interdigital intervals.

PERIPHERAL CONTROL SOFTWARE

A. General

4.91 The peripheral control software (Fig. 9) is responsible for directing the central control in those actions necessary to control the switching network. Network actions include the interconnection of lines, trunks, service circuits, and tone sources associated with a central office. The peripheral control routines supply the following services:

(a) Hunt for an idle path through the network

(b) Administer the network map and path memory

(c) Control the loading and execution of peripheral orders in the peripheral order buffers.

4.92 The network is comprised of line link networks and trunk link networks, providing 2-wire metallic paths through eight stages of switching. Of the many discrete paths through the network, there are only three types of paths. Between any two network terminals the paths are line-to-line (L-L), line-to-trunk (L-T), and trunk-to-trunk (T-T). Any network path or connection can be in one of the following path states:

(a) ***Connected:*** If the path is in use and current flows, the path is said to be connected. In order for current to flow through a path, all the network crosspoints that comprise the path must be closed and any trunk or junctor circuit(s) in the path must be in the cut-through state. A record of this connected path is kept in duplicated call store (referred to as path memory). This record permits a determination of which link and junctors are used in the path. The in-use links and junctors are marked busy in the link map area of call store. This record prevents the links and junctors used in the busy path from being used in any other path.

(b) ***Reserved:*** If the path is in use and no current is flowing, the path is said to be reserved. Link map and path memory records similar to those kept for a connected path are retained for this path state. Any number of network crosspoints that comprise the reserved path may be closed.

(c) ***No-Path:*** If the path is not in use, then the path is said to be in the no-path state (idle).

4.93 The three network states mentioned can be configured to form eight transitions for network path(s). These eight transitions and their associated states are listed in Fig. 10. Figure 10 also illustrates the terms and use of the network transitions as they relate to an idle path (no-path), memory, and path hardware. Two of the transitions (C and S) require network and signal distributor relay operations. Another two (T and A) require signal distributor relay release operations. In all cases, the network path crosspoints are closed prior to operating the appropriate signal distributor point. This is to ensure network crosspoints do not switch current.

B. Entry to Peripheral Control

4.94 Entry to the peripheral control subsystem is made from any client program (call processing or maintenance) requiring a network path (or circuit) change, eg, dialing connection, ringing connection, incoming trunk test, etc. This network path change may require a single transition or several simultaneous transitions to complete the proper network path. Therefore, the client will perform one or both of the following functions:

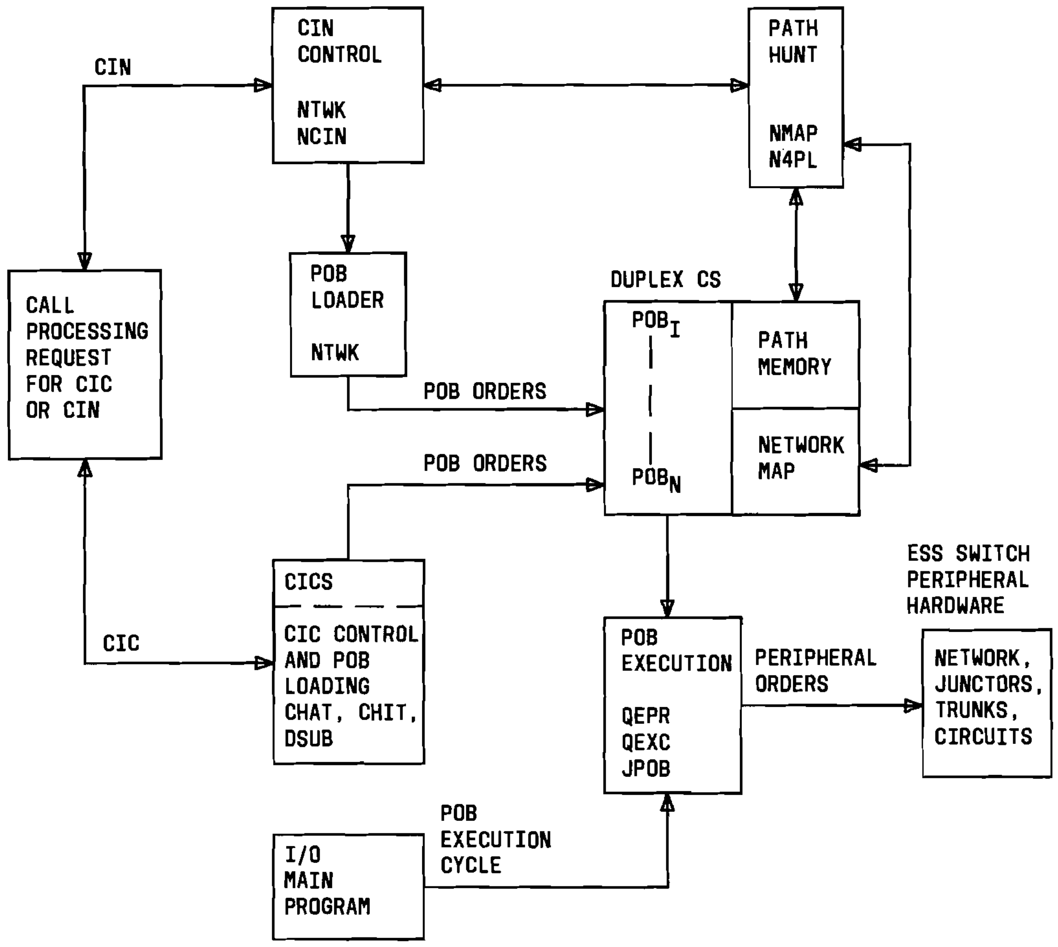
(a) Change in network (CIN)

(b) Change in circuit (CIC).

Change In Network (CIN)

4.95 For each transition involving two or more terminals, there is a unique subroutine called a CIN. This CIN is generally requested with the aid of a macro call. This macro consists of a series of instructions which sets up the calling sequence (or entrance requirements) for a CIN subroutine and subsequent transfer to a sub-CIN subroutine.

4.96 The CIN programs transfer program control to the network path hunt program when a request to find an idle path between two terminals is made. This hunt for an idle path is accomplished by interrogating the activity or busy/idle bits in the network link map. The network link map contains a



LEGEND:

CHAT	- CHANGE IN CIRCUIT HEAD AND RELAY ACTION TABLES
CHIT	- CHANGE IN CIRCUIT TABLES
CIC	- CHANGE IN CIRCUIT
CICS	- CHANGE IN CIRCUIT SUBROUTINES
CIN	- CHANGE IN NETWORK
CS	- CALL STORE
DSUB	- DISTRIBUTOR AND SCANNER SUBROUTINES
JPOB	- J-LEVEL PERIPHERAL ORDER BUFFER
NCIN	- CHANGE IN NETWORK PROGRAM
NMAP	- NETWORK MAP ADMINISTRATION PROGRAM
NTWK	- NETWORK TRANSITION CONTROL PROGRAM
N4PL	- NETWORK TABLES
POB	- PERIPHERAL ORDER BUFFER
QEPR	- POB ADMINISTRATION
QEXC	- QUEUE EXECUTION

♦Fig. 9—Peripheral Control Subsystem♦

OLD STATE (FROM)	NEW STATE (TO)	TRANSITION
NO-PATH	RESERVED	RESERVE (R)
NO-PATH	CONNECTED	CONNECT (C)
RESERVED	NO-PATH	ERASE (E)
RESERVED	RESERVED	HOLD (H)
RESERVED	CONNECTED	SET-UP (S)
CONNECTED	NO-PATH	ABANDON (A)
CONNECTED	RESERVED	TO-HOLD (T)
CONNECTED	CONNECTED	KEEP (K)

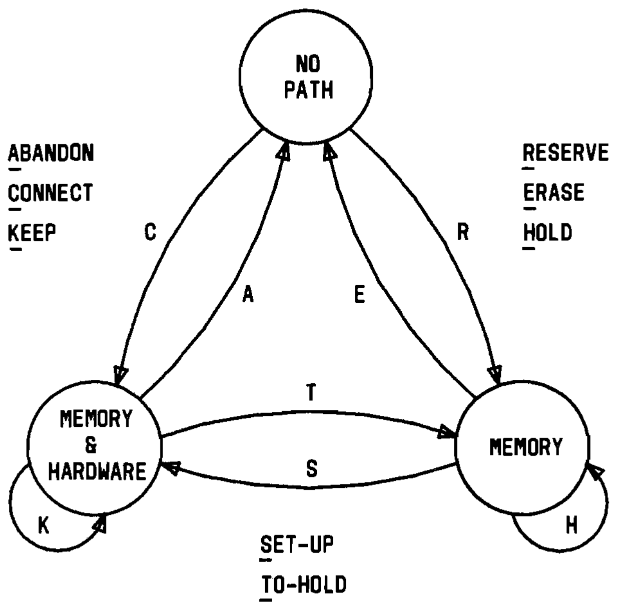


Fig. 10—Network Path Transitions

busy/idle indication for all links in the ESS switch network. A path from one terminal to another can be reserved by changing the busy/idle bits associated with the path to busy. This path selection process is performed by the network path hunt programs.

Change in Circuit (CIC)

4.97 For each transition involving the control of trunk(s) or service circuits via a signal distributor point, there is a unique subroutine called a CIC. The CIC expands program orders into the required data to effect the desired circuit state change. The CIC subroutine also provides distributor and scanner subroutines which load peripheral order buffers, and perform scanning and central pulse distributor operations without a peripheral order buffer.

4.98 After the appropriate peripheral orders are formed (by the CIN or CIC routines), they are placed into a peripheral order buffer. The peripheral order buffer holds the orders in the proper execution sequence until activation time; then the peripheral order buffer execution program unloads the orders and sends them to the peripheral system.

C. Peripheral Order Buffer Execution

4.99 The purpose of the peripheral order buffer execution programs is to control the rate at which signal distributor, network, central pulse distributor, and scanner orders are sent to peripheral units. These orders perform the function of switching telephone traffic. Prior to peripheral order buffer execution, these orders are placed, at the request of a client program, in a peripheral order buffer. During every fifth J-level interrupt (approximately every 25 ms) the peripheral order buffer execution program sends the orders loaded in that peripheral order buffer to the appropriate peripheral units. However, with incoming step-by-step trunks this normal peripheral order buffer scheme is not fast enough to recognize an abandon and an immediate reseizure. Therefore, J-level peripheral order buffers are set up which rapidly handle orders associated with these bylink trunks. The speed increase is due to J-level peripheral order buffers being loaded in J-level rather than base level. The J-level peripheral order buffer execution precedes normal peripheral order buffer execution. The peripheral order buffer execution programs can be divided into the following functions:

- Memory usage
- Peripheral order buffer administration
- Peripheral order buffer execution
- J-level peripheral order buffer actions.

OPERATOR FUNCTIONS

A. General

4.100 The operator functions programs (Fig. 11) process all calls requiring operator assistance or intervention. The actions of the operator function programs are described in the following paragraphs.

B. No Test Calls

4.101 Operator no test calls are handled by pident OFNT. No test calls allow an operator to verify that a line is legitimately busy or break in on an existing call in an emergency. When an operator originates a no test call, OFNT sets up a connection from the operator to the desired line, if possible, or returns a signal to the operator if a connection is not possible. This may occur if there is trouble on the line, blocking due to traffic, or if the line has temporary transfer active.

C. Emergency Manual Line (EML) Service

4.102 The emergency manual line service program (OFML) allows the operator to provide service to critical customers manually by bypassing the switching machine in case the ESS switch is severely overloaded or inoperative.

D. Miscellaneous Outgoing to Switchboards and Desk

4.103 The miscellaneous outgoing to switchboards and desk program (OFGT) sets up connections to various trunks such as recording completing, intercept, business office, and repair service. It disconnects the digit receiver, seizes an operator register, and reserves a network path and a trunk of the appropriate type. When an operator answers, it gives the operator a class of service tone about the calling line and sets up the previously reserved path between line and operator.

E. Toll Switch Origination

4.104 Calls originated by an operator, for example, to complete a toll call, are handled by OFTR. It seizes an incoming register to store needed information. If the called line is not busy, OFTR seizes an operator register, transfers information from the incoming register to the operator register, and releases the incoming register. The OFTR then sets up a path to a ringing trunk, and establishes a talking path if the called party answers. The OFTR also processes coin collect and return, ringing control, and disconnects.

F. Traffic Service Position System (TSPS)

4.105 The TSPS program handles all calls routed to a traffic service position. This includes spe-

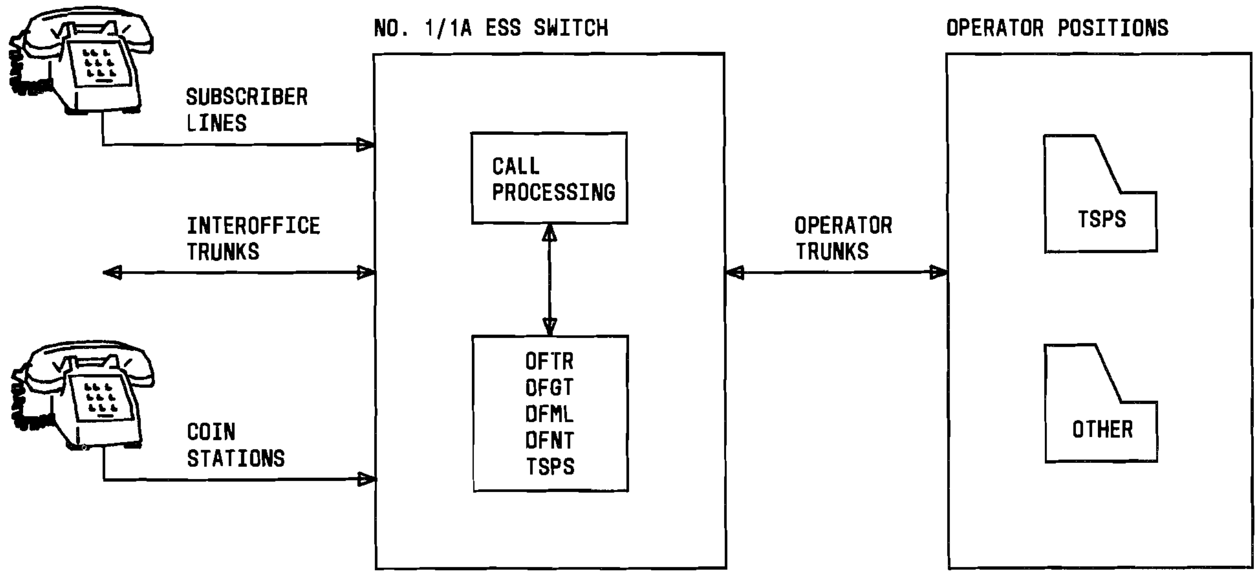


Fig. 11—Operator Functions Software Subsystem

cial toll calls such as collect or person-to-person calls. It interfaces with OFTR to seize an operator register and connect a ringing circuit. The TSPS also has entries which place a 911 emergency trunk or a regular TSPS trunk which is on maintenance onto the high and wet list.

G. Toll Operator Signaling

4.106 The toll operator signaling program (TOPR) handles signaling to and from switchboards. These signals are sent over intertoll trunks and access toll switches include ringback, ring forward, TSPS double ring forward, and alerting wink. The TOPR is entered from the scan point change director program when a trunk ferrod changes states. It takes appropriate actions based on what type of signal is detected. It also handles answer and disconnect.

CHARGING SOFTWARE

A. General

4.107 The AMA facilities of the ESS switch (Fig. 12) provide a means of collecting certain billing data related to the charging of customer calls.

B. Collecting Local AMA Data

4.108 Once the call processing programs determine that a particular line origination is requesting a billable service, an attempt is made to put an AMA register on the call. Pident AMAC is entered from call processing to seize an AMA register. Once the AMA register has been linked to the call, billing information such as the calling directory number and called directory number is transferred from the originating register (OR) to the AMA register. Subsequent data entries to the AMA register include call type (CTYP), answer time, special service marks, disconnect time, etc. Time of day information is supplied

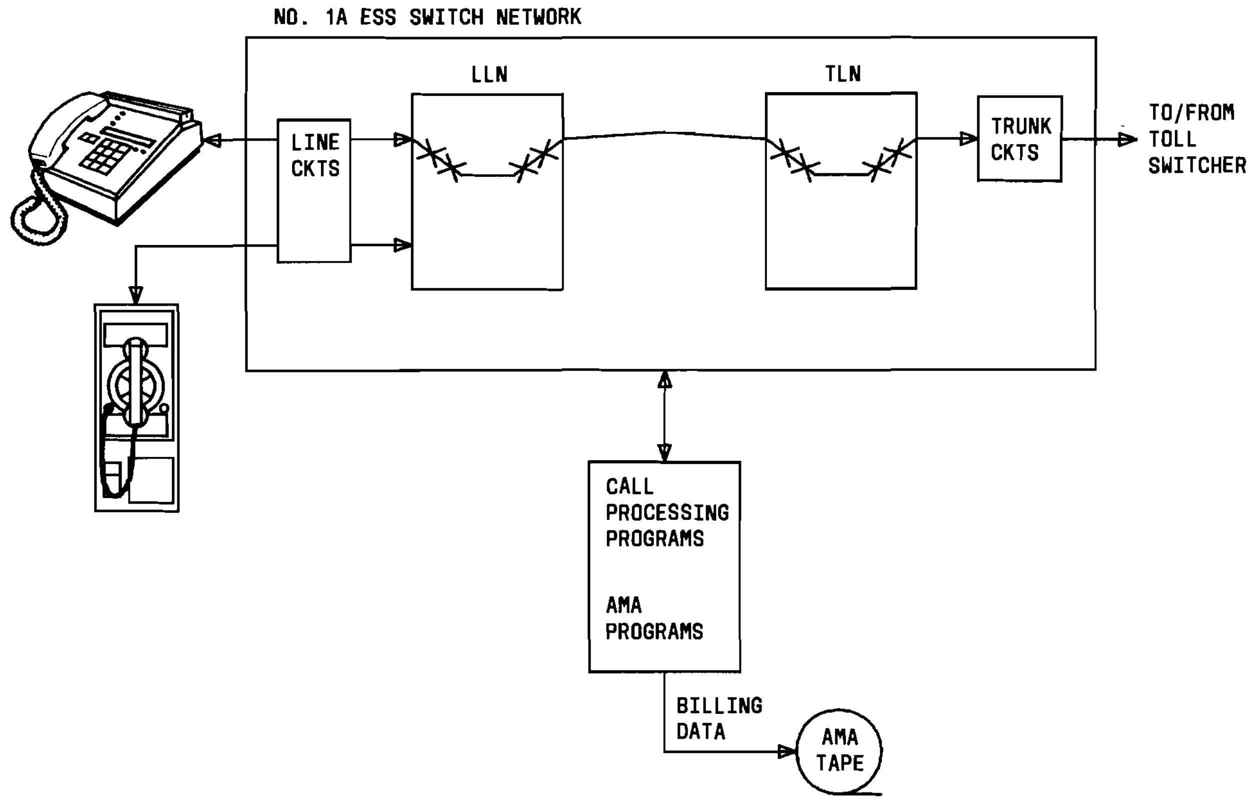


Fig. 12—AMA System

by the ESS switch system clock, in units of hours, minutes, seconds, and tenths of seconds.

4.109 After the call has terminated and all the required billing information is in the AMA register, the AMA data accumulation program (AMAC) organizes the AMA register data into a binary coded decimal (BCD) format for storage in the AMA buffer area of duplicated call store. The stored BCD formatted information for each call must be a multiple of 5 BCD digits. When required, the information is padded (extended) to a multiple of 5 BCD digits. Padding is done with noncheck dummy (NCD) characters. The AMA buffer, administered by AMAC, consists of a dedicated block of words for duplicated call store and additional words of call store are provided for overflow. After each AMA register is reformatted into the buffer, AMAC checks for a buffer-full condition. If full, AMAC sets up an entry to the AMA data transfer program (pident AMDX) which will unload the AMA buffer and transfer the data (which is reformatted again by AMDX) to another buffer for subsequent output to tape.

4.110 Normally only completed calls are recorded by the AMA system. However, the call processing programs are allowed to insert a number of special marks in the OR. These marks, when present, will cause incomplete calls to be recorded.

4.111 The AMA programs collect the charging data for all automatically billable calls. Charging and other call related data are initially collected into an OR by the call processing programs. Call processing then enters AMAC to associate an AMA register with the call. At the time of initial seizure of an AMA register, AMAC is called as a subroutine of the call processing client currently in control of the call. This is because AMA is concerned only with billing functions and not with such things as calling subscriber abandon or network path configuration. The AMAC is entered on outgoing calls just before the outgoing call control program is given control. On intraoffice calls, eg, message rate, AMAC is entered just before the ringing program gets control.

4.112 The AMA register initialization requires that an OR be on the link of registers serving the call, although any standard call register, eg, coin charge, may be master. Initial billing information, such as calling directory number, billing index or chart class (depending upon service marks in the OR), and called directory number, is copied from the OR into the AMA register. Upon entry, AMAC attempts to associate a 13-word AMA register (depending on the type of billing) with the call.

4.113 If all AMA registers are busy because of high traffic, AMAC decides whether the call will be allowed to complete. Generally, if AMAC fails to seize an AMA register, then single unit message rate and coin calls are allowed to go with no billing data collected. All other calls are given reorder tone. If there is an AMA hardware failure (resulting in a lack of buffer space and AMA registers), then all calls are allowed to go with no billing data collected.

C. AMAC Interfacing

4.114 Once an AMA register is put on a call, returns to AMAC from the call processing programs (to report things such as answer, disconnect, abandon, etc) are via a program tag (PT) table, The PT is a pointer set by AMAC into the AMA register. The PT defines the proper return entry to AMAC from the calling program.

D. AMA Register-to-Buffer Processing

4.115 After all billing data for a particular type of call has been collected in an AMA register (note that this does not necessarily imply that the call has terminated), AMAC is reentered via the program tag table. For example, if billing data on a call has been completed with a disconnect time, the disconnect program (pident DISC) reenters AMAC via the program tag. Following checks for available space in the AMA buffer, AMAC builds the AMA register data into the required data formats and stores the formatted data in the buffer. The data formatting routines, based on the type of billing data collected, select other routines which in turn select the formats, ie, data groups, to be built for the particular call type. After the formatting and buffer loading is complete, the buffer input pointer (which points to the next available buffer entry slot) is updated and buffer full condition is checked. If full, AMA_GO sets a queuing condition to queue AMA registers waiting to be unloaded and sets up a call to the AMA data transfer program.

4.116 If AMA register queuing occurs and an AMA register does not clear the call register queue within 1 to 6 minutes, an audit will dump it off. Since only about 0.5 second is required for AMDX to copy a full buffer to tape, if an AMA register is stuck on the queue for longer than a couple of seconds, then a major AMA hardware failure has occurred and some AMA data will be lost.

E. AMA Midnight Routine

4.117 At midnight, AMAC is entered from ECMP, and all AMA registers in the office are checked for traffic for the day's tape. Registers which are busy, ie, collecting billing data on a call, are found by testing their PT values. The AMA buffer is readied for a new day's activity by doing a final write to tape. This end-of-day processing results in a clearing of the AMA buffer with the buffer input pointer reset to the first buffer slot.

4.118 For any in-process AMA call, other checks are made to determine the number of midnights passed since the call entered the system. If an AMA register is busy on a call at midnight, the register's midnight counter (AMA register item MNT) is scored. This counter has a maximum count of nine. Midnight counts of 2 to 9 generate an AMA output message on the maintenance TTY. The message includes the calling directory number, number of midnights passed since the call entered the system, and the address of the AMA register serving the call. If the MNT count reaches 10, the AMA register is taken off the call with a dummy code entered for disconnect time. The call is left up.

4.119 Other checks are made to catch disconnects which occur just before midnight but after the MNT counters are scored. This situation could occur if a call was in the "timing for final disconnect" state at midnight.

COIN FUNCTIONS

A. General

4.120 The coin programs handle charging for local and coin zone calls from coin stations. These stations may be dial-tone-first or coin first. The functions performed by coin programs include the following:

(a) Seize and release coin charge registers and, if needed, AMA registers

(b) Test for presence of initial and overtime deposits

(c) ➧Call either ringing and answer detection programs or outpulsing programs to➧ connect the coin station to called line

(d) Determine call rates and timing intervals

(e) Collect initial and overtime deposits

(f) Test for local overtime deposits

(g) Connect and alert an operator when an additional deposit is required

(h) Disable TOUCH-TONE service when connected to an operator to prevent fraudulent use of the TOUCH-TONE service signals

(i) Time initial period ➧on local and coin zone calls➧ and overtime periods on local calls

(j) Collect/return coins at disconnect

(k) Provide disconnect on coin calls.

➧Coin calls are classified into three charging classes:

(a) Local Coin Call: A call within the coin customers' local calling area.

(b) Coin Zone Dialing: An arrangement whereby multiple message unit calls may be dialed from a coin telephone without the necessity of dialing a 0+ or 1+ prefix. Coin zone calls terminate outside the local calling area but generally do not qualify for toll rates.

(c) Toll and Special Services: A toll call, is a call that is neither local nor coin zone and is usually routed through a TSPS office. Special service calls are directory assistance, repair service, 911, dial 0, etc.

Toll calls from coin lines require the assistance of either a toll operator or TSPS; hence, they are classified operator assisted.

LOCAL COIN

A. Operation

4.121 The origination stage of a local coin call is processed by the dialing connection program in a normal manner and the dialing stage of a local coin call is processed by the digit analysis program also in a normal manner. Immediately after the last digit is dialed, the digit analysis program transfers the call over to a coin charge program (COIN). The

COIN program first seizes a coin charge register (CNC) and initializes it. The CNC administers charging for the coin call. The COIN program then checks whether the coin station is dial tone first or coin first. If the station is dial tone first, COIN tests the coin station for presence of the initial deposit. The initial deposit test is not performed on coin first stations because coin first stations cannot produce an origination signal without the presence of an initial deposit. If the coin customer remitted the initial charge, COIN invokes either the ringing and answer detection programs or the outpulsing programs for connecting the coin line to the called line. If the coin customer did not satisfy the initial charge, the coin programs route the coin line to the insufficient deposit announcement which informs the coin customer to remit the proper rate and redial the call.

4.122 The COIN program regains control of the local coin call when the called party answers. After regaining control, COIN checks whether the call type is local coin with overtime or local coin without overtime. If the call type is local coin without overtime, COIN marks the call chargeable, releases the CNC and waits for the call to end. When the call ends, COIN collects the initial deposit and disconnects the call. The disconnect programs perform the bulk of the disconnect actions. If the coin customer hangs up before the called party answers, the initial deposit is returned and the coin line is disconnected.

4.123 Local coin with overtime call type consists of one initial period and zero or more overtime periods. Durations of both the initial and the overtime periods are specified by the operating company. Upon regaining control of a local coin with overtime call type following an answer, COIN marks the call chargeable and begins timing the initial period. If the call ends during the initial period, the coin program collects the initial deposit and disconnects the call. Disconnect of a local coin with overtime call type is handled entirely by COIN. The disconnect programs are not used.

4.124 Twenty-four seconds before the initial period expires, COIN collects the initial deposit. The COIN program then waits 24 seconds and tests the coin station for overtime deposit. If the overtime deposit is present, COIN begins timing the overtime period. If the overtime deposit is not present, COIN connects both parties to a local coin overtime announcement which prompts the coin customer for the overtime deposit. The COIN program waits another 24 seconds following the announcement and performs a second overtime deposit test. If the overtime deposit is present this time, COIN begins timing the overtime period. Otherwise, COIN disconnects the call.

4.125 Twenty-four seconds before the overtime period expires, COIN collects the overtime deposit, waits 24 seconds and tests the coin station for the next overtime deposit. If the overtime deposit is present, COIN program begins timing the next overtime period. Otherwise, COIN connects both parties to the local overtime announcement, waits 24 seconds following the announcement and performs the next overtime deposit test. If the test passes, COIN begins timing the next overtime period. If the test finds no deposit, COIN disconnects the call. The COIN program repeats this overtime period sequence for as many overtime periods (all overtime periods are of the same duration) as the call lasts.

4.126 When the call ends, COIN collects the overtime deposit and disconnects the call. This disconnect is also handled entirely by COIN.

4.127 An operating company may optionally use an overtime monitoring operator (OMO), in lieu of the local overtime announcement for overtime deposit prompting.

COIN ZONE

4.128 Origination and dialing stages of a coin zone call are handled in the normal manner. After the last digit, control of the call passes to the COIN program. The COIN program seizes and initializes a coin charge register, seizes a coin zone operator (CZO), reserves a network path between the CZO and the coin line, and connects the coin line to audible ringing tone. When the CZO answers, COIN program reswitches the coin line from the audible ringing tone to the CZO. The CZO informs the coin customer how much to deposit for the initial period and monitors the deposit. When the CZO is satisfied with the deposit, the operator disconnects from the call thus signaling COIN to complete the call. Upon receiving this signal, COIN disconnects the coin line from the CZO and invokes either the ringing program or the outpulsing program to connect the coin line to the dialed party.

4.129 The COIN program regains control of the coin zone call when the called party answers.

If the coin customer hangs up before the called line answers, the initial deposit is returned and the call is disconnected.

4.130 Upon regaining control of the coin zone call following an answer, COIN marks the call chargeable and begins timing the initial period. Twenty-four seconds before the initial period ends, COIN collects the initial deposit. The COIN program then waits 24 seconds and connects both the coin customer and the terminating party to the CZO where the parties remain until the call ends. If the call ends during the initial period timing, COIN collects the initial deposit and disconnects the call.

4.131 The CZO measures the duration of the overtime period. When the coin zone call ends, COIN signals the CZO that the call ended. The CZO then computes the overtime charges, informs the coin customer (the CZO may have to ring the coin line) to deposit the computed charge and monitors the deposit. When the coin customer remits the correct amount, the CZO instructs the customer to hang up and disconnects. Upon detecting both the coin line and the CZO on-hook, COIN collects the overtime deposit and disconnects the call.

A. Toll and Special Services

4.132 This class of coin calls is not processed by the coin charge program but by programs providing special services. Toll calls from coin lines are routed either to toll operators or to TSPS with TSPS being the more common method. The TSPS/toll operators complete the call to the desired destination and perform all charging functions for the call. The operators determine the cost of the call, prompt the coin customer for the proper deposit, and collect the deposited coins.

4.133 A typical toll call flows as follows:

(1) Following the last digit, the digit analysis program returns any deposited coins (this is an office option) and connects the coin line to TSPS.

(2) The TSPS computes the cost of the initial period, informs the coin customer to deposit the computed charge, monitors the deposit for proper amount, and completes the call when the initial period charge is satisfied.

(3) When the terminating party answers, TSPS begins timing the initial period.

(4) Upon timeout, TSPS collects the initial deposit and informs the coin customer that the call has entered into overtime and that overtime period charges will be levied.

(5) The TSPS then begins timing the overtime period.

(6) When the call ends, TSPS computes the cost of the overtime period based on the measured duration, informs the coin customer to deposit the computed charge, monitors the deposit of coins for proper amount, collects the deposit and disconnects.

(7) Upon detecting an on-hook from the TSPS, No. 1/1A ESS disconnect program disconnects the coin line from the TSPS.

4.134 The TSPS does not perform coin collects directly as the previous paragraph alluded to. Rather, the TSPS transmits a coin collect request signal to the No. 1/1A ESS switch. The TSPS service program in conjunction with the coin control (COCN) program in the No. 1/1A ESS switch performs the actual coin collect function. The TSPS has the capability to request coin collects, coin returns, and ringbacks. The coin return request is used to return the initial deposit when the terminating party does not answer. The ringback request is employed for recalling the coin customer back to the phone should the customer hang up at the end of a call instead of flashing.

4.135 Operator-assisted toll calls (such as person-to-person where only a "0" is dialed) are handled similarly to TSPS toll calls. The digit analysis program returns any deposited coins (an office may elect to retain the deposit and credit it toward initial period) at the end of dialing and connects the coin line to a general assistance operator. The operator obtains any further information required for completing and billing the coin customer and completes the call. The operator then times the duration of the call, computes the charges for the call, requests deposits from the coin customer and at appropriate stages of the call, requests the No. 1/1A operator programs to collect coins, return coins, or ringback the coin line. The operator programs in conjunction with coin control programs temporarily seize control of the call, perform the operator requested coin function, and return the control of the call back to the operator.

4.136 Special service calls such as directory assistance and business bureau calls from coin lines are routed similarly to calls from noncoin lines with two exceptions. The first exception is at the completion of dialing. Any deposited coins may be returned, collected or retained at the telephone company option and on a per trunk group basis. The second exception (arises at the end of a coin call), occurs when any coins retained at the end of dialing are returned at the end of the call.◀

B. Coin Station Testing

4.137 Programs are also provided to enable maintenance personnel to test a coin station. When a special number is dialed from the coin station, the coin line is connected to a coin station test line which then tests the operation of the ground removal relay in dial tone first phones and may optionally also test:

- Loop resistance
- Coin collection
- Coin return
- Operating time of coin collect relay.

4.138 The programs provide coded rings to indicate the results of the tests. If the coin station is taken off-hook during these rings, the program prepares to receive a digit requesting another test. It also begins timing. If the craftperson does not request a test within 60 seconds, the call is disconnected. If a digit is received, the program initiates the test associated with that digit.

TRANSLATIONS SOFTWARE

A. General

4.139 The translations program and data package provide a means of translating information from one form to another. For example, a telephone is identified to a customer by a directory number. At the line side of the network, the line termination connecting this particular telephone is assigned a line equipment number. Translating from line equipment number-to-directory number or directory number-to-line equipment number is a primary task in many call processing activities. The translations system provides this and numerous other translating services.

4.140 The translation system consists of a data base and collection of control routines which provide specific kinds of translated information on request from other programs. When the requesting (ie, client) program requires service, it provides the translation system with the necessary input information (Fig. 13). The translation data base, in simplex call store, is interrogated like a dictionary by the translation control routines. The translation output is found and delivered to the client in a predetermined fashion. The output may consist of the following:

- Addresses of equipment numbers
- Numerical quantities
- Generic translation data
- An output indicating a special situation.

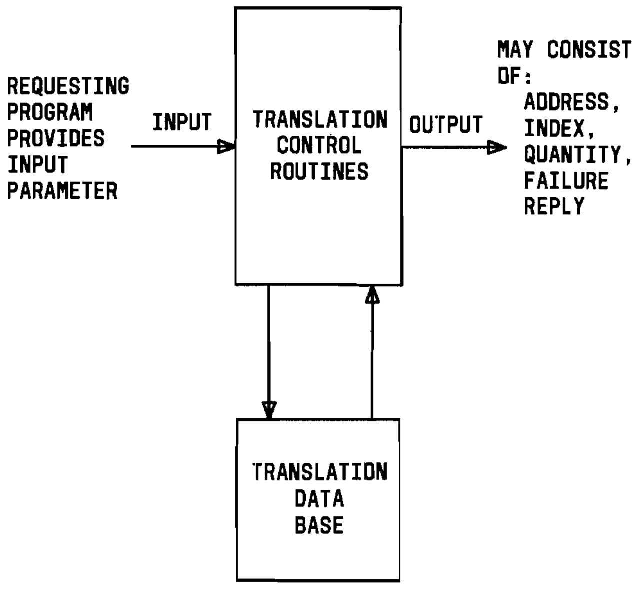

REQUESTING PROGRAM PROVIDES INPUT PARAMETER
INPUT
TRANSLATION CONTROL ROUTINES
OUTPUT
MAY CONSIST OF:
ADDRESS, INDEX, QUANTITY, FAILURE REPLY
TRANSLATION DATA BASE

Fig. 13—Translation System

B. Addresses of Equipment Numbers

4.141 Addresses of equipment numbers are required by the client to do specific jobs—such as, operating a relay using a central pulse distributor or a signal distributor, reading a line or a trunk scan point, setting up network paths, etc.

C. Numerical Quantities

4.142 Numerical quantities specify directory numbers, trunk group numbers, billing indexes, and route pattern numbers. These quantities have the same number of bits in all offices but may lead to different actions in each office.

D. Generic Translation Data

4.143 Generic translation data (translation data for determining call processing operations) has fixed numerical quantities; therefore, a particular binary number has the same meaning in all offices. For example, part of the output information from a line equipment number translation indicates if the line is for TOUCH-TONE service. A particular group of output data bits provides this information. These bits are interpreted in the same manner in every office by the call processing programs.

E. Special Output Situations

4.144 Special output situations are the result of finding a special condition when making the translation. For example, if the directory number is found unassigned, the call will be routed by the client program to intercept treatment. In this case, the return from the translation program is made to a different point in the client program than when an assigned entry is detected.

F. Translators

4.145 A translator consists of all of the data connected with a particular type of input. Because of growth considerations in any office, most translators are broken into subtranslators linked together by a head table. In most cases, the head table is referenced by the master head table. A subtranslator corresponds to a growth unit of the input. For instance, since the quantity of line equipment numbers in an office depends on the number of remreed line switch circuits, the line equipment number subtranslator contains translation information for 1024 line equipment numbers. In general, the translation input is divided into two parts: the subtranslator selector, and the subtranslator index. The selector is the number of the subtranslator in the head table, and the index is the number of the item (word number) in the subtranslator. In some cases, the translation information required may not fit into the limited subtranslator space (one translation word) so auxiliary blocks and/or expansion tables may also be needed.

4.146 A full translator can consist of a head table, subtranslators, scattered auxiliary blocks, abbreviated codes, subauxiliary blocks, and associated expansion tables.

Locating Translators

4.147 A specific translator can be identified in memory by locating its address in the base translator block, ie, the master head table. The base translator block is located in simplex call store at a fixed Compool defined address and with a specified length. The master head table consists of the first 200 (octal) words of this fixed block. The master head table, containing addresses of other tables, serves as a directory for locating the translators.

Translator Types

4.148 Translation data exists in simplex call store as a structured collection of tables. The basic table entry is the 24-bit translation word. The set of tables devoted to the conversion of certain input data into certain output data is called a translator. Examples are line equipment translator, directory number translator, numbering plan area code and office code translators, trunk network number-to-trunk group number translator, trunk network number-to-peripheral equipment translator, etc. Each set of tables (a translator) is linked according to a hierarchic pattern. Tables high in the hierarchy contain pointers to (addresses of) lower tables. The lowest tables in the hierarchy (such as the subtranslator, the auxiliary block, or the abbreviated code expansion table) contain the actual translation data. Translators may be divided into two general types: multilevel and expansion.

Multilevel Translator

4.149 The multilevel translator consists of a maximum of six levels of information in a hierarchy. These levels are as follows:

Level	Name
1	Head table
2	Subtranslator
3	Primary translation word

4	Auxiliary block or expansion table
5	List
6	Subauxiliary block

4.150 The first level in the hierarchy is the head table. This head table serves as a directory for locating all translation data related to the input. This table usually contains an address of a subtranslator.

4.151 The second level in the hierarchy is the subtranslator. Each subtranslator corresponds to a growth unit of the telephone office. For instance, there is a subtranslator per line switch frame or circuit, per trunk switch frame or circuit, per 1000 directory numbers, etc.

4.152 The binary representation of the input to the multilevel translator is divided into two parts: the subtranslator selector and index. The selector and index can be visualized three different ways.

(a) The selector identifies the unit and the index identifies the item within the unit.

(b) The selector is the number of the subtranslator and the index is the number of the item within the subtranslator.

(c) The selector is the word number in the head table which contains the address of the subtranslator; the index is the word number in the subtranslator which contains the primary translation word.

4.153 A subtranslator is a table which consists of one translation word per index. This word, the primary translation word, is the third level in the hierarchy and contains either the complete data associated with the input, or if one translation word is insufficient, an auxiliary address to an auxiliary block or expansion table (which is the fourth level). When the primary translation word contains an auxiliary address, the complete data associated with the input are contained in the auxiliary block words. To recognize an auxiliary address, the four most significant bits of the subtranslator word contain zero. Therefore, any subtranslator primary translation word for which the four leftmost bits are zero is interpreted as an auxiliary address.

4.154 The auxiliary address, in the primary translation word (PTW) is the address of the word in the auxiliary block which contains the word number (WRDN). This WRDN either indicates the length of the auxiliary block if the block is 31 words or less or indicates that the block exceeds 31 words. The WRDN is located in the six most significant bits of the first word of the auxiliary block. When the block exceeds 31 words, the WRDN is 000000 and is located in the second word of the block. The true length of the block is then located in the word preceding the auxiliary block.

4.155 Another level, the fifth level of the translator, is the list (eg, speed calling list). The list is used when additional information other than the standard found in the auxiliary block is needed. The list, like the subtranslator, has an associated index. The address of the list may be found in one of the data words of the auxiliary block.

4.156 The last level is the subauxiliary block. The subauxiliary block is used if one word is insufficient to contain the information found in the list (eg, a 10-digit number for a speed call list is contained in a subauxiliary block). Therefore, like the subtranslator, the list may contain a subauxiliary address instead of the data usually found in the list.

Expansion Translator

4.157 The expansion translator has only one table, called an expansion table or simply a table. This type of translator has only an index as its input. The translator is no more than a list of words. Like the subtranslator in the multilevel translator, it may or may not have auxiliary blocks associated with it.

4.158 An example of the expansion translator is the unit type lengths table which is located in the base translator block. The unit type lengths table is a list of words containing the number of subtranslator translation words per unit type. The input is the unit type number. Therefore, to find the quantity of subtranslator words for any unit type, the unit type is used as the index. For example, for unit type 5 the quantity of units is found in the sixth word of the unit type lengths table.

4.159 The following translators are not pointed to by addresses located within the master head table or auxiliary master head table. Each of these translators has a fixed location in unduplicated call store which is defined in Compool:

- Master head table

- Head table lengths table
- Unit type table
- Unit type lengths table
- Junctor circuit number-to-junctor network number head table
- Junctor network number-to-junctor circuit number head table
- Junctor network number-to-junctor network number/junctor circuit number.

Indexing into the master head table, unit type table, head table lengths table, and unit type lengths table translators is accomplished by utilizing fixed coding which is a function of the generic program algorithms.

➧Call Store and Program Store Memory Expansion Feature

4.160 The memory expansion program increases the memory spectrum to maximum size supported by the 1A processor to 20 million octal words. With the expanded memory size, call store is expanded to 32 K-codes and program store to 30 K-codes. Call store is split into three parts:

(a) Duplicated call store (DCS): Contains transient information

(b) Lower unduplicated call store (LUC): Contains translation information

(c) Higher unduplicated call store (HUC): Contains translation information.➧

RECENT CHANGE (RC) SOFTWARE

A. General

4.161 The RC subsystem (RCSS) consists of a collection of recent change message interpretation programs designed to accept recent change (RC) input messages, check them for validity, and generate the proper translation data for storage in translation memory. The RCSS provides maintenance and service order with a means of modifying the translation data to reflect changes in customer service requirements and in the internal system. Figure 14 provides a generalized system diagram of the RCSS.

B. RCSS Functional Description

4.162 Inputs to the RCSS may occur simultaneously from up to four different sources:

- Customer lines
- The service order activation phone
- The office RC channel, and
- The service order TTY (includes paper tape or magnetic cartridge)
- Maintenance TTY.

These requests for RCSS service may originate for any number of reasons and, in the case of inputs from customer lines, are not under central office control. However, the end result in each case is that the translation data is dated to reflect requested changes.

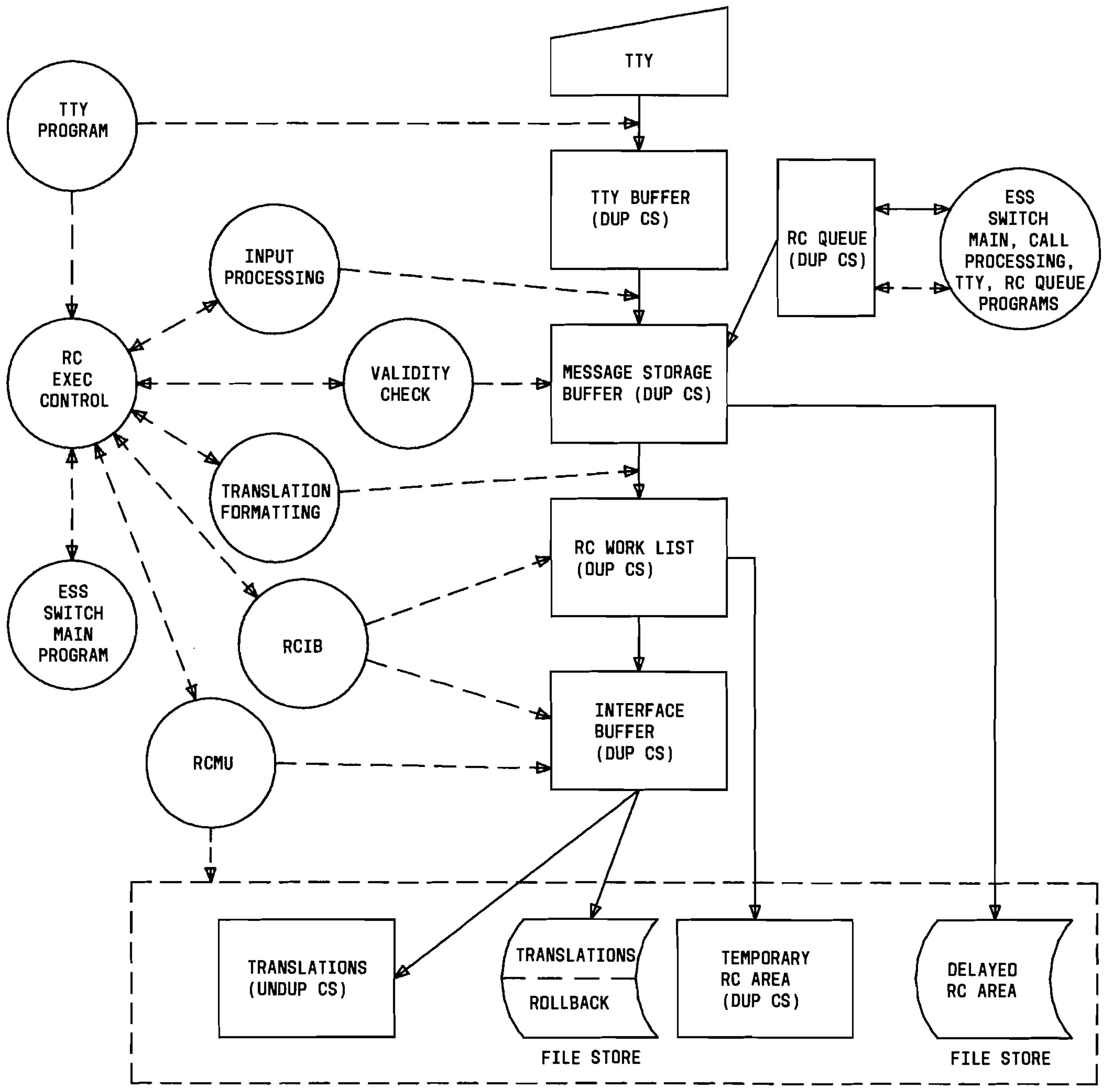
C. Translation Data Storage

4.163 Translation data is stored in unduplicated (ie, simplex), call store. Recent changes with permanent status are stored in the translations data area of unduplicated protected call store with two complete backup copies stored on disk in the file stores in No. 1A ESS switch. Recent changes that are temporary (eg, call forwarding changes) are stored in the temporary RC area in duplicated call store. File store also provides for storage of delayed recent changes and translation rollback data, ie, a copy of the original translation data (before modification) of every translation word affected by a permanent status RC.

D. Queuing of RC Inputs

4.164 Speed calling requires the RCSS to handle an increasingly large volume of inputs. The basic philosophy is that customer originated requests for changing a speed call list entry will be queued in a searchable queue until the RCSS is free to process the entries.

4.165 The RC queue consists of a dedicated block in duplicated call store whose size (length in call store words) is defined in parameters and, thus, is office engineered. The first word in the queue is reserved for TTY input requests. Should a request from the TTY be received while the RCSS is busy pro-



LEGEND:
DATA ———
PROGRAM CONTROL — — —
RCIB - RECENT CHANGE INTERFACE BLOCK BUILDER
RCMU - RECENT CHANGE MESSAGE UPDATE

Fig. 14—No. 1A ESS Switch Recent Change Subsystem

cessing a queue entry, the TTY channel number is loaded into the first queue word position. The RCSS scans the queue from top to bottom, thereby giving the highest priority to the TTY. Only the TTY channel number is held in the queue, as character buffering is accomplished in the TTY buffer. The remaining queue entries are processed in the order of their occurrence in the queue. Conversely, the queue is read from bottom to top in response to a request from the speed calling program. This ensures that the most recent information is returned for call processing purposes; eg, it is possible that two queue entries for the same speed calling list entry could exist at the same time. Reading from bottom to top yields the most recent information.

E. Customer Line Inputs

4.166 The queue control program in the RCSS interfaces directly with the call processing programs which interpret speed call list changes from customer lines. When a customer dials the proper access code to change a speed call list entry, the call processing program decodes the request and stores the change information in a holding register. The RCSS is then requested to load the change information into the RC queue. The RCSS then ensures that the request is valid and can be loaded into the queue. Once the queue entry is processed, the speed call list change is active and can be accessed as necessary by the speed calling program.

4.167 At the completion of processing a queue entry, the RC queue is checked for a waiting TTY request for service. If no TTY requests are present, the next queue entry is unloaded and processed. After processing an RC message from the TTY, the RCSS will always check the queue for any entries waiting to be processed, delete the entry that was just processed, and move the remaining entries up.

F. Inputs From the Office RC and Service Order TTYs

4.168 Most of the changes to the translation data in the No. 1A ESS switch memory will be the result of RC messages input from the TTYs, either manually or via paper or magnetic tape. RC messages entered via the TTY may have either immediate or delayed status. Unless entered as a delayed message, the RCSS will immediately process the message, format new or modified translation data, and then update the translation data in call store. If the RC is of a permanent type, the translation data in unduplicated call store is changed. The translation data image in file store is then brought up to date.

G. Rollback

4.169 There can be instances where an RC message or several RC messages affect the translation data in such a way as to degrade or disrupt system activities. To correct problems of this nature, it is necessary to "undo" the RCs suspected of causing the problems by reinstating the original translation data, as it appeared before the RC message was input. This operation is termed rollback.

4.170 Each time an RC message affecting the permanent translation data in unduplicated call store is entered, a rollback block (RBB) is built by the RCSS and stored in the rollback area on disk. The disk rollback area is a dedicated block of disk storage whose length is Compool defined. Each RBB, identified by an internally generated RC order number, contains the address and original data of each translation word affected by the associated RC message.

H. Rollback Area Management

Type 1 Rollback

4.171 Each RC message that is accepted by the RCSS and is to be inserted into the translation data base creates a single RBB in the rollback area. Each RBB contains the address and old data for each word of translations altered by the RC. The RBB is created and stored in the file store rollback area before any updating of the translation data takes place. This is done in case the update process is for any reason interrupted (eg, interrupt, disk return failure, etc). If the update should fail before the entire update is complete, the RBB data is used to remove the partially completed RC. This kind of rollback is referred to as Type I Rollback. Type I rollback is automatically initiated by the RCSS to remove a partially completed RC.

Type II Rollback

4.172 Type II rollback is the process of removing or canceling RCs which successfully update translation data and are effective with respect to call processing. Type II rollback must be manually requested either from the TTY or via the master control center (MCC).

Rollback Data Storage Concepts

4.173 The rollback area is logically circular; ie, the physical beginning and end of the rollback area act as though they were adjacent memory locations in a ring buffer. Thus, as RBBs are stored and the rollback area fills up, additional RBBs overlap the first RBBs stored causing that particular rollback data to be lost. To avoid this overlap, office personnel regularly write a new translation tape. This tape is a "snapshot" of translations data at that particular point in time. When this tape dump is performed, all RBBs which were created prior to the old latest system tape are discarded, freeing up a large (approximately 40 percent) portion of the rollback area. Associated with the rollback area are three indexes or pointers: TAPE1, TAPE2, and NEXT. The NEXT index points to where the next RBB will be stored when a new RC message is successfully processed. The TAPE2 index shows where the NEXT pointer was when the latest translation tape was written. The TAPE1 index, likewise, marks the position of the NEXT index when the next to the latest translation tape was written.

4.174 If the latest translation tape dump had been followed by RC messages 13, 14, 15, 16, and 17, then to restore translations to the state that existed when the latest tape was written, it is necessary to rollback RCs 17, 16, 15, 14, and 13. This is a rollback to TAPE2. If RC 7 was the first RC following the TAPE1 index, then to restore translations to the state that existed when the next to the latest tape was written, RCs 12, 11, 10, 9, 8, and 7 must also be rolled back. This is a rollback to TAPE1. It is not possible to rollback beyond the time of the writing of the next to the latest tape; ie, RBBs 6, 5, 4, 3, 2, and 1 (those preceding the TAPE1 index) cannot be rolled back and should be considered nonexistent.

4.175 When a new translation tape is written, the rollback area tape indexes are repositioned to reflect the existence of a new translation tape and that the old latest translation tape is now the next to the latest translation tape. Assume that a new translation tape has just been written and that no RCs have been input since. The TAPE2 index is now pointing to the same position as is the NEXT index and the TAPE1 index is moved up to the old TAPE2 position. At this point, a rollback to TAPE2 (the latest translation tape) would have no effect since no RCs have been input since the latest translation was written. A rollback to TAPE1, however, could be done. This would rollback the RCs existing between the TAPE1 and TAPE2 indexes.

4.176 It is necessary to write new translation tapes on a regular basis to avoid filling up the rollback area and overtaking the TAPE1 or TAPE2 indexes. The frequency of writing translation tapes depends on the RC activity level in a particular office. Tape dumps may have to be performed every other day for some offices and once a week for others. The rule is that when the portion of the rollback area used since the latest tape dump reaches 40 percent, a tape dump should be performed as soon as practical, eg, that same evening. The percentage of the rollback area used since the latest tape dump is reported by an "RC:WARNING" output message when the percentage exceeds 40 percent and by the REPT:RC CENSUS output message as an hourly printout or in response to the "OP:RCCENSUS" input message. Since no indication is given of how much of the rollback area has been used since the next to the latest tape was written, the 40 percent rule must be strictly adhered to or the TAPE1 index can be overtaken without warning. Should the TAPE1 index be overtaken, it is moved up to coincide with the TAPE2 index with a resulting "REPT:RC WARNING" output message stating that a rollback to the next to the latest tape (TAPE1) is no longer possible. If additional RCs are entered beyond the 90 percent mark until the TAPE2 index is overtaken, the TAPE1 and TAPE2 indexes will be moved forward as many RBBs as necessary to store the new RBB. A "REPT:RC WARNING" output message will state that complete rollback to the latest system tape (TAPE2) is no longer possible. A partial rollback to TAPE2 can be performed, but the rollback is limited by the TAPE2 index which has been moved and no longer represents the state of translations that existed when the latest translation tape was written.

4.177 Type II rollback can be performed via the MCC with a Phase 4, 5, or 6 or by means of a TTY input message without going through a phase. There are four modes of Type II rollback by MCC request:

- Rollback to a particular order number
- Rollback to a specified number of orders
- Rollback to the TAPE2 index, and
- Rollback to the TAPE1 index.

4.178 Type II rollback via the TTY is done without a phase; of course the ESS switch must be sane enough to allow input/output via a TTY channel for this method to work. TTY rollback is restricted to 20 orders per rollback request, and only as far back as the TAPE2 index; ie, TTY rollback is limited to a rollback to the latest translation tape.

4.179 Each RBB has an internal order number assigned to it before being stored in the rollback area. This RCSS assigned order number is completely independent of the user specified order number entered with the ORD keyword in the RC message. The internal order number is of the following form:

MMDDSSSS

where MM = month (1-14s); DD = day of month; SSSS = daily sequence number (0-7777s). The internal order number, in addition to providing a unique identifier for each RBB, also gives information on when the RBB was created and stored in the rollback area.

4.180 Each day at midnight, the date is updated and the sequence number is reset to zero. This numbering scheme is useful in that it defines what can be called "safe rollback points." Assuming that all RC activity is halted around the hour of midnight, due to the working hours shift change and that translations is in a valid state so far as can be determined, it should always be safe to rollback to midnight of any particular day. This is done by rolling back through order number MMDD0000.

4.181 For example, assume that all RC activity for June 26 ended at 10:00 p.m. and the system seemed to be in good shape. The next day RC activity resumed and at 3:20 p.m. that afternoon serious system problems began to appear with indications of translation data mutilation due to RC errors. The decision is made to rollback RCs to clear the problem. Rollback of an arbitrary number of RCs is dangerous because the resulting state of translations at the completion of the rollback is not known. Rollback to TAPE2 or TAPE1 is maybe too drastic, possibly causing disruption of service to a large number of customers. Rollback to midnight of the previous day, a known safe point, is recommended in this situation. An MCC rollback in a phase 4, 5, or 6 through internal order number 06330000 (octal) is performed without delay resulting in the removal of the bad RC message(s). When the bad RC is isolated and corrected, the day's RC messages can be reentered using the paper tape or magnetic cartridge backup. But, as a result of the rollback, speed calling list changes will be lost and each affected customer will have to reestablish his list. Call forwarding RCs are kept in the temporary RC area, produce no rollback data, and consequently, are not affected by rollback. The example illustrates the importance of the rollback data to extricate an office from service disrupting problems caused by faulty translation data. It should emphasize the importance of keeping the rollback data clean.

4.182 An easy way to generate messy rollback data and run the risk of generating translation data errors is to try to fix a bad RC with additional RCs. The use of "RC:PSWD" is often abused in this way. The proper way to correct a bad RC is to rollback the order and then reenter it correctly.

I. Delayed RCs

4.183 A delayed RC message may be entered by specifying DELAY in the message format. The entire message is processed normally, including syntax and data checks, but no translation data updating is done. Instead, the message is compressed and stored in the delayed RC area in file store. The message is not effective until the RC message is activated. Activation is accomplished by another RC message from the TTY or via the service order activation phone.

J. Inputs From the Service Order Activation Phone

4.184 A central office telephone set whose major class of service is activate service order may be used to activate delayed RCs. The delayed RC is activated by dialing the order number, nnnnn, filled out to five digits with leading zeros. If no delayed message with the dialed order number is found, reorder tone is heard. If the RCSS is busy and cannot process the activation code, busy tone is heard and the user may try again. If the delayed message associated with the dialed order number is found in memory, dial tone is returned and other order numbers may be dialed as appropriate.

> ***Note:*** Receipt of dial tone does not guarantee that the delayed message was activated.

The TTY should be checked for an ACPT RC18 output message indicating successful activation.

QUEUE AND GENERAL PURPOSE SOFTWARE

A. Queue Administration Program (WQUE)

General

4.185 While processing a call, the system may encounter various busy conditions. When these conditions occur, the call is delayed by having it wait in a queue or list until the register or hardware necessary to continue the processing of the call becomes available. The queue administration program (WQUE) loads the various queues when requested by client programs due to busy conditions, and unloads the same queues when the equipment or registers are available. If a call is abandoned or a time-out has occurred, the item can be removed from a queue at the request of the client program.

4.186 Subsystems interfacing with the queue administration program are shown in Fig. 15.

Functional Description

4.187 This program administers two types of queues: the fixed length queue and the variable length queue. A fixed length queue is a call store block of memory engineered to a length dependent

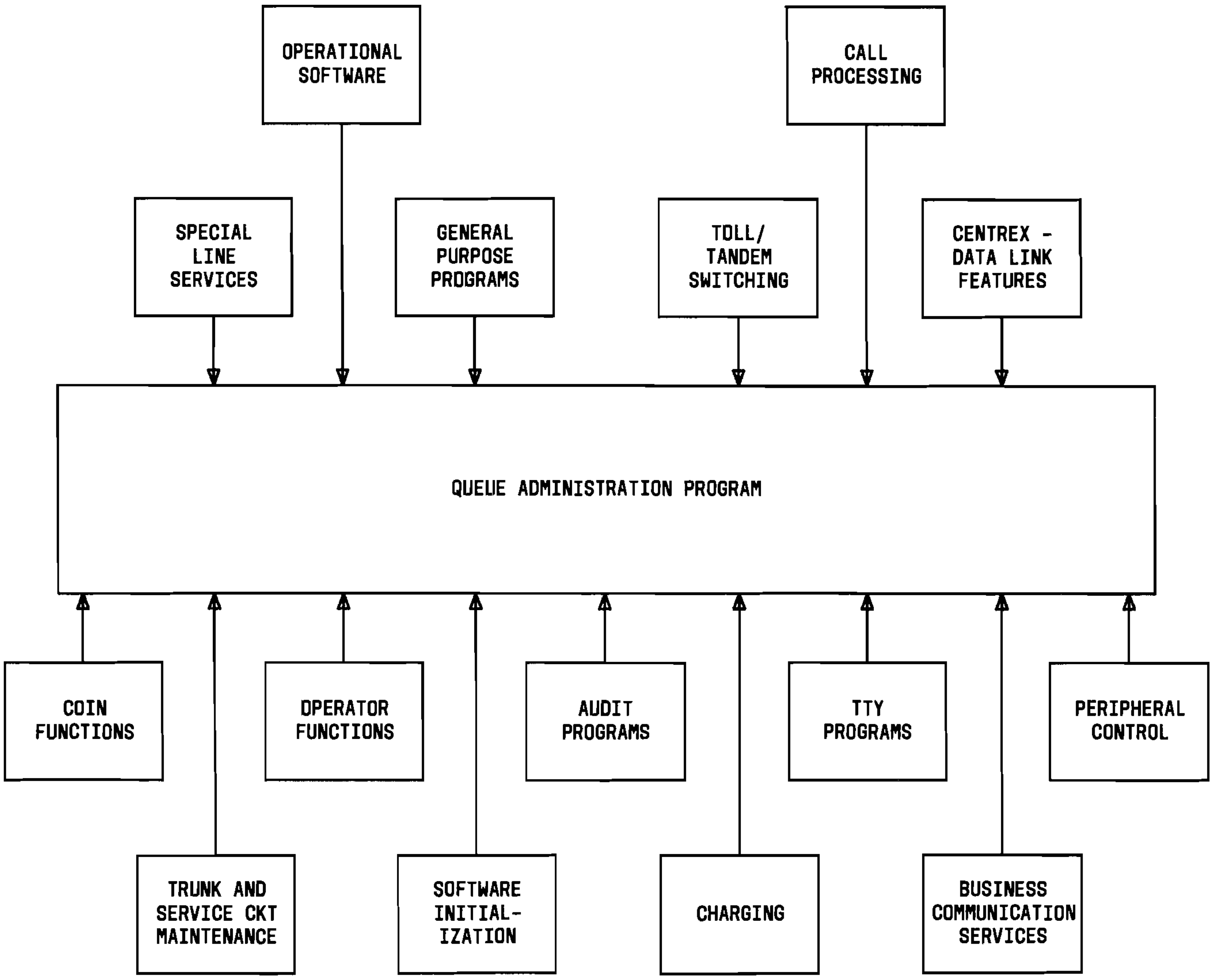


Fig. 15—Subsystem Interfacing With Queue Administration

upon the office size and traffic requirements. This block contains information on customers unable to be served due to the unavailability of equipment. Each fixed length queue has a 4-word head cell: two words for loading information (load pointer and load bottom pointer), and two words for unloading information (unload pointer and unload bottom pointer).

4.188 A variable length queue is a 1-way list of call registers which have been linked together waiting for release of equipment, release of call store memory, a time-out, etc. This type of queue requires a 2-word head cell. The first word contains the address of the first client register on the queue, and the second word contains the address of the last client register on the queue. The linkage is completed via the queue word of each register which contains the address of the next register on the variable length queue.

4.189 When an item is loaded on either a fixed length queue or a variable length queue, the associated executive control main program flag for the specific queue is set, indicating that an item is now on a specific queue. With the flag bit set, the executive control main program returns periodically to the unload entry for the queue until the queue is empty, at which time the flag bit is turned off.

4.190 As a result of an abandon, a time-out, etc, a client can be removed from a queue before the executive control main program unloads the entry. In such cases, an entry on a fixed length queue is removed by zeroing the location in the queue. Linkage to the location is established via the client register's queue word which contains the address of the client register entry on the fixed length queue. An entry on a variable length queue is removed from the queue by searching from the beginning of the list until the client register address to be removed matches the contents of the queue word of one of the registers on the list. At this point, the list is updated by substituting the next client register address on the list for the one just removed from the queue word.

B. General Purpose Programs

General

4.191 General purpose programs perform preliminary work and particular functions for client programs. Some of the general purpose programs provide service routines which return control to the client programs when execution is complete. Others perform jobs of considerable magnitude and remain in control of a call during real-time breaks.

Functional Description

4.192 Following is a brief description of the major functions performed by the general purpose programs.

➧*Supervision Modernization (SUPERV)*

4.193 The SUPERV programs deal with the analysis and delivery of call control signal to client programs. Signaling information and input/output structures are controlled by these small group programs, resulting in simplification of call processing functions. These programs also allow the introduction of a nonscanner controlled signaling system, such as CCIS, without requiring a dedicated software interface.

4.194 The SUPERV programs are concerned with:

(a) Control and reporting of trunk seizures and answers

(b) Control and reporting of line and trunk disconnects

(c) Control and reporting of line and trunk reanswers

(d) Processing of operator wink, flash, and other information signals.➧

Incoming Trunk to Busy, Overflow, or Special Service Circuit Program (YFTO)

4.195 Other programs use the YFTO program to connect an incoming trunk to busy tone, regular overflow tone, common overflow tone, or a special service circuit trunk such as high tone, low tone, announcement, or milliwatt test circuit. Upon entry to YFTO, a network path involving the incoming trunk may or may not exist. If a path does exist, the YFTO program will abandon and erase the path involving the incoming trunk and then connect the trunk to a tone or special service circuit trunk. Upon successful completion of the connection, supervisory scans for detecting the disconnect of the incoming trunk are initiated.

4.196 If the requested connection is to a busy tone trunk and a blocked or busy condition is encountered, the YFTO program attempts a connection to common overflow tone. If the requested connection is not to a busy or an overflow tone trunk, an attempt is made to connect to common overflow tone if a blocked condition is encountered, or to a busy tone if a busy condition is encountered. If a blocked or busy condition is found during an attempt to connect to common overflow tone or regular overflow tone, the incoming trunk is placed on High and Wet until it disconnects.

Originating Line to Busy, Overflow, or Special Service Circuit Program (YTTO)

4.197 The YTTO program is used by other programs to connect an originating line to busy tone, regular overflow tone, common overflow tone, or a special service circuit trunk such as high tone, low tone, announcement, or milliwatt test circuit. Upon entry to YTTO, a network path involving the originating line may or may not exist. If a path does exist, the YTTO program will abandon and erase the path involving the originating line and then connect the line to a tone or special service circuit trunk. Upon successful completion of the connection, supervisory scans for detecting the disconnect of the originating line are initiated.

4.198 If the requested connection is not to an overflow tone trunk and a blocked or busy condition is encountered, the YTTO program attempts a second connection—this time, to common overflow tone. If a blocked or busy condition is found when an attempt is made to connect to common or regular overflow tone, the call is abandoned and all memory associated with the call is idled. This will result in the originating line receiving dial tone if it remains off-hook.

Scan of Single Master Scanner Point Program (YFDS)

4.199 Client programs use the YFDS program to determine if a scan point on a master scanner is busy or idle. The client program provides the master scanner number of the scan point to be scanned at entry to YFDS. The scanner number is expanded to determine the enable address, the row, and the answer bit position of the scan point. Orders are sent on the peripheral bus to read the scan point. Upon completion of the scan, the YFDS program notifies the client program of the scan point condition (busy or idle).

Report and Miscellaneous Subroutines (COPR)

4.200 The COPR program consists of numerous subroutines which perform various functions in connection with call register administration. Services provided for client programs include the following:

(a) Link, in a circular linked list, a specified register which is not already linked to the list to a specified register which may or may not be already linked.

(b) Unlink a register from a circular linked list, release the register and restore it to its idle linked list, and pass on a report of a change in state to the next register on the linked list.

(c) Accomplish charge delay timing for master registers. This timing is required on all coin charge, hotel-motel charge, and AMA message rate-without-overtime calls.

(d) Grant or deny clearance to a requesting nonmaster register.

(e) Set supervisory scan points upon return from a successful peripheral order buffer execution.

(f) Set disconnect supervision on a line-to-line, line-to-trunk, or trunk-to-trunk connection.

Seize and Release Routines and L-, J-, and T-Bit Administration (YAHA)

4.201 This program seizes a register from its idle linked list or releases a register and restores it to its idle linked list for client programs. The kinds of registers provided for include call registers (originating, disconnect, etc), junior registers (MF junior, etc), and auxiliary memory blocks (path memory annex registers, etc).

Register Linking Routine (YCLK)

4.202 This program links a given register to a single register or to a circular linked list of call registers on a single call.

Miscellaneous Register Subroutines and Tables (YMRG)

4.203 General purpose program YMRG consists primarily of linked list register hunt routines, register identification-program tag (RI-PT) routines, and general release register routines.

4.204 The linked list register hunt routines refer to the parameter area data in program store to deliver to a client program, one after another on repeated entries, the addresses of all registers of a specified type. These routines are used primarily by audit programs in their checking or rebuilding of register linked lists but are also used by call processing programs.

4.205 The RI-PT routines are used to select a return address to a call processing program on the basis of information stored in the state word of a call register. The reason for the return, that is, the type of information conveyed, determines which of four methods of return is used:

(a) The disconnect method is used to report a disconnect (ie, scan point change from off-hook to on-hook).

(b) The timing method is used to report the removal of a call register from a timing linked list upon time-out.

(c) The queue method is used to report the removal of a register from a queue.

(d) The standard method is used to report any expected event except the three preceding cases. For example, a change from on-hook to off-hook is reported by the standard method.

4.206 The general release register routines are designed to release a single register or all registers in a circular linked list of call registers on a single call, or all registers but one in such a circular linked list.

Call Store Zeroing Program (ZERO)

4.207 This program is used to rapidly zero a block of adjacent call store locations. Client programs may enter program ZERO at any point to zero the appropriate number of call store words.

C. Call Trace Program (TRCE)

General

4.208 The call trace program uses information in call store to find the terminal (line, trunk, or service circuit), if any, which is connected to a known terminal. On failure to find the sought terminal in a connected path, the trace program gives an indication of the state of the known terminal in the system.

4.209 Call trace may be requested via various software functions or via TTY input messages to identify a line equipment number or trunk network number if it is connected to either a known directory number or a known trunk.

Functional Description

4.210 This program is functionally divided into two routines, one of which is used to find the terminal connected to a line, and the other to find the terminal connected to or reserved for a trunk.

Call Trace for Lines

4.211 The input to call trace for lines is a valid line equipment number or remote equipment number (for remote switching system). After determining that the given line is marked busy, the trace program searches for the line equipment number in the path memory associated with the given line link network.

4.212 When the line equipment number cannot be found and the line bit is busy, flags are set for audits to investigate and, if possible, correct the state of the line bit and path memory.

Call Trace for Trunks

4.213 The input to call trace for trunks is a trunk network number. If the trunk is in a stable and valid path, the program hunts for the other terminal.

4.214 Failure to find the trunk network number on a list or in a busy state implies that the path memory for trunks is not correct. A flag is set requesting an audit which restores the trunk to the trunk idle linked list, if necessary, and updates path memory.

D. Queuing Calls to Multiline Hunt Groups

General

4.215 An incoming call directed to a multiline hunt group is immediately serviced if there are any idle hunting group terminals. If all terminals are busy, the call is placed on a queue and waits its turn. While waiting, the calling party receives audible ringing tone. The call that has been on the queue the longest will be the first call served when a line becomes available. (The number of calls that can be queued for a multiline hunt group depends on the number of queuing registers provided for that group.) Up to three call waiting lamps may be provided on the customer's premises for the purpose of informing the customer of the length of time calls are on the queue before being answered. If an incoming call cannot be placed on the queue, the calling party will receive busy tone unless the incoming call is a foreign exchange call. In this case, the new call is placed on the high and wet list (not connected to busy tone) to prevent charging.

Functional Description

4.216 Following is a brief description of programs functional in the queuing of calls to multiline hunt groups.

(a) Queue and Administration Processing (QAPR): This program contains basic routines responsible for determining if a call should go on queue and if there is room on the queue for a call. In addition, these routines start the process to give tone to the customer. The QAPR program also contains general queue unloading routines. Other routines, which are entered from ECMP every flagged class D, perform on-queue processing functions, determining if there is an idle agent and, if so, taking a call off the queue and terminating the call to the idle agent. Additional routines detect timeouts and disconnects.

(b) Give Audible, Disconnect, and Line Termination Routines (QTAL): This program contains routines which, depending on the type of origination, call the proper routines to perform switching, determine what type of tone or announcement the customer gets, and finish putting the call on queue. For a call that fails to get the necessary facilities when coming off the queue, this program contains routines to give reorder to the customer, given the type of incoming facility. Other routines prepare the call coming off the queue terminating to an intraoffice call to flow into the normal legs as if the call were never queued. There are a number of general routines that perform such functions as unlinking and releasing ringing registers and selecting audible route.

(c) Customer Interface and Special Auditing Routines (QCIA): This program contains a routine to determine the oldest call on the queue. It also performs special auditing functions.

(d) Queue State Information Features (QSIF): This program contains a routine that places into the queuing register the time that a call was placed on queue. In addition, there are routines connected with the call waiting lamp feature that determine states and send lamp orders.

SPECIAL SERVICES SOFTWARE

A. General

4.217 With the advent of stored program control to provide telecommunication services and control of switching networks, many special services have become available to the telephone customer. As with any programmable system, the services available are determined by the particular programs provided. Thus, the services available for any particular ESS switch central office are primarily dependent upon the software and vary from office to office. (See Fig. 16.)

4.218 The special service software can be separated into three general groups:

- Centrex-Data Link Services
- Business Communication Services
- Special Features.

B. Program Description

Centrex-Data Link Services

4.219 ***Centrex Console Lamp and Key Program (CXIO, CXKY):*** Provide all the normal (nonmaintenance) input/output functions for the centrex data link. The programs also interrogate key signals to determine their effect on the centrex console and any call on the console. These programs also investigate supervision reports.

4.220 ***Centrex Attendant Line and Trunk Seizure Program (CXLO):*** Provides control for seizing lines and trunks to establish a connection between the calling party and the centrex attendant.

4.221 ***Centrex Trunk Code-Call Answer Program (CXTA):*** Entered when digit analy-

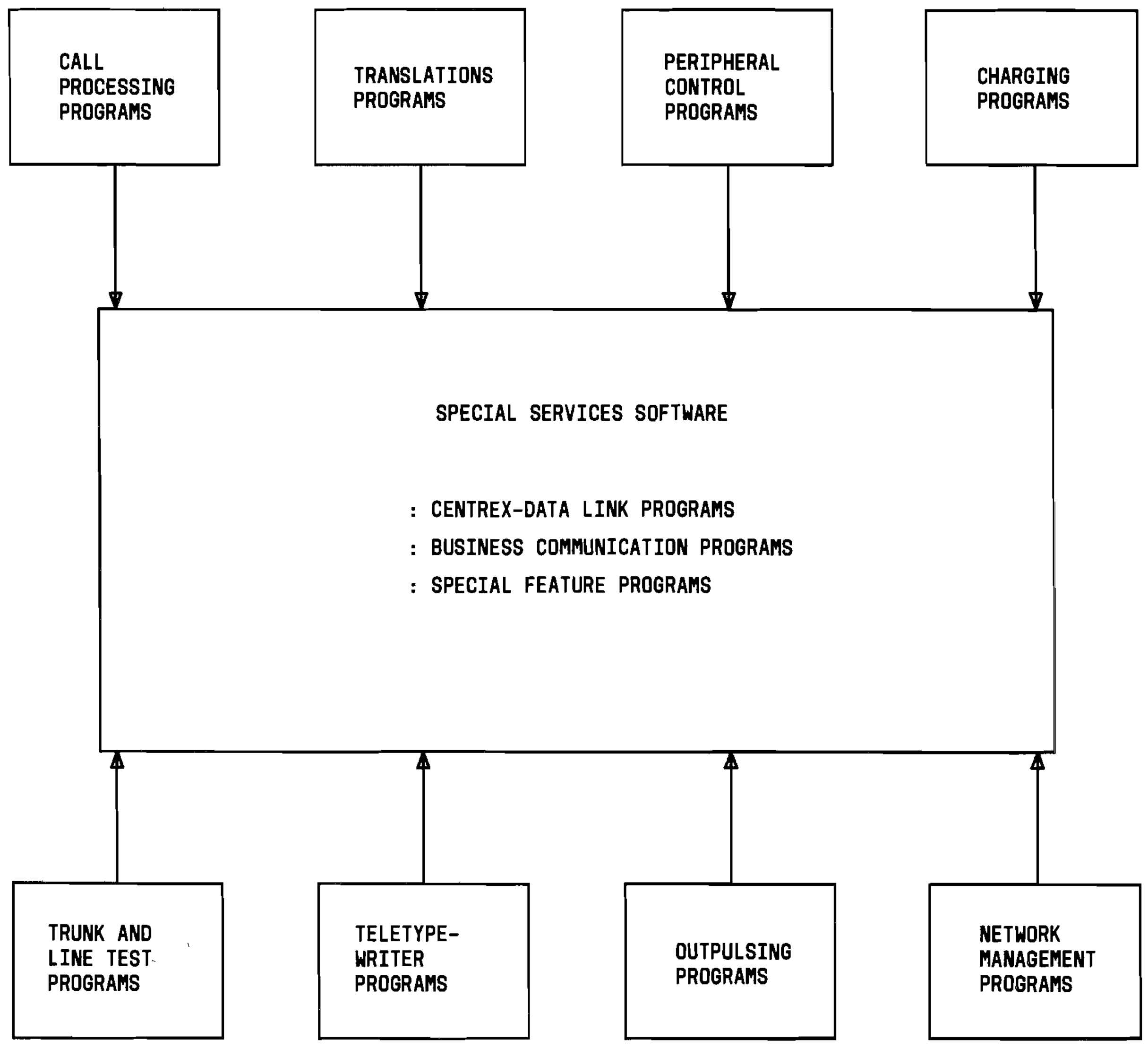

CALL PROCESSING PROGRAMS
TRANSLATIONS PROGRAMS
PERIPHERAL CONTROL PROGRAMS
CHARGING PROGRAMS
SPECIAL SERVICES SOFTWARE
: CENTREX-DATA LINK PROGRAMS
: BUSINESS COMMUNICATION PROGRAMS
: SPECIAL FEATURE PROGRAMS
TRUNK AND LINE TEST PROGRAMS
TELETYPEWRITER PROGRAMS
OUTPULSING PROGRAMS
NETWORK MANAGEMENT PROGRAMS

Fig. 16—Special Services Program Interface

sis programs recognize that a code call answer code has been dialed. The program then activates signaling devices (bells, horns, etc) to sound the called party code.

4.222 ***Centrex Console Lamp Control Program (CNLP):*** Provides an interface between the central office and the attendant console. As the state of attendant calls progress, this program transmits orders to and from the central office to administer particular console lamps and activate switching functions.

Business Communication Services

4.223 ***Centrex Simulated Facilities Program (CXSF):*** Administers usage counts and associated centrex console trunk-busy lamps after the program has determined that a call has reached the maximum allowable number of active calls that are being monitored for the specific simulated group.

4.224 ***Hotel-Motel Program (HMTL):*** Records the number of message units to be charged to a guest each time they place a call.

4.225 ***Automatic Identified Outward Dialing Program (AIOD):*** Used to make an association between a calling PBX station and the trunks being used. This information is then made available to call charging programs.

4.226 ***Busy Verify-Trunk Test Program (CXBV):*** Controls specific functions used to test the working condition of a trunk via the centrex attendant console.

4.227 ***Centrex Tandem Tie Line Program (CX1X):*** Used in conjunction with other call processing programs to give the ESS switch centrex central office the ability to act as a switching office in a tandem tie line network.

4.228 ***Centrex Trunk Preemption Program (CXTP):*** Used in conjunction with other call processing programs to complete calls over trunk groups which interface with 4-wire switching offices.

Special Features

4.229 ***Call Waiting Program (WAIT):*** Controls specific supervisory and administrative functions relating to the call waiting feature.

4.230 ***Call Forwarding Program (TXFR):*** Provides the necessary logic to complete the many path connections available in the call forwarding feature.

4.231 ***Add-On and Conference Programs (ADxx):*** Handles all events necessary to complete a call with add-on and conference features. The programs take control after normal call processing programs have recognized a request for adding a third party.

4.232 ***Customer Changeable Speed Calling Program (CCAD):*** Recognizes speed calling codes and collects and validates a change in the speed call list. Also the program associates the abbreviated code with the original telephone number.

4.233 ***Call Forwarding Usage Program (CFUP):*** Provides the necessary logic to interface with the AMA programs to provide billing information for call forwarding usage.

4.234 ***Reverting Call Program (RVRT):*** Handles all call processing duties when a customer wishes to establish a talking path to another customer with the same line equipment number. This program is used in a 4, 2 and 0 party condition.

MEASUREMENT SOFTWARE

A. Purpose of Measurement Software

Traffic Measurements

4.235 The traffic data measurement program is used to generate, accumulate, collect, and print out ESS switch traffic data. This traffic data consists of peg, usage, and overflow counts generated by call processing and maintenance programs as specific events occur, and by the traffic data measurement program as requested according to a time schedule.

4.236 Peg count is a cumulative count of the number of times a specific event occurs during a fixed time interval; eg, the number of times the system attempts to seize a customer digit receiver for dial tone in 1 hour.

4.237 Overflow is a cumulative count of the number of times an attempt to cause a specific event failed because of the lack of specific facilities (circuits or paths). Overflow counts are accumulated for specific time intervals. A count (or score) on an overflow register does not always indicate that a call has been "lost." It sometimes indicates that a call has been delayed, ie, placed in a queue and held until the desired circuit becomes available.

4.238 Usage is a cumulative count of the number of items found in a busy state during each periodic scan of a particular group of items. The busy conditions found during each scan are totaled for a fixed period (eg, 1 hour) to obtain usage.

4.239 Traffic counts are generated for the following purposes:

(a) Engineering and administration

(b) Sampling the level of service being provided

(c) Measuring the effect of maintenance routines

(d) Serving as a base for special merchandising or revenue studies

(e) Billing for specific services.

4.240 The following are descriptions of various types of traffic count output messages.

(a) H—HOURLY BUSY HOURS collect items needed during the office busy hours.

(b) C—HOURLY CONTINUOUS THROUGH DAY collection involves an hourly or half-hourly count for a continuous period of time.

(c) DA15—SELECTED QUARTER HOUR (Q SCHEDULE) provides traffic data at 15-minute intervals without destroying the validity of the hourly H and C schedules.

(d) TC15—FIXED QUARTER HOUR is a fixed collection of 15-minute traffic counts on the quarter hour.

(e) S—SPECIAL STUDIES COLLECTION is a collection of items from the H and C schedules singled out by the office for special studies.

(f) TC24A—A PORTION OF THE DAILY SCHEDULE contains a 24-hour cumulative total of 27 office counts.

Plant Measurements

4.241 The plant measurement program (PPMP) is designed to provide the management of telephone company plant departments with a quantitative summary of the condition of the central office, and the impact of this condition on customer service. The PPMP is a collection of routines to collect significant data on system performance and to produce this data on demand or on a daily and monthly basis in a readable format.

4.242 There are three plant measurement outputs. These are:

(a) PM01—daily plant measurement output message

(b) PM02—monthly plant measurement output message

(c) PM03—daily or monthly counts per TTY manual request.

B. Functional Description of Measurement Software

4.243 Measurements are accumulated in up-down counters, accumulators, holding registers, and totaling registers as discussed below.

Up-Down Counters

4.244 Up-down counters are provided for all items for which usage is measured. Each time the item is busied, one is added to its corresponding up-down counter. When the item is idled, one is subtracted.

Accumulators

4.245 Accumulators are provided for all items for which peg or overflow counts are maintained. Each time a given event occurs (or in the case of overflow, fails), one is added to the corresponding accumulator. Accumulators are also provided for those items whose usage measurement is to be added, at regular timed intervals, to a cumulative total.

Holding Registers

4.246 Holding registers are provided for all items whose counts are assigned to a collection schedule. At collection time, the program moves the contents of specified accumulators to corresponding holding registers and zeros the accumulators. The counts are subsequently printed out from the holding registers.

Totaling Registers

4.247 Totaling registers are provided for all items whose counts are measured on more than one schedule. Totaling registers receive intermediate totals for the longer measuring intervals; eg, four quarter-hour totals are added in hourly totaling registers.

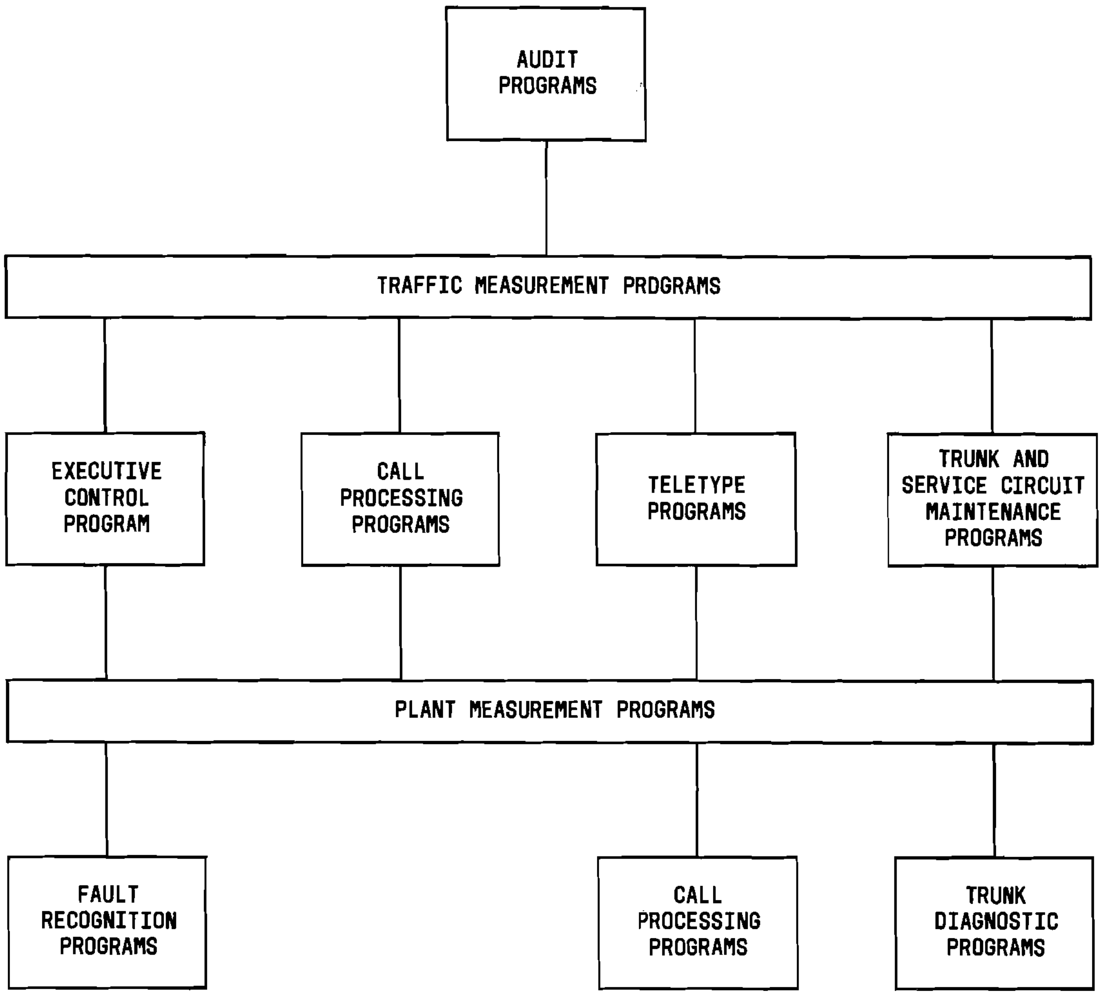
C. Interface With Other Programs

4.248 See Fig. 17 for an interface diagram of the measurement programs.

NETWORK MANAGEMENT SOFTWARE

A. General

4.249 The network management subsystem programs provide various capabilities for administering network traffic controls which are applied during traffic congestion periods for limiting the amount of traffic in congested switching facilities. The controls will permit a maximum number of calls to be completed without allowing the traffic congestion to spread throughout the network. The control capabilities include.


AUDIT PROGRAMS
TRAFFIC MEASUREMENT PRDGRAMS
EXECUTIVE CONTROL PROGRAM
CALL PROCESSING PROGRAMS
TELETYPE PROGRAMS
TRUNK AND SERVICE CIRCUIT MAINTENANCE PROGRAMS
PLANT MEASUREMENT PROGRAMS
FAULT RECOGNITION PROGRAMS
CALL PROCESSING PROGRAMS
TRUNK DIAGNOSTIC PROGRAMS

♦Fig. 17—Measurement Program Interfaces♦

(a) Code blocking

(b) Trunk group controls (TGCs) (manual and automatic via receipt of dynamic overload control [DOC] signals)

(c) Generation of DOC signals

(d) Reroutes.

B. System Network Management Control Interface

4.250 The interface between the network management programs and other system programs is depicted in Fig. 18.

4.251 The audit programs interface the network management programs to perform audits of the network management indicators and associated scan points.

4.252 The master control center programs provide the interface for displaying the status of the network management controls.

4.253 The system operational control software provides periodic entries into the network management programs for determining maintenance actions, printing controlled traffic data, and controlling specific network circuitry.

4.254 The TTY software provides the input/output interface for the craftsperson to apply manual control parameters to the network maintenance functions.

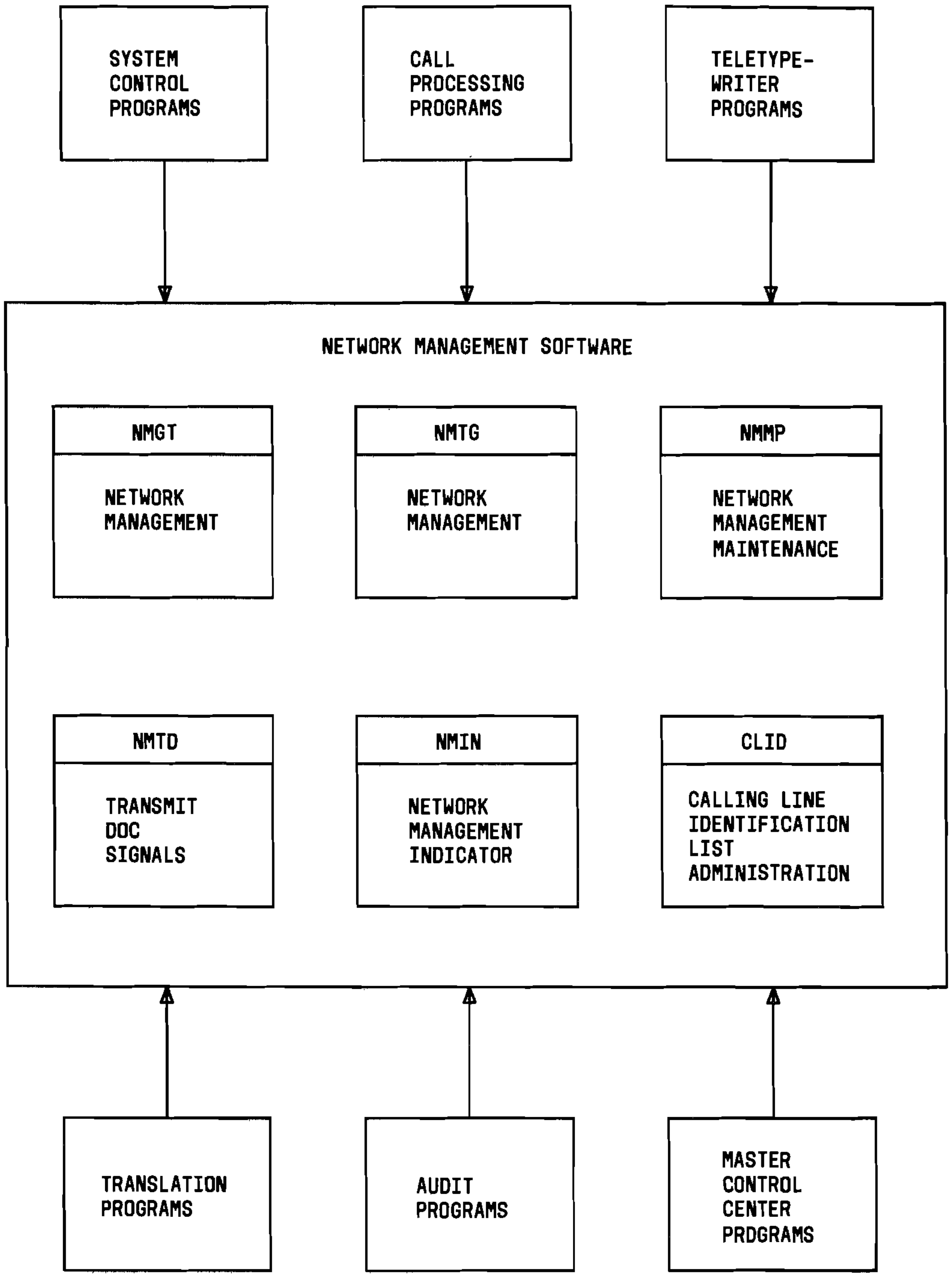


Fig. 18—Network Management Program Interface

4.255 When all the digits have been received on a call and if network management controls have been activated, the call processing programs interface with the network management programs to apply the network control to the call.

4.256 The network management programs are interfaced by the translation software whenever the trunk group control indicator signifies that the controls are activated.

C. Network Management Software Function

Network Management (NMGT) Program

4.257 The NMGT program performs the following functions:

(a) Activates code blocking controls via TTY input message

(b) Deactivates code blocking controls via TTY input message

(c) Clears code blocking entries in memory

(d) Administers OUTGOING LOAD CONTROL lamp

(e) Administers DOC signals

(f) Audits memory associated with DOC scan points

(g) Updates indicators (real-time breaks, timing, code block status, DOC data, and trunk group number data).

4.258 The DOC signals are administered every 2 seconds. Those, which are to be applied, are determined by comparing the length of the incoming overload control queue to the MC1 and MC2 values for each type of receiver and for real time.

Code Blocking

4.259 Code blocking controls are applied by the ESS switch for limiting outgoing traffic routed to congested traffic areas in order to prevent the congestion from spreading throughout the switching network. The controls are based upon the destination dialed code (area code, office code, 6-digit code, 7-digit code, 10-digit code) as determined by the office code and numbering plan area code translators. The code blocked calls are routed via one of three fixed route indexes, either to a no-circuit announcement or to one of two emergency announcements. The control for each code is applied as percentage of call attempts. The percentages are 50, 75, 87-1/2, and 100 percent. The code blocking controls are activated and deactivated via TTY input messages. When blocking controls are in effect, the call processing software transfers to the NMGT program to process the control blocking function.

Trunk Group Controls (TGCs)

4.260 The TGCs provide the capabilities to limit traffic to a congested area based on the trunk group over which a call is to be routed. There are two types of trunk group controls: preprogrammed and flexible. Preprogrammed controls are activated on prespecified trunk groups. Flexible controls can be activated on any trunk group in the office. Three control options are available to the flexible control: namely, (1) CANCEL-TO, (2) CANCEL-FROM, and (3) SKIP. The flexible control has a trunk reservation option which is used to limit the number of attempts offered to a trunk group when less than the specified number of trunks remain available. One of two thresholds, protected reservation of equipment (PRE) and directional reservation of equipment (DRE), may be applied to the trunk reservation option. The threshold, PRE, is used in reserving facilities for the first-routed traffic. If the PRE threshold is exceeded, all traffic alternate-routed to this trunk group is inhibited from searching for an idle trunk in any trunk group and is routed to the no-circuit announcement. The threshold, DRE, is used in reserving facilities for incoming traffic. If the DRE threshold is exceeded, all traffic to this trunk group is inhibited from searching for an idle trunk in any trunk group and is routed to the no-circuit announcement.

Dynamic Overload Control (DOC) Signals

4.261 Program NMGT provides the use of DOC signals which are sent from tandem and toll offices to connected ESS switch offices requesting that they limit the amount of traffic being received. The ESS switch offices use the DOC signaling feature when shortages exist in real time, MF receivers, DP receivers, revertive pulse receivers, or other call switching resources. The NMGT program processes two levels of signaling for the shortages of real time and receivers. The shortage of real time is determined indirectly from the E-E cycle time through the length of the incoming overload control queue. The shortage of receivers for each receiver type is determined by the length of the queue for that receiver type. The program checks the thresholds for real time and receiver shortage every 2 seconds. When a threshold is exceeded, the sending of DOC signals is initiated. When the shortage drops below the threshold value, the transmission of the DOC signal is stopped. The two levels of signaling for the shortage of real time and receivers are MC1 and MC2. The MC1 level indicates that the machine is sufficiently congested to cause substantial delays in receiver attachment. The MC2 level indicates that the machine is considerably more congested than MC1 level. The MC2 level indicates delays of 40 to 80 percent of the receiver holding time. The MC3 signal is sent when the switching facility is incapable of processing calls. The command source for the MC3 signal is derived from either of two lamp signals on the MCC. They are: (1) emergency action phase in progress and (2) repeated time-out. The transmitted DOC signal is an ON signal interrupted by an OFF signal every 30 seconds. The interruption is provided by a duplicated hardware interrupter. The interrupter is monitored by network management maintenance program (NMMP) which switches the interrupters, sounds a minor alarm, and prints a TTY output message when a fault occurs.

♦Network Reroute Control (NMRR) Program

4.262 The NMRR program provides network managers the capability to reroute traffic away from congested or trouble switching facilities to other facilities having sufficient switching capacities. The rerouting controls may be applied to both local and tandem calls. An audit of the control data slots in memory is performed and errors are printed via the TTY. The program provides for manual activation and deactivation of the control. Options to the reroute control are trunk hunting, percentage of blocking, type of traffic, and "to trunk group" selection.♦

Network Management Indicator (NMIN) Program

4.263 The NMIN program performs two major functions:

(a) Updates network management indicators (machine status, network status)

(b) Provides listing of trunk group no-circuit data.

Status Indicators

4.264 A visual display indicating the current machine and network status is activated via signal distributor points by program NMIN. Every 10 seconds, an entry is made to NMIN from the main program to update the indicators. The display indicates the status of:

- Transmitters (MF, trunk DP, revertive pulse)
- Receivers (TOUCH-TONE service customer DP, trunk DP, trunk revertive pulse, and trunk MF)
- Line load control
- Incoming load control
- Incoming overload queue
- Matching loss
- Machine congestion
- Internal queues.

Network Management Maintenance (NMMP) Program

4.265 The primary purpose of the NMMP program is to perform maintenance on the DOC transmit function and applique circuitry.

Transmit Dynamic Overload Control Signals (NMTD) Program

4.266 The NMTD program administers the transmission of DOC signals which are sent from a tandem or toll office to connected local offices requesting that they limit the amount of traffic in the network.

Network Management (NMTG) Program

4.267 The NMTG program provides the capabilities to apply controls which limit the amount of traffic leaving an office based on the trunk group over which a call is to be routed.

4.268 This program processes four types of trunk group controls:

(a) CANCEL-TO (controls the number of call attempts offered to a trunk group)

(b) CANCEL-FROM (controls the number of call attempts overflowing a trunk group)

(c) SKIP (controls the number of call attempts offered to a trunk group)

(d) Trunk reservation (allows the selection of a specified number of trunks in a trunk group).

Calling Line Identification List Administration (CLID) Program

4.269 The function of the CLID program is to provide the interface to the TTY programs for maintaining the code blocking controls in the system memory, to provide a listing of the controls upon request, and to provide trace information on calls to selected directory numbers.

FILE STORE ADMINISTRATION SOFTWARE

A. General

4.270 The file store is a bulk storage device that provides the 1A processor with disk memory which supports call store and program store. These programs are paged into program store from file store when needed. File store also contains back-up copies of unduplicated call stores.

B. Functional Description

4.271 Many programs require a rapid transfer of data to or from file store/APS. Duplication of routines in all programs desiring file store/APS services is avoided by providing a single program to process all requests for data transfer. The file store administration program provides an interface between the file store and client programs desiring a transfer of data. This interface, as well as others, is shown in Fig. 19.

4.272 The file store administration program is functionally divided into three parts. They are (1) the file store administration submit program, (2) file store administration answer dispenser program, and (3) file store service routines.

File Store Administration Submit Program (FSSP)

4.273 A client program directly enters FSSP to request a file store service. Initially, a client submits either a READ request (a request to transfer data from disk to main memory) or a WRITE request (a request to transfer data from main memory to disk). In order to avoid erroneous reads or writes and unnecessary fault interruptions, FSSP performs preliminary software checks on client input. If these checks pass, FSSP activates the file store hardware necessary for the file store controller to process the request. Requests may be queued in call store when a request cannot be immediately processed. The FSSP notifies the client via an immediate success return when its request is activated (entered in buffer register or placed on queue) or via one of two other immediate return addresses if circumstances prohibit the request's activation. After FSSP returns to the client, the client can continue other processing while the file store controller performs the file store service requested.

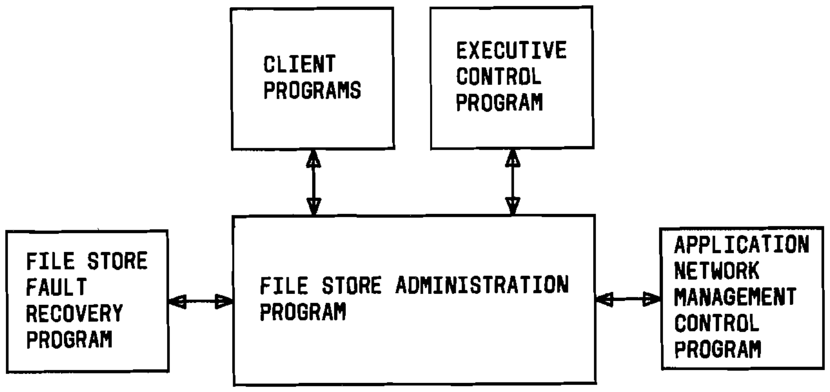

CLIENT PROGRAMS
EXECUTIVE CONTROL PROGRAM
FILE STORE FAULT RECOVERY PROGRAM
FILE STORE ADMINISTRATION PROGRAM
APPLICATION NETWORK MANAGEMENT CONTROL PROGRAM

Fig. 19—File Store Administration Program Interfaces

File Store Administration Answer Dispenser Program (FSAP)

4.274 The executive control main program routinely enters FSAP on base level to:

(a) Detect completed requests, judge the status of the request, and give the client an appropriate completion return

(b) Unload queued requests.

4.275 A file store service is completed only after both file store hardware and FSAP software actions are accomplished. The hardware action is complete when the applicable file store controller actions terminate. The software action is complete after FSAP examines the request's status. If error conditions exist, FSAP interfaces the file store fault recovery program to resolve the error.

4.276 When both hardware and software actions are complete, FSAP notifies the client via the appropriate success or fail completion return address. When a failure return is made, the client is given a return code for one of the following types of failures:

- Software error
- Hardware failure
- Maintenance identification tag mismatch
- Failure of a job retried
- Cancellation of job.

File Store Service Routines (FSSR)

4.277 The FSSR is a separate entity of the file store administration program and consists of several routines that are available for use by other programs. In general these services enable a client to:

(a) Check the status of a file store and a disk file; ie, file store controller and disk file in service or out of service.

(b) Translate a main memory address to its corresponding disk address, if the data is backed up on disk.

(c) Verify the consistency of an identification tag and a disk address.

♦ATTACHED PROCESSOR SYSTEM

A. General

4.278 The Attached Processor System (APS) is used to replace the 1A processor disk files. The APS (Fig. 20) consists of a duplex 3B processor utilizing a 300-megabyte disk system and an Attached Processor Interface (API).

4.279 The API supports the attached processor communications link (APCL), protocol between the 1A and 3B Processors. The APCL protocol has both efficient block transfer and message-handling capabilities. This is accomplished by having the API interface with direct memory access (DMA) facilities of both processors.

4.280 The APCL protocol also includes a high-priority maintenance message communication capability which is supported by the API. These messages are communicated in a closely coupled, synchronous, high-priority way by using the 3B input/output interrupt and the 1A auxiliary unit bus maintenance interject mechanisms.

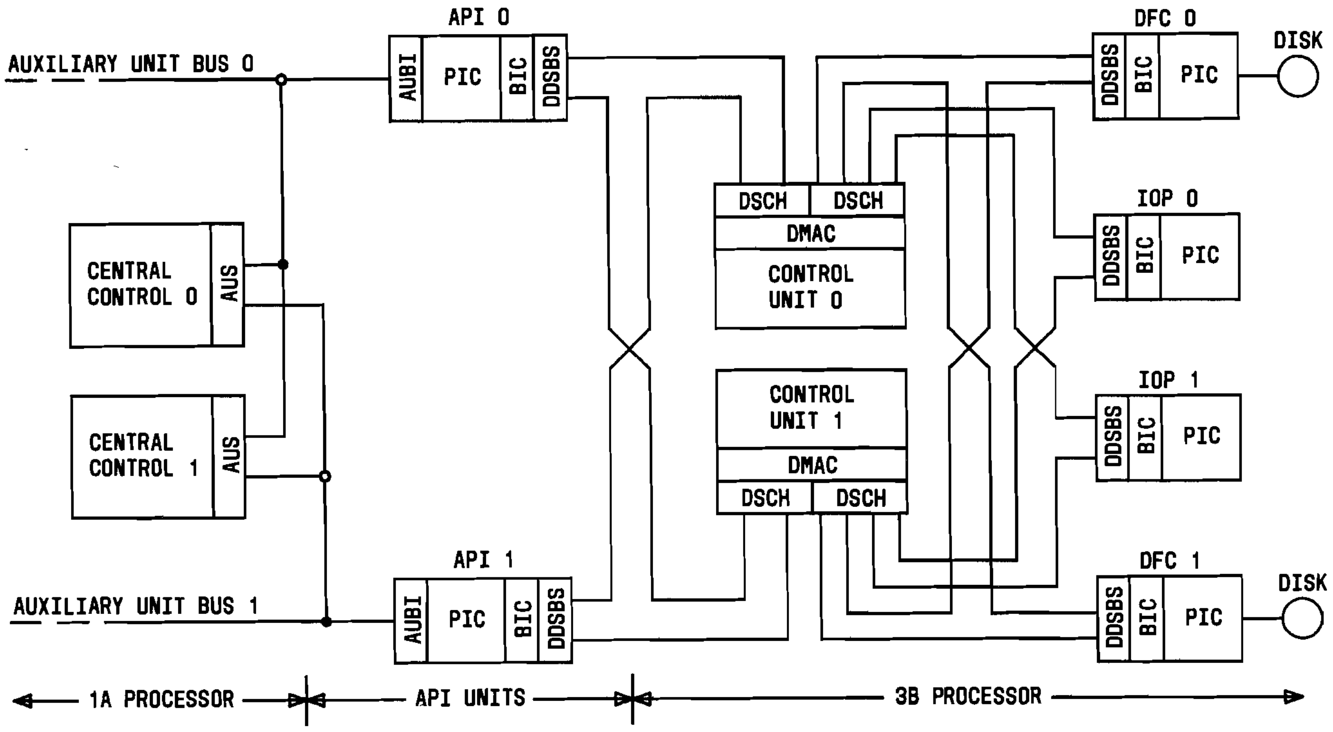
B. Attached Processor Message Handler

4.281 At the heart of the APS common software system is the attached processor message handler (APMH) program on the 1A and 3B sides. All interprocessor activity except the command system (message or DMA control) passes through the APMH. The APMH directs asynchronous-queued messages to either its equivalent APMH process on the other processor through the API or to local clients. The APMHs, therefore, impose common message formats also to be interpreted by the API. The APMHs also use a DMA job control buffer to administer the necessary DMA control information. Messages are loaded and unloaded from a pair of circular message buffers.

C. File Manager Interface (FMI)

4.282 The 3B disk cannot DMA directly into the 1A memory, nor is it compatible with the 1A file store access modes. The FMI resolves the 1A and 3B differences by providing these functions:

- 3B disk access of 1A main memory
- Sending 1A messages to Duplex Multi-Environment Real-Time Operating System (DMERT)



LEGEND:

API	- ATTACHED PROCESSOR INTERFACE
AUBI	- AUXILIARY UNIT BUS INTERFACE
AUS	- AUXILIARY UNIT SEQUENCER
BIC	- BUS INTERFACE CONTROLLER
DDSBS	- DUPLEX DUAL SERIAL BUS SELECTOR
DFC	- DISK FILE CONTROLLER
DMAC	- DIRECT MEMORY ACCESS CONTROLLER
DSCH	- DUAL SERIAL CHANNEL
IOP	- INPUT/OUTPUT PROCESSOR
PIC	- PERIPHERAL INTERFACE CONTROLLER

Fig. 20—Attached Processor System Hardware Block Diagram

- Changing 1A disk word flexibility to 512 byte format
- Issuing an OPEN command to DMERT/File Manager (FM) at bootstrap time
- Maintaining "normal" and "update" copies during update procedures.

D. Disk Administration Interface (DKADI) Program

4.283 The DKADI on the 1A side uses the APCL protocol. The purpose of DKADI is to convert the information provided by the disk administration program (DKAD) macros into the equivalent DMERT-like FM calls acceptable to the APS. For example, a Read_Either macro call is converted into the appropriate FM_READ for the FMI and supplementary information unique to the DKAD macros is saved.

1A PROCESSOR INPUT/OUTPUT SOFTWARE

A. General

4.284 The 1A processor input/output software provides an interface between hardware, used by the processor to communicate with telephone personnel, and the client programs. There are many programs which request some kind of input/output service. Request for input/output service may be automatically initiated or called upon manually from the MCC. The input/output software has no knowledge of the request source and there is no fixed association of programs and input/output channels. Many programs may be simultaneously directing output to the same channel. The input/output main control program must interface with several other programs in order to give complete service to any request. This is shown in Fig. 21.

B. Input Message Software

4.285 The input/output handler is run on base level and is scheduled at least every 60 ms by the application executive control program. The input/output handler polls the hardware looking for a service request. Polling is performed by sending a gated control pulse to all input/output unit selectors (IOUSs) via the peripheral control bus. Each IOUS community has a dedicated reply bus bit position to send a reply. The bit position represents the IOUS member number times two plus input/output community. A zero reply indicates some input/output unit controller (IOUC) in the IOUS has work. A read of the community poll request register indicates which IOUC. If the poll request register is zero, it indicates a maintenance problem and control is passed to the process F-level recovery program for resolution. Otherwise, the reply from the IOUS indicates whether there is one or more IOUCs with a service request and/or maintenance request in the request register waiting to be read. The input/output handler recognizes five types of service requests:

- Input (start of input) request
- Input (unload buffer) request
- Output (load buffer) request
- Idle request
- Break request.

4.286 Each input/output channel has associated with it a character buffer in the hardware.

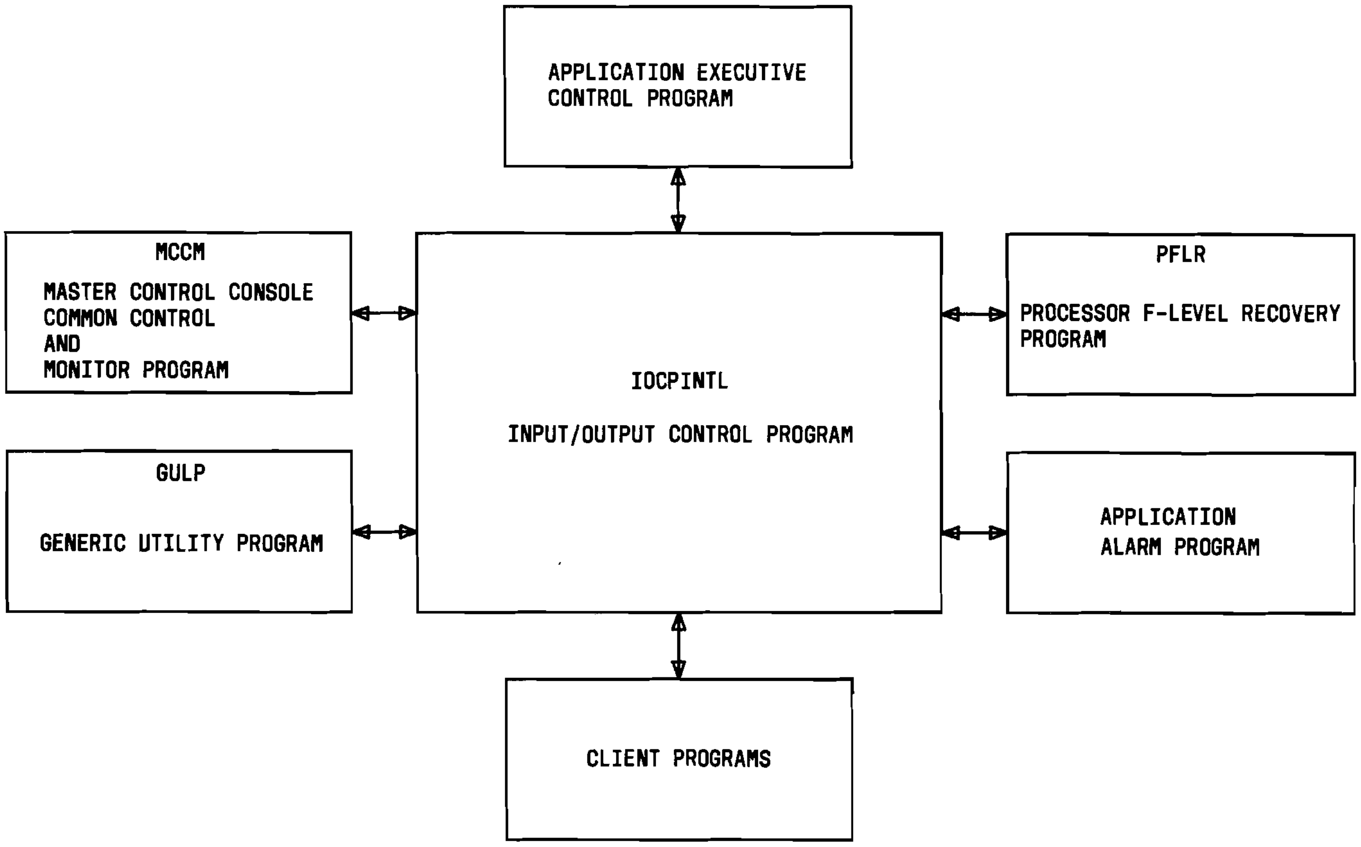

APPLICATION EXECUTIVE CONTROL PROGRAM
MCCM
MASTER CONTROL CONSOLE COMMON CONTROL AND MONITOR PROGRAM
IOCPINTL
INPUT/OUTPUT CONTROL PROGRAM
PFLR
PROCESSOR F-LEVEL RECOVERY PROGRAM
GULP
GENERIC UTILITY PROGRAM
APPLICATION ALARM PROGRAM
CLIENT PROGRAMS

Fig. 21—Input/Output Control Program Interfaces

(The buffer is physically located in the input/output unit.) When an input message comes from the terminal to the buffer, the first character changes the hardware state from idle to input and brings up a service request. When the input/output handler detects a channel with a service request with the hardware state marked as input and the software state marked as idle, it realizes that an input message is beginning. It then changes the software state from idle to input so that the input message will not be interrupted by output messages for that channel. The input/output handler also puts the channel on 2-minute timing to ensure that the channel is not kept in the input state when there is no input being entered.

4.287 The input/output handler stores the characters in a 32-word input character buffer (ICB) contained in the channel memory block (CMB). A CMB is a 44-word block of memory used to hold information about messages going to and coming from a channel. Each channel has a CMB. The 32-word ICB can accommodate input messages of up to 96 characters; however, input messages are limited to a single line on the input device.

4.288 The input/output handler fetches the characters from the hardware buffer and stores them in the ICB until the hardware recognizes the end-of-input message character and brings up a service request. The input/output handler realizes that the hardware has recognized the end-of-input message character and stores the final characters in the ICB. It then sets an input translation request flag for this channel. There is an input translation request flag associated with each channel which is scanned by the input translator. The input/output handler puts the channel on 5-second timing to wait for an acknowledgment. The client program which handles the message is responsible for generating the acknowledgment.

4.289 The input translator is entered during every base level cycle from the application executive control program. It scans the input translation request flags looking for a channel that needs work. When the input translator finds the input translation request flag set for a normal message, it transfers to a translation subroutine which parses the message in the ICB.

4.290 If the message is recognized by the input translator as a valid input message, the input translator transfers to the client. The client program is passed the address of the input message data area which contains a message and the channel number on which the message was entered. The input message data area contains flags indicating what keywords in the message were entered. The client program then has 3 ms to gather the data about the message, determine whether this is a valid message, and generate an acknowledgment. The client has up to 5 seconds to generate an acknowledgment. Once the client program returns to the input translator, the input translator is free to look for other input messages to translate on its next entry. The input translator only handles one client per entry.

C. Output Message Software

4.291 Output messages may be the result of an input message, an MCC request, or automatically generated by the system. Output data tables (catalogs) are used to convert formats which are easily read by operating personnel.

4.292 To request that an output be sent to a channel, the client program calls the print call handler. In the print call handler, the client program specifies the catalog for the output message and the raw data for the output. The client may also specify the priority of the output message and from one to five channels or message classes. The print call handler has three returns to the client:

(a) Success: The output message is valid and buffered for output.

(b) Fail: The output message is valid but there are insufficient resources to buffer the message.

(c) Checksfail: The request is invalid.

4.293 If the required number of output message registers (OMRs) are available, the print call handler calls a subroutine to move the client's raw data into the OMRs. The print call handler then puts the output message into the output translate list and returns success to the client.

4.294 The output translator is entered during every base level cycle. It scans the translate output link list for the address of a control OMR which needs work. Once it finds work, an initial action is to transfer to an alarm program to determine if an audible alarm is appropriate, and if so, start the alarm.

4.295 From the output message catalog, the output translator gets fixed text for the message and information to convert the raw data passed by the client to a string of characters. It then loads these characters into the OMRs that were seized by the print call handler. When translation is completed, all OMRs which were reserved and not used are idled; and, the translated output message is put on the output link list. After the output translator puts the output message on the output link list, it sets a demand flag to request the execution of the output message starter.

4.296 The output message starter checks each one of the specified channels to see if it is idle. When the starter finds an idle destination, the starter places the channel in the output state, sends the first buffer of characters of the message to the hardware, and changes the software state from idle to output.

AUXILIARY DATA SYSTEM SOFTWARE

A. General

4.297 The ADS operational programs are common to systems using the 1A processor. These programs handle the transfer of data between 1A processor memory and ADS magnetic tape storage. Examples of when transfers are made are during:

- Accounting procedures
- Program updating
- System memory dumps
- System reinitialization
- General data handling.

4.298 The ADS is a versatile medium speed, data handling system utilizing magnetic tape storage. The capability exists for adding data links and other data services as these devices become available in the future.

4.299 The ADS is comprised of data unit selectors, data unit controllers, and data units. An ADS community consists of two data unit selectors, up to 16 data unit controllers, and the associated data units. The ADS may be equipped with two communities.

B. ADS Organization and Operation

4.300 Figure 22 shows the organization of the ADS and its interface with central control. The data unit selectors are provided in pairs and are located on the auxiliary unit bus. Each data unit selector is permanently associated with a particular auxiliary unit bus. Data unit selector 0 is connected to auxiliary unit bus 0 and data unit selector 1 is connected to auxiliary unit bus 1. Data unit selector-to-data unit controller interface is accomplished with duplicated internal buses called data unit bus 0 and data unit bus 1. Each data unit bus is permanently assigned to a data unit selector. However, each data unit selector in the pair is capable of communication with any of the data unit controllers in the community.

4.301 Data transfer functions within the ADS are a composite of hardware, software, and manual functions. These functions include reserving the required units, mounting and demounting of tapes, reserving and initializing tables, etc.

4.302 The data unit controllers are control units which are capable of accessing program store and call store memory with minimal interference from central control actions. After a data unit controller has been properly initiated, a data transfer from program store or call store to magnetic tape or from magnetic tape to call store or program store is completed automatically. Upon conclusion of the data transfer, the data unit controller sends a signal to central control to notify the system software of job completion.

4.303 Programs that use the ADS storage capability are known as client programs. These programs have a wide variety of requirements. Each has its own unique data organization, frequency of use, and data integrity requirements.

4.304 The data unit administration program (DUAD) is the main program which interfaces between these client programs and the hardware subsystem. The DUAD program controls the data transfer functions as required by the client programs. It is responsible for assembling the necessary resources, both physical units and logic, to perform a given data function. Other responsibilities include checking the validity of a request, initializing the ADS hardware, and verifying completion and returning request status to the client.

4.305 The DUAD interfaces with other programs in order to offer a full service to the client programs. This interface is illustrated in Fig. 23.

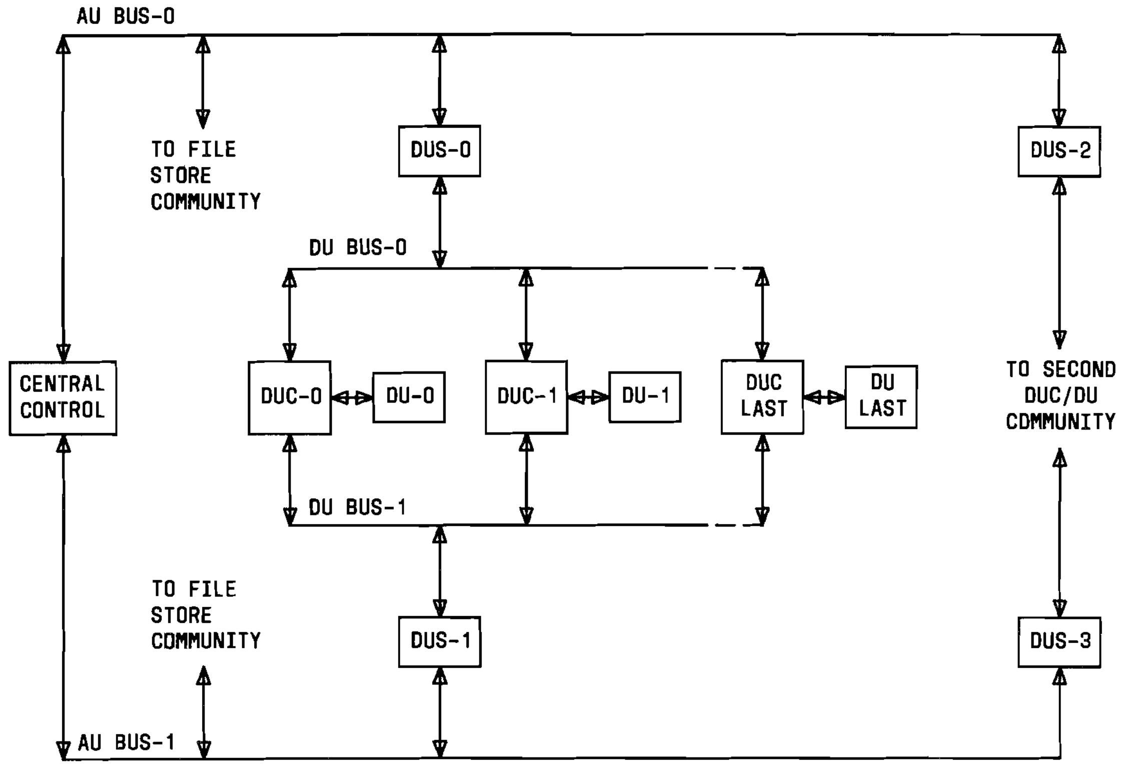


Fig. 22—Auxiliary Data System Interface Diagram

➧PERIPHERAL UNIT CONTROLLER

A. General

4.306 The peripheral unit controller is a general purpose microprocessor based system which controls the digital carrier trunk (DCT) and data link facilities.

4.307 The PUC, Fig. 24, consists of two duplicated controllers, each consisting of a hardcore, memory, ESS switch-PUC interface, and a PUC-peripheral interface. The controllers are normally operated in the duplex mode, synchronized by means of a clock. They are initially brought into synchronization by a firmware routine coordinated by the two controllers.

4.308 Each hardcore contains two microprocessors, a read only memory (ROM), a random access memory (RAM) which is a write-read memory, and a maintenance circuit which matches the operations of the two microprocessor complexes of the hardcore (Fig. 25).

4.309 The status and control of the PUC is maintained via biplar central pulse distributor points, unipolar central pulse distributor points, and ferrods.

B. Peripheral Unit Controller (PUC) Software

4.310 The PUC software subsystem programs provide the capability for the use of a PUC to control the switching process between the central control and the DCT and data link transmission facilities. Figure 26 shows a block diagram depicting the PUC subsystem functional interface between the ESS switch application programs and the PUC.

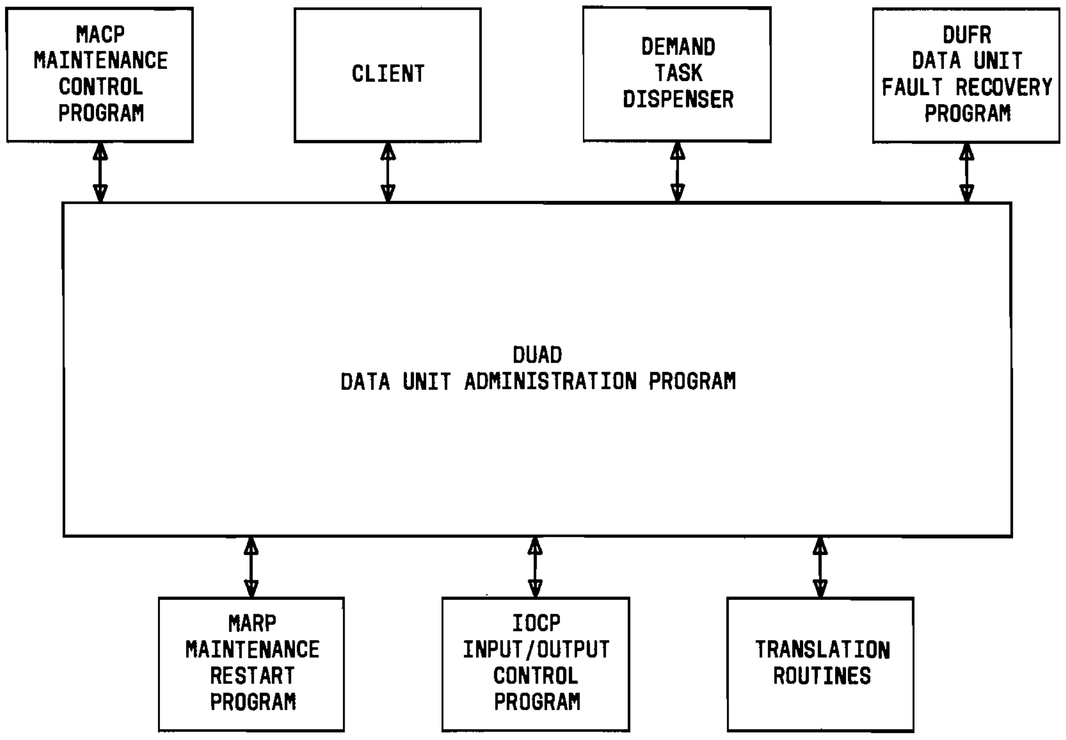


Fig. 23—Data Unit Administration Program Interface

PUC-ESS Switch Communication

4.311 The central control communicates with the PUC via orders. The PUC communicates with the central control via messages loaded in a dedicated area of the PUC memory called the message buffer. The message buffer appears to the central control as a scanner.

4.312 Each PUC contains a duplicated message buffer. Both of the message buffers are normally operated in the duplex mode (both buffers operable, one active and the other on standby.).

4.313 Data is transmitted from the central control to the PUC over the peripheral unit address bus (PUAB) in data blocks (messages). All data sent to the PUC passes through the first in/first out (FIFO) buffer stack.

Buffer Unload

4.314 Message buffers are unloaded during the class C level job schedule. Each time a PUC message buffer is unloaded, a finished unload order is sent to the PUC from the base level program. This order informs the PUC to update the control word that is used to prevent the unloading program from unloading a message the second time. The message buffer, associated with the master controller, is unloaded in the duplex mode. The inactive buffer is unloaded, when requested by maintenance, in the simplex mode.

C. PUC Maintenance

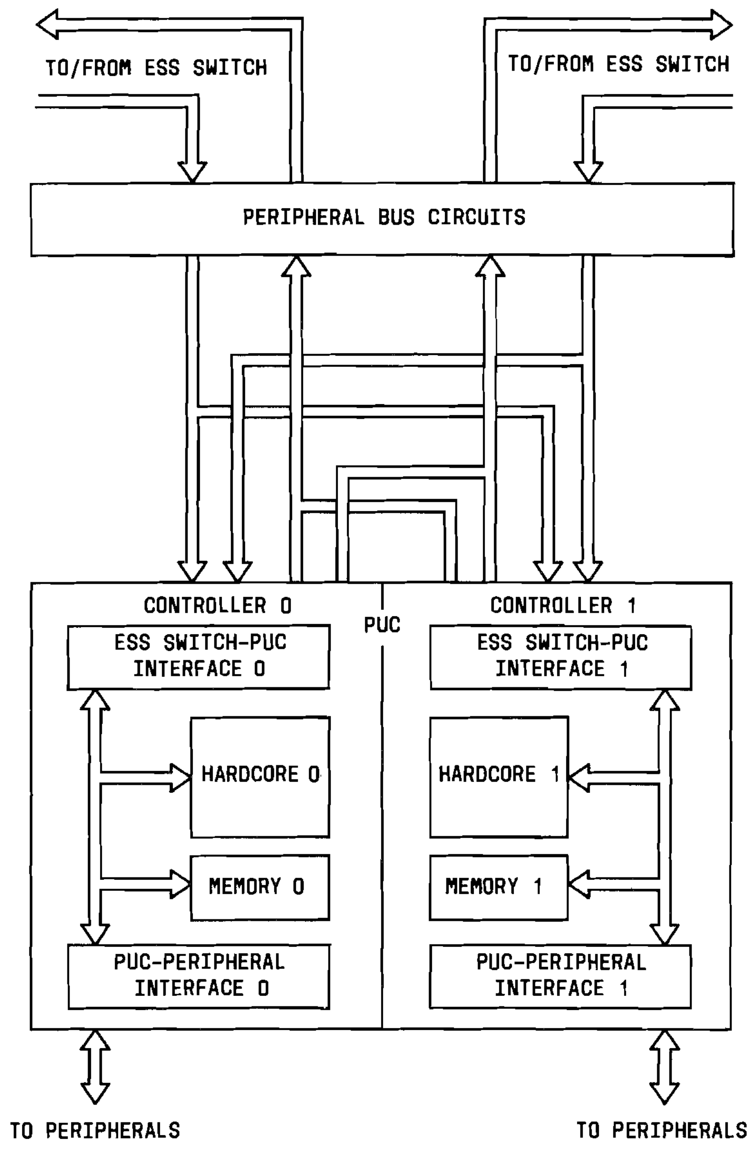
4.315 Software maintenance is based on:

(a) The ESS switch has ultimate control of all PUC maintenance states.

(b) The PUC provides the first level of fault recovery and all internal diagnostic routines.

(c) The ESS switch controls diagnostics of the PUC-ESS switch interface circuitry.

(d) The SCAB contains parity.



♦ **Fig. 24—PUC Block Diagram** ♦

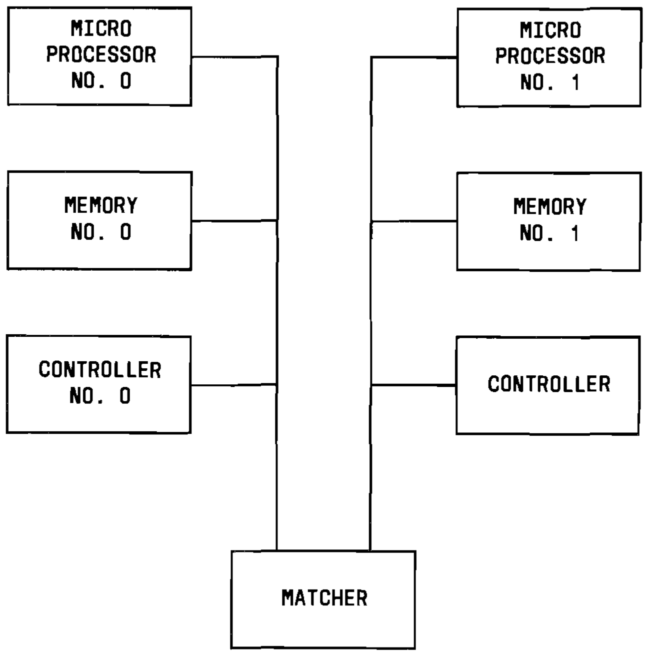
PUC/DATA LINK (DL)

A. General

4.316 The PUC is a self-checking microprocessor controller. The PUC/DL is a particular application of this general purpose microprocessor-based controller. It is designed to serve as a general purpose data link controller for a number of projects requiring a DL from ESS switch central offices. The PUC/DL provides an interface between a No. 1/1A ESS switch central office and up to 16 bidirectional data links. Provisions are made for connecting multiple links to the same destination giving protection from link failures. The redundant link in use can be changed during communication without loss of data. The PUC/DL applications are as follows:

(a) Remote Switching System (RSS) feature (Fig. 27)

(b) Electronic Tandem Switch (ETS) feature (Fig. 28)



♦ Fig. 25—Block Diagram of Hardcore ♦

(c) Local and Toll CCIS feature (Fig. 29).

4.317 The RSS feature serves as a switching entity primarily in the 150 through 1500 line size. The PUC/DL serves as the interface between the data link control function for the RSS remote terminal and the host ESS switch central control. The RSS has applications such as: a replacement for small and slowly growing community dial offices, a new wire center vehicle, and a large pair gain system. The RSS is remotely controlled from a host ESS switch. The RSS remote terminal performs as an extension of the host system and receives command information over a dedicated data link. A single host ESS switch is capable of controlling up to 31 RSSs which may be as far as 75 through 175 miles away. In this application the PUC/DL performs the following functions:

(a) Receives data from the host ESS switch, buffers and formats the data into the appropriate data link message protocol, and then transmits it serially to the data link

(b) Responds to data link reconfiguration request from the host central control

(c) Receives serial data from the RSS, buffers it, and signals the host central control so that the information received can be read by the host system

(d) Detects its own PUC faults and reports to the host central control

(e) Performs diagnostic tests when requested by the host central control or on a timed basis

(f) Performs audits of its own memory when requested by the host central control

(g) Reinitializes itself when requested by the host central control.

4.318 The ETS feature provides ETS customers with limited access and control of customer related ESS switch stored data. Customer interface mediums to this ESS switch data include a Customer Administration and Control System (CACS) or a Local Customer Administration System Terminal (LCAT) which is simply a data terminal. Both interface to the ESS switch by a 300 bps data link terminating in the central office at the PUC/DL. The PUC is connected to the ESS switch via the peripheral unit bus communication structure.

4.319 The Local/Toll CCIS feature provides for exchanging information between processor-equipped switching systems over a network of signaling data links between offices in both the local and toll network. All signaling data, including the supervisory and address signals necessary to control call setup and take down, special service related signals, as well as network management signals, are exchanged by these systems over the signaling links. These signaling links may be provided for the CCIS feature by the PUC/DL.

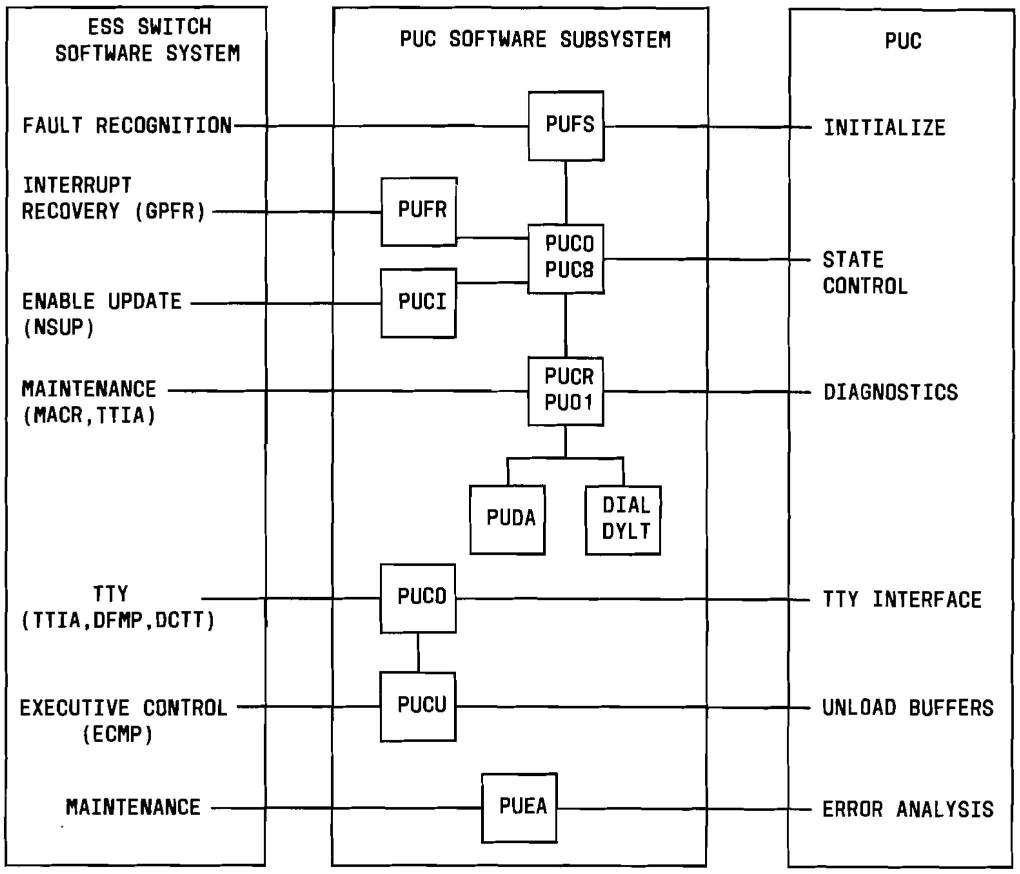
B. Software Interface

PUC/DL Input Data Flow

4.320 The incoming data to the PUC/DL from central control is transmitted over the PUAB and the FIFO buffer. This buffer contains 256 words with each word containing 16 bits of data.

4.321 The transferring of information between the PUC/DL and the ESS switch is accomplished by the PUC/DL application firmware.

4.322 The input data messages are loaded into buffers in the PUC scanner answer memory



♦Fig. 26—PUC Software Subsystem Functional Interface♦

(SCAM). A total of four out of eight SCAMs having 64 words each are provided for storing the input data. (See Fig. 30).

4.323 The PUC/DL firmware separates each input word into three data fields (high order control plus two data fields). The high order control field is checked for parity. If the parity is incorrect, an error report is returned to the ESS switch. The control field also contains indicators for priority, application, maintenance, and the message type that identifies the order.

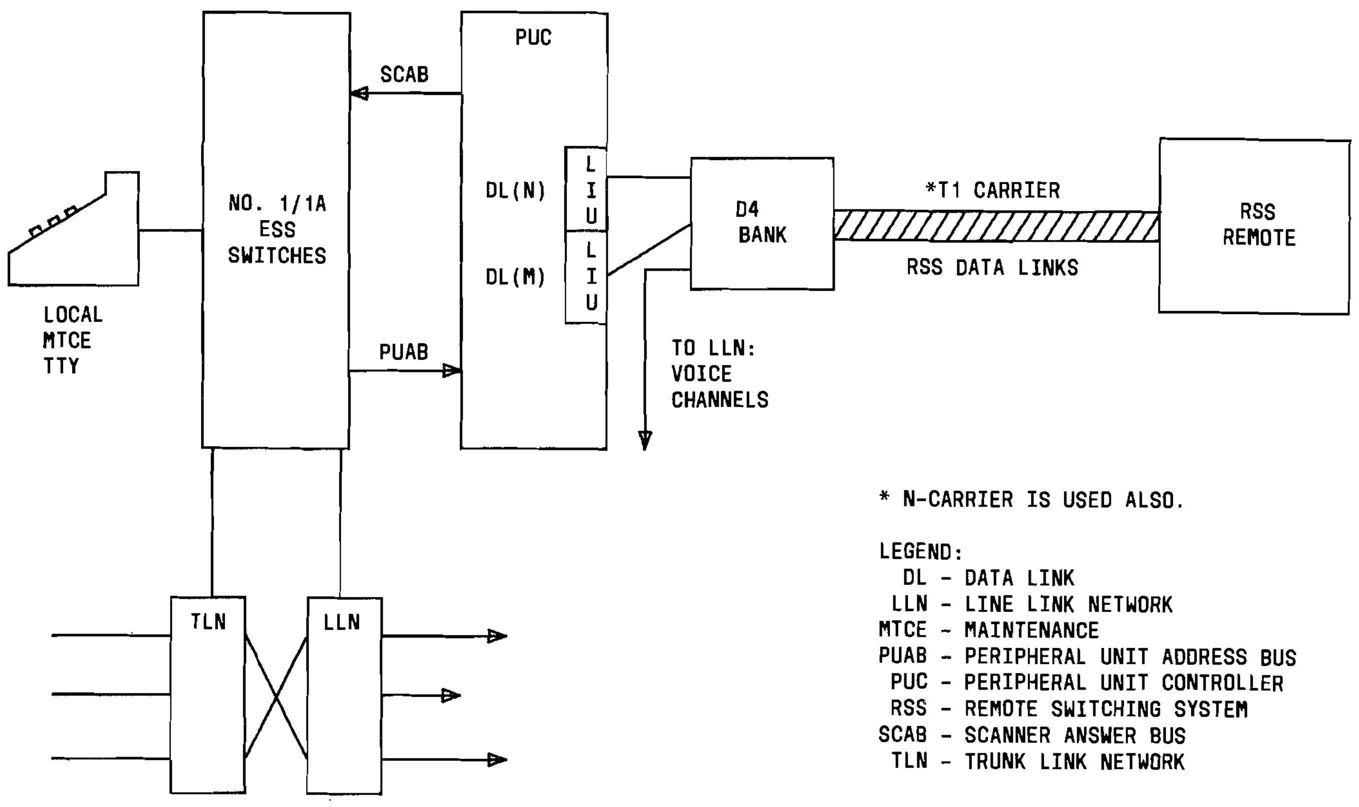
4.324 The incoming message words from the ESS switch are separated into three categories:

- Data
- Parameter
- Maintenance.

Data Messages

4.325 The data message (Fig. 31) is marked with a destination buffer number specifying a buffer in the PUC/DL read/write memory. The data messages begin with a heading word. The middle field of the heading word contains the destination number. This specifies the transmit buffer where the words which follow the heading word will be stored. The heading word, after being interpreted by the PUC/DL, is discarded. The low-order message field of the heading word contains a word count indicating the number of words to be written in the FIFO buffer. The data words following the heading word are identified by all data bits zero in the high order field.

4.326 When each new data word is stored in the destination buffer (Fig. 32), it is added to the end of the list. If the buffer is full, the present message is discarded and an error is reported. Cumulative counts are kept of all words unloaded from the



♦Fig. 27—PUC/DL RSS T1 Carrier Office Configuration♦

FIFO buffer and the destination buffer. These counts are recorded in the SCAM where they are read and used by the ESS switch to control the input rates and prevent overflow during normal operation.

Parameter Messages

4.327 The parameter messages are used to define the operation of the PUC/DL. Most parameters are sent when the unit is brought on-line. The receipt of a parameter causes a table entry to be initialized or changed and an audit to be generated. Its purpose is to change the internal data structures to agree with the new parameter.

4.328 The parameter messages are also used to specify the number and length of destination buffers; the number, length, and location of data output buffers in the SCAM; and the destination and protocol type of each data link.

Maintenance Messages

4.329 The maintenance messages are processed by the controller maintenance programs. They are used for bringing a data link on-line, clearing an error count, and switching data links.

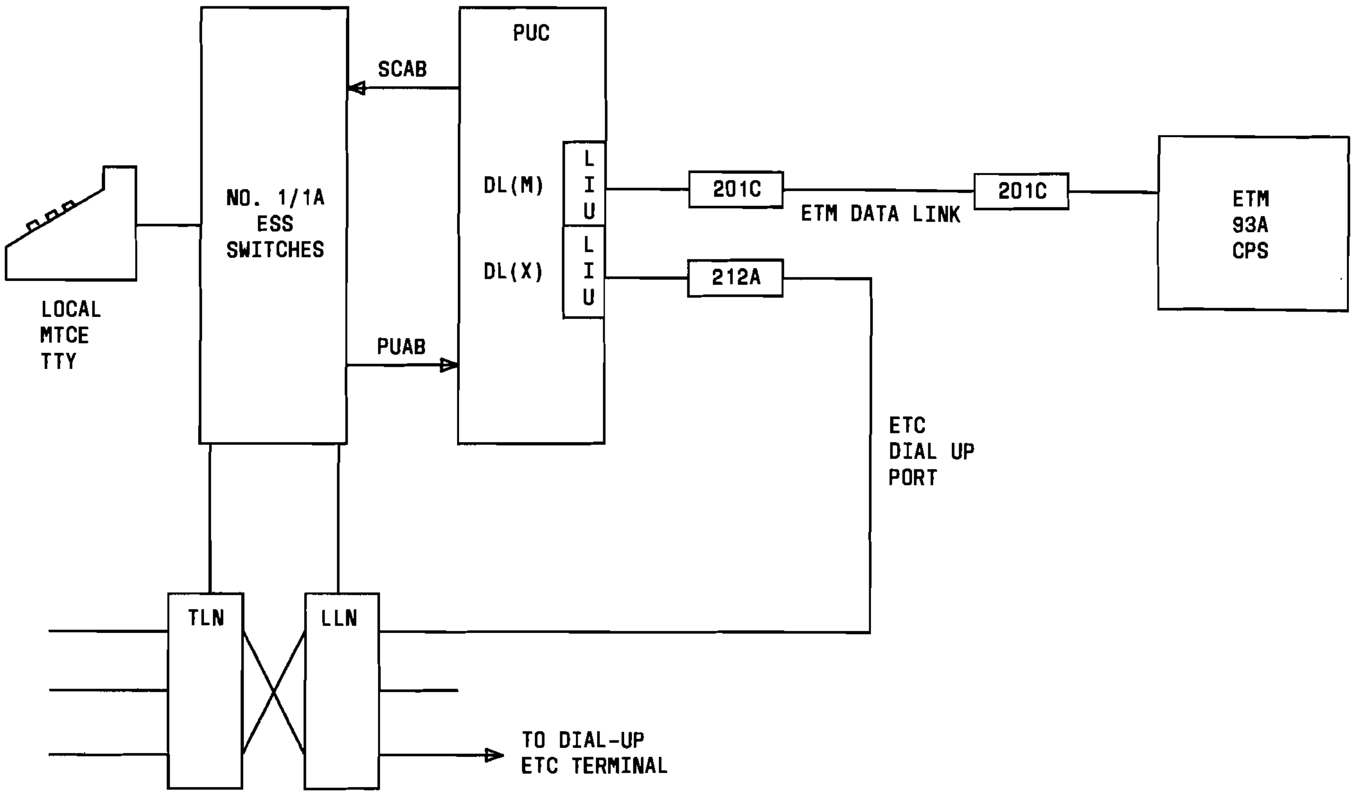
PUC/DL Output Data Flow

4.330 All output data and control information from the PUC/DL to the ESS switch passes via the PUC SCAM. The data is divided into three types:

- Incoming data link messages
- Error and status reports
- Special indicators.

Incoming Data From Data Links

4.331 The incoming data from the data links is deposited in the circular buffers located in the



LEGEND:

CPS	-	CUSTOMER PREMISES SYSTEM
DL	-	DATA LINK
ETC	-	ETS FOR CUSTOMER ADMINISTRATION CENTER SYSTEM
ETM	-	ETS FOR MESSAGE DETAIL RECORDING
LIU	-	LINE INTERFACE UNIT
LLN	-	LINE LINK NETWORK
MTCE	-	MAINTENANCE
PUAB	-	PERIPHERAL UNIT ADDRESS BUS
SCAB	-	SCANNER ANSWER BUS
TLN	-	TRUNK LINK NETWORK

♦ **Fig. 28—PUC/DL ETS Office Configuration** ♦

SCAM. The SCAM load pointer is used for loading the data in the SCAM. The unload pointer used for unloading the buffers is located within the ESS switch.

Error and Status Reports

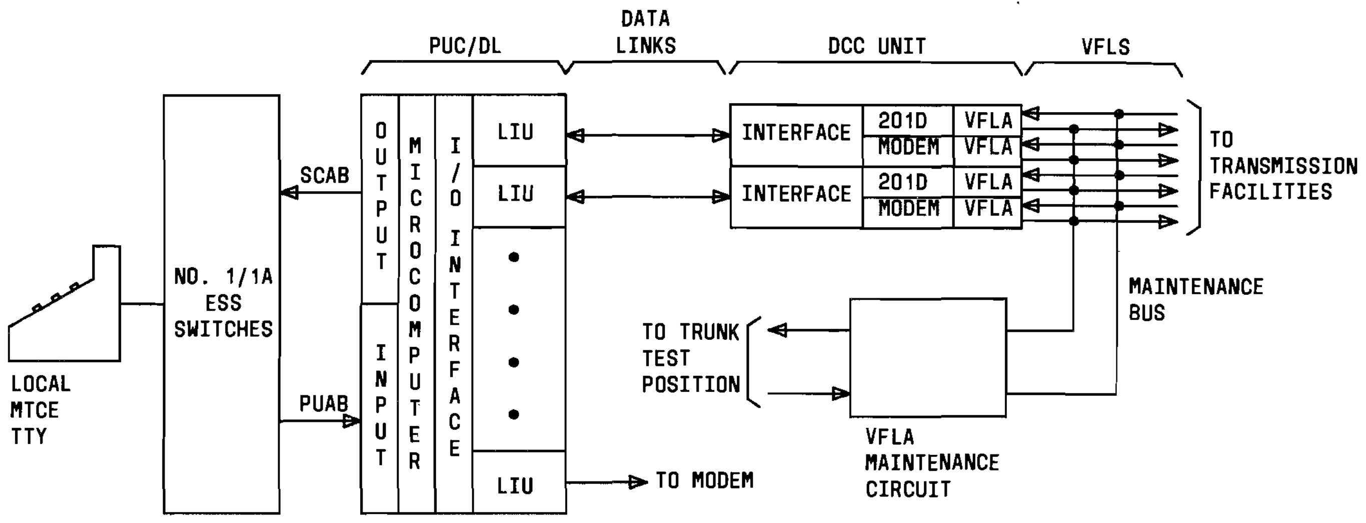
4.332 The maintenance messages contain the error and status reports. These messages are stored in the general reply buffer which is provided with a load and an unload pointer. A maintenance message is sent only when there is available space in the buffer.

Special Indicators

4.333 The special indicator type includes unload counters and pointers. An indicator is also provided to indicate when a data link switch is in progress.

Parameters

4.334 Parameters are single-word application maintenance messages which contain information used for updating and activating data links. They contain one byte of parameter information and



♦Fig. 29—PUC/DL CCIS Office Configuration♦

two bytes of information that distinguish the parameter message from other types. The four categories of parameters are:

- General
- Destination
- Data link basic
- Data link protocol.

General Parameters

4.335 The general parameters are applicable to the operation of the entire PUC/DL facility.

Destination Parameters

4.336 The destination parameters contain information relating to the input and output buffers. Each set (16 maximum) defines the destination buffer length, scratch block length, and the SCAM buffer length.

Data Link Basic Parameters

4.337 The data link basic parameters are used to specify the information applicable to individual data links. The two categories of data link basic parameters are protocol number and destination number.

Data Link Protocol Parameters

4.338 The data link protocol parameters are associated with the protocol routines and are applicable only to the protocol that is assigned to the data link.

C. MAINTENANCE

PUC/DL Faults

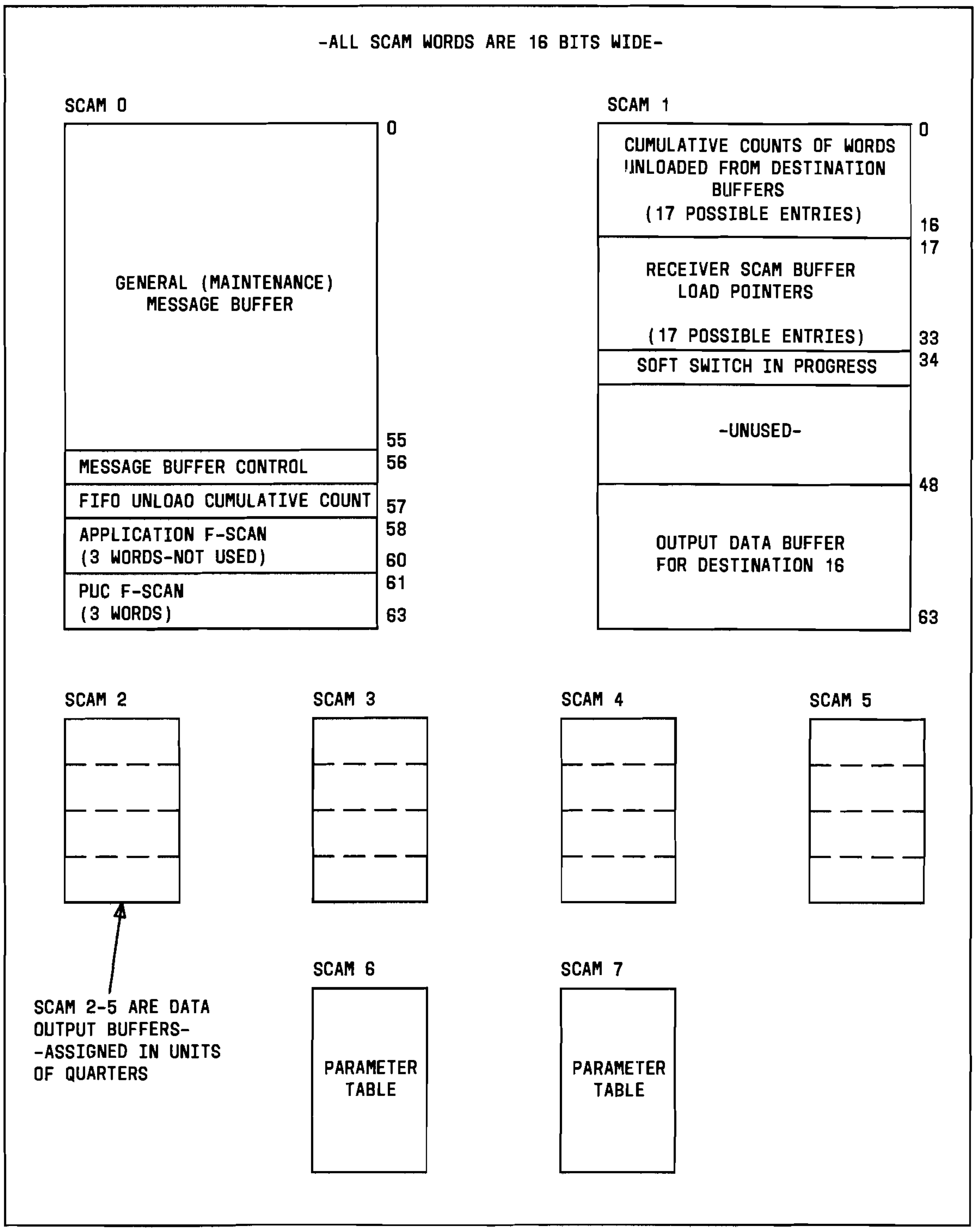
4.339 The PUC/DL faults are reported to maintenance personnel via the input/output terminals. Both the maintenance and the diagnostics on the PUC/DL facility are controlled by the ESS switch software pidents.

Data Link Recovery

4.340 The operational capability of the data links is checked by the PUC. When a trouble (carrier loss, protocol response failure from the remote end, excessive error rates) is detected, the PUC sends the report to the ESS switch. The recovery from a data link fault is handled automatically by the recovery pident PUDR. The recovery action involves establishing a working configuration with the data link, diagnosing the faulty link, and reporting the fault to the maintenance personnel via an output message.

Diagnostics

4.341 The diagnostic function includes the testing of various sections of the data link by looping


-ALL SCAM WORDS ARE 16 BITS WIDE-
SCAM 0
0
GENERAL (MAINTENANCE) MESSAGE BUFFER
55
MESSAGE BUFFER CONTROL
56
FIFO UNLOAD CUMULATIVE COUNT
57
APPLICATION F-SCAN (3 WORDS-NOT USED)
58
60
PUC F-SCAN (3 WORDS)
61
63
SCAM 1
0
CUMULATIVE COUNTS OF WORDS UNLOADED FROM DESTINATION BUFFERS (17 POSSIBLE ENTRIES)
16
17
RECEIVER SCAM BUFFER LOAD POINTERS (17 POSSIBLE ENTRIES)
33
SOFT SWITCH IN PROGRESS
34
-UNUSED-
48
OUTPUT DATA BUFFER FOR DESTINATION 16
63
SCAM 2
SCAM 3
SCAM 4
SCAM 5
SCAM 2-5 ARE DATA OUTPUT BUFFERS- -ASSIGNED IN UNITS OF QUARTERS
SCAM 6
PARAMETER TABLE
SCAM 7
PARAMETER TABLE

LEGEND:
FIFO - FIRST IN-FIRST OUT
SCAM - SCANNER ANSWER MEMORY

Fig. 30—PUC SCAM Layout

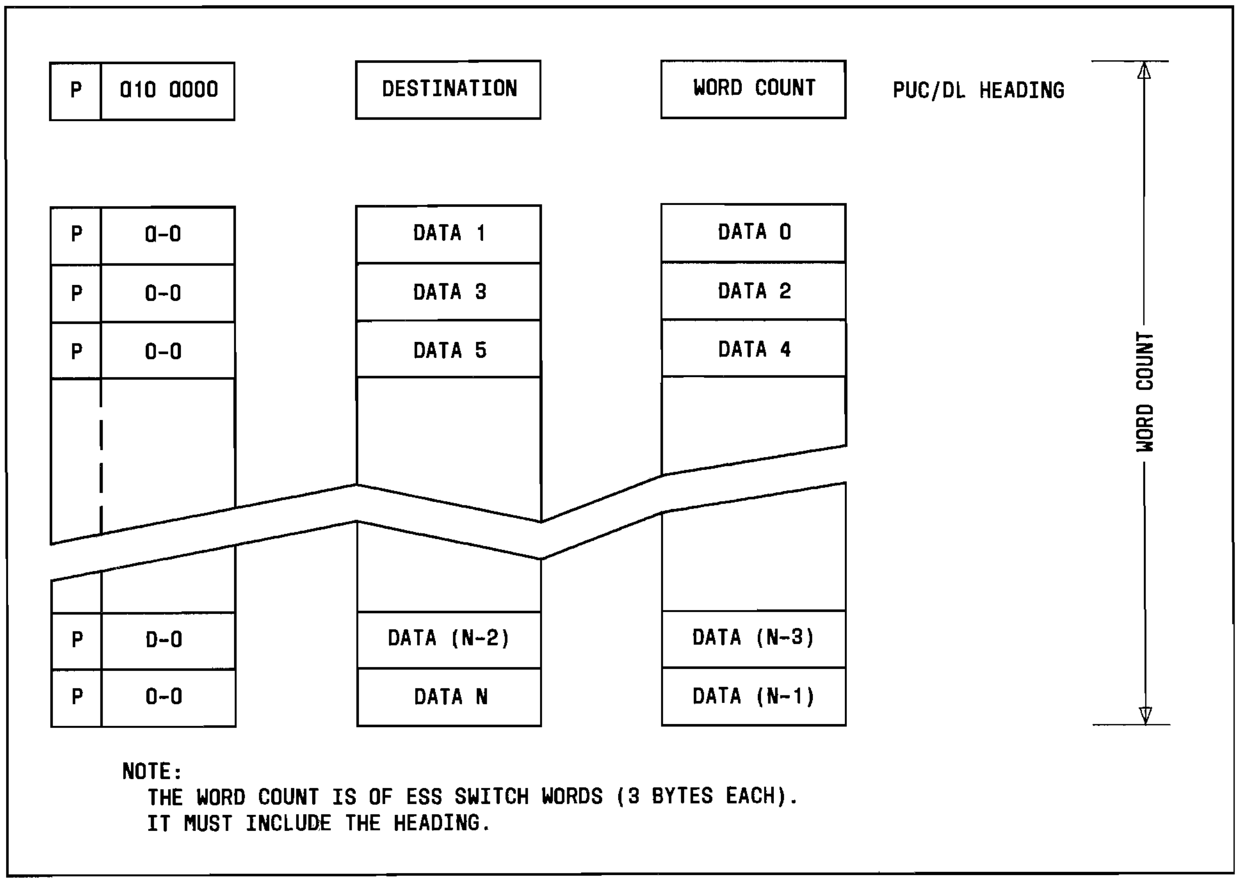


Fig. 31—Input Data Message

signals at different interfaces. The returned signal is compared to the transmitted signal with mismatches indicating a failure.

Maintenance Orders

4.342 The PUC maintenance orders are processed by the controller maintenance programs. They are normally used for bringing a data link on-line, clearing an error count, or switching between redundant data links to the same destination.

Audits

4.343 Audits are performed to build and initialize data structures in the read/write memory. The parameter tables located in SCAM 6 and 7 provide the input data for the audit routines. The audits provide space for the destination buffers and scratch blocks when these areas are requested by the parameter inputs. Whenever parameters are changed, the entire sequence of audits are performed.

STATION MESSAGE DETAIL RECORDING (SMDR) AND EXPANDED MESSAGE DETAIL RECORDING (XMDR)

A. General

4.344 The station message detail recording (SMDR) feature records considerable information about calls processed by ETS. This information can be used for accounting or cost control by the customer. The information is sent to the 93A or 94A customer premises system (Fig. 33 and 34) via data link and stored there on 9-track magnetic tape. The customer must have a Local Message Detail Recording System (LMDRS) or a Centralized Message Detail Recording System (CMDRS). The information is also used by the Bell System to assist in network design and maintenance.

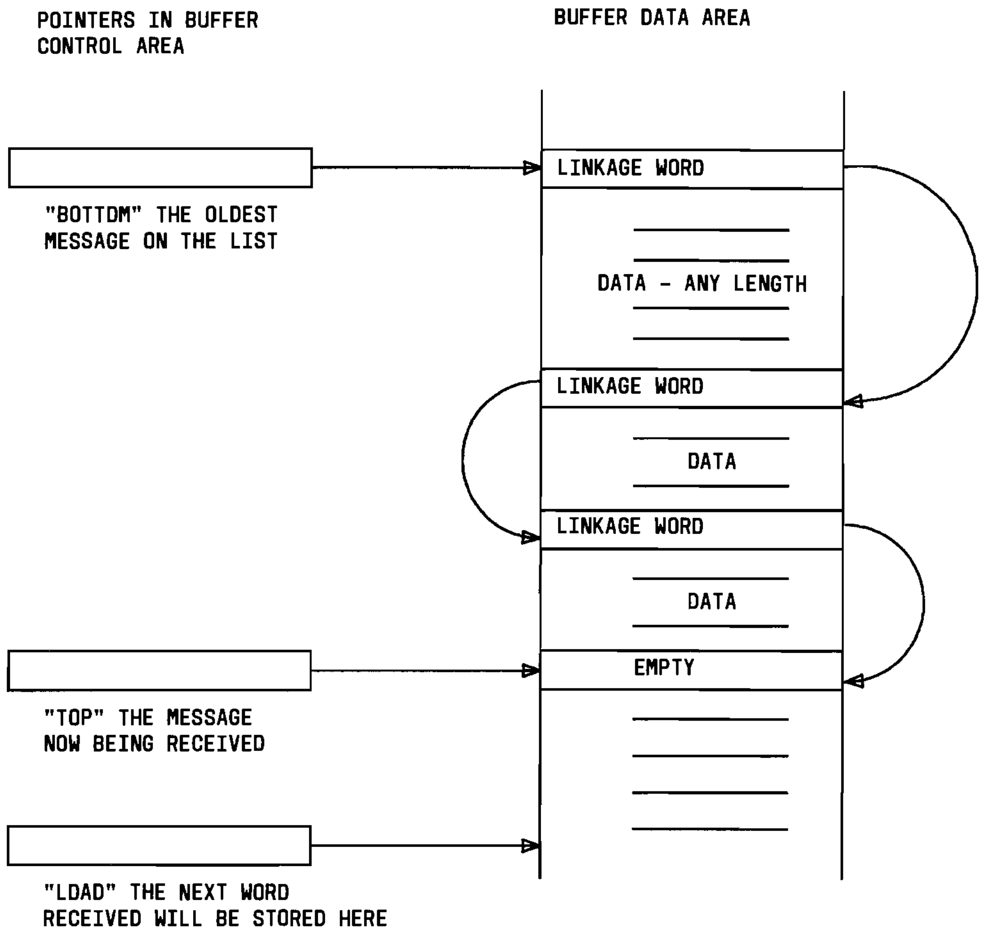
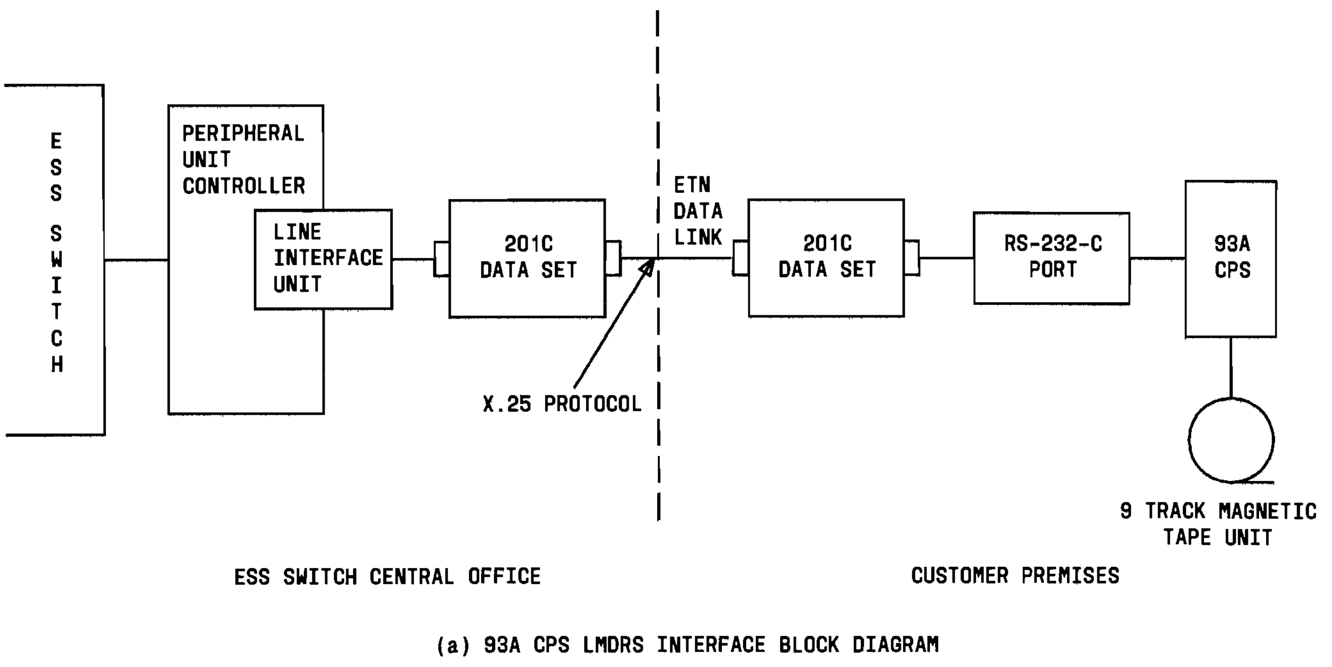

POINTERS IN BUFFER CONTROL AREA
BUFFER DATA AREA
LINKAGE WORD
"BOTTOM" THE OLDEST MESSAGE ON THE LIST
DATA - ANY LENGTH
LINKAGE WORD
DATA
LINKAGE WORD
DATA
EMPTY
"TOP" THE MESSAGE NOW BEING RECEIVED
"LOAD" THE NEXT WORD RECEIVED WILL BE STORED HERE

Fig. 32—PUC/DL Destination Buffer Structure

4.345 In 1E6/1AE6, SMDR provides records for originating calls, terminating ETS calls, and ineffective attempts. Calls to collocated stations where only the extension is dialed are not recorded. With 1E7/1AE7, additional information is available with the expanded message detail recording (XMDR) feature. "Dial 9," outgoing wide area telephone service (WATS), noncolloated common control switching arrangement (CCSA), foreign exchange (FX), and tie trunk calls with "1xx" access codes may be included on XMDR records. The SMDR feature only includes originating calls and ineffective attempts. Table B lists the information available on SMDR and XMDR records, along with the number of digits in each record.

4.346 When a call originates from or terminates to an ETS station, the MDRO or MDRI bits in the Centrex Common Block are checked to determine if a record is to be made. For XMDR, an XMDR indicator bit must also be set; and, for "dial 9" calls, the XMDR90 bit must be set. Various hooks are inserted into pidents CX1C, CXOR, ORDL, and ICAL to check the status of these bits.

4.347 If it is determined that a record is to be made, a routine in pident AMAC is called to seize and initialize an appropriate automatic message accounting (AMA) register. Routine AMMD18 is used to seize and initialize an 18-word AMA register for SMDR originating calls. Routine AMMD13 is used to seize and initialize a 13-word AMA register for SMDR terminating calls. Routine AMXMNO is used to seize and initialize an 18-word AMA register for XMDR.


E
S
S
S
W
I
T
C
H
PERIPHERAL UNIT CONTROLLER
LINE INTERFACE UNIT
201C DATA SET
ETN DATA LINK
X.25 PROTOCOL
201C DATA SET
RS-232-C PORT
93A CPS
9 TRACK MAGNETIC TAPE UNIT
ESS SWITCH CENTRAL OFFICE
CUSTOMER PREMISES
(a) 93A CPS LMDRS INTERFACE BLOCK DIAGRAM

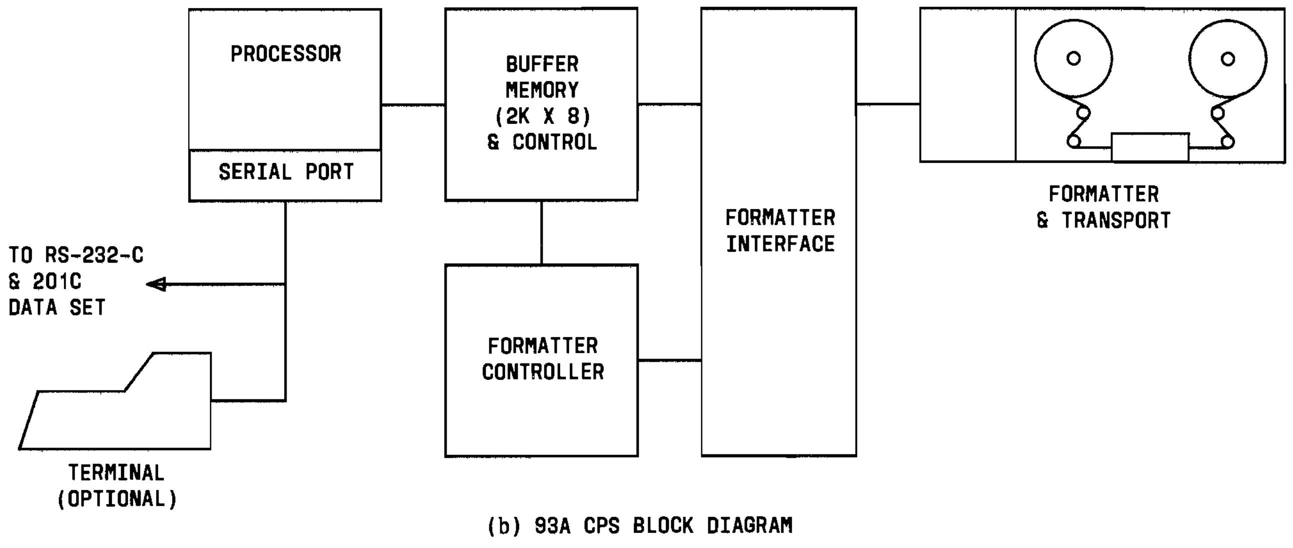
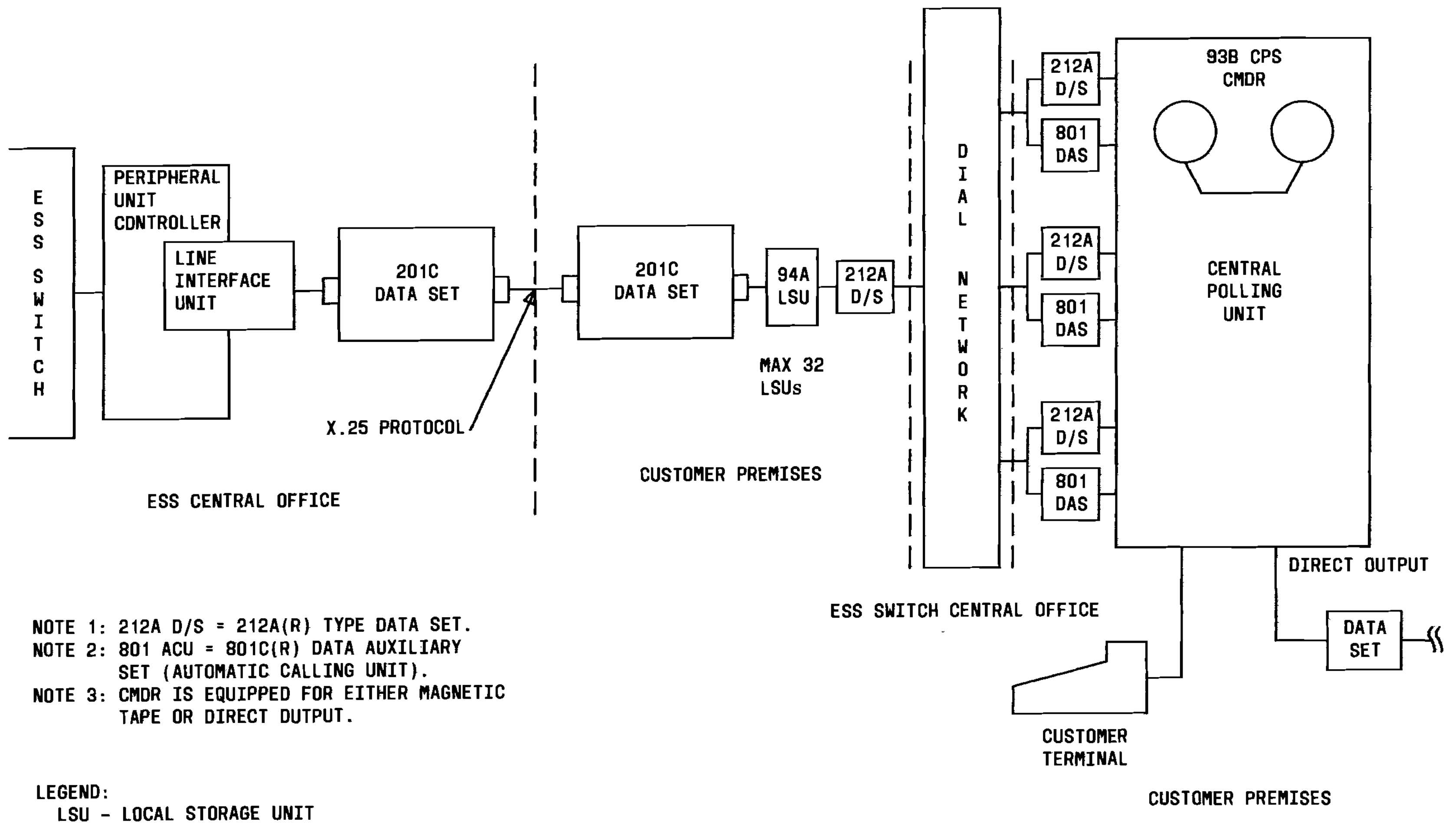


Fig. 33—SMDR/XMDR Hardware Configuration for LMDRS

4.348 Hooks are inserted into the main call processing programs to save information at appropriate times in the AMA register, for example, at the end of digit reception, or end of outpulsing.

4.349 Records are formatted and sent to the customer by routines in pident MDRO. There are several different record types. All record types are outputted by routine MDOUTP. If a record cannot be sent to the PUC immediately, MDOUTP will place it on a queue. The queue will then be unloaded later by MDQSRV. Routine MDQSRV is run as an E-level job and will output a maximum of five message detail records per entry.

4.350 Record type 01 contains SMDR information about an originating call. The information contained in this record type is as listed in Table B.



NOTE 1: 212A D/S = 212A(R) TYPE DATA SET.
NOTE 2: 801 ACU = 801C(R) DATA AUXILIARY SET (AUTOMATIC CALLING UNIT).
NOTE 3: CMDR IS EQUIPPED FOR EITHER MAGNETIC TAPE OR DIRECT OUTPUT.

LEGEND:
LSU - LOCAL STORAGE UNIT
D/S - DATA SET

♦Fig. 34—SMDR/XMDR Hardware Configuration for Centralized Station Message Detail Recording System♦

♦TABLE B♦

INFORMATION CONTAINED IN SMDR AND XMDR RECORDS

ITEM	NUMBER OF BCD DIGITS IN SMDR RECORD (TYPE 01)	NUMBER OF BCD DIGITS IN XMDR RECORD (TYPE 05 OR 06)
Call event code	1	1
Service feature	1	1
End of dial time	7	N/A*
Time change	1	1
Answer time	7	7
ARS pattern	1	N/A*
Outgoing trunk ID	6	6
Called number	10	16
Facility restriction level	1	N/A*
Incoming trunk ID	6	6
Calling number	7	7
Authorization code	6	N/A*
Account code	8	8
End of outpulsing time	7	7
Midnights passed	1	1
Disconnect time	7	7
Access code	N/A*	5
Record type	2	2

* Indicates that the item is not included in the specified record type.

4.351 Record type 02 contains SMDR information about a terminating call.

4.352 Record type 03 is a time change record for both SMDR and XMDR. When the ESS switch clock is changed, the old and new clock states are sent to ETS customers by routine MDTCHG.

4.353 Record type 04 is data information for both SMDR and XMDR. It is sent to the customer each midnight by routine MDDATE. This record includes an ESS switch identifier, the time, date, and a count of how many records have been lost either by the ESS switch or 93A CPS.

4.354 Record type 05 contains XMDR information for noncollocated CCSA, WATS, tie trunk, or FX calls. Record type 06 contains XMDR information for "dial 9" calls. Both record types have the same format and contain information as listed in Table B.

B. Account Codes

4.355 The customer may optionally have account codes. An account code is a number dialed before the called number. Optional account codes are recorded on SMDR records. Customer dialed account recording (CDAR) account codes may be included in XMDR records. The account code may be from 3 to 8 digits in length. The first digit of all account codes for a given customer is a unique Lead Account Code digit to indicate that an account code is being dialed. It must be chosen so as not to conflict with any on-network number (no on-network number can start with that digit). All account codes for any one customer must be of the same length. The second digit of the account code must not be 0 or 1, so that is will not conflict with a numbering plan area (NPA) code. This enables the call processing routines to recognize the dialed number as an account code, collect the proper number of digits, and record it. The Lead Account Code digit is not recorded on SMDR.

C. Call Event Codes

4.356 The call event code indicates how the call was disposed. The SMDR call event codes include:

- 0 - Completed directly
- 1 - Queued and completed
- 2 - Invalid NPA or NXX
- 3 - Invalid authorization code
- 4 - Insufficient FRL
- 5 - All facilities busy
- 6 - Abandoned on queue
- 7 - Timed out from queue
- 8 - Miscellaneous failure without queuing
- 9 - Miscellaneous failure after queuing.

Codes 0, 5, and 8 are also used for XMDR.

D. Service Feature Codes

4.357 The service feature code indicates that there were feature interactions on the call which affect the contents of the record.

(1) Station billing on attendant handled call applies.

(2) The record applies to the base to remote portion of a forwarded call.

(3) The call was routed to the attendant, due to the toll diversion feature (XMDR only).

AUTOMATIC CALL DISTRIBUTION (ACD)-ESS SWITCH MANAGEMENT INFORMATION SYSTEM (AEMIS) DATA BASE (MSDU)

A. General

4.358 The ACD-ESS Switch Management Information System (AEMIS) (available with ACD2 only) is a minicomputer-controlled system designed to:

- Measure and analyze agent/traffic data and provide detailed agent/traffic information
- Performance calculations
- Summarize past history
- Short-term forecast to the ACD manager.

4.359 To perform all of the AEMIS functions, a data base of the necessary data (describing the ACD) has to be established for the AEMIS by the No. 1/1A ESS switch. This is accomplished by the management information data base update program (MSDU) via a centrex data link. MSDU uses the 1-second entries provided by the block data link loading function of the centrex data link (DLIO) feature to format and load the data link orders. System configuration and control requires the inquiry-response system (IRES) feature; therefore, the IRES feature must be loaded for the AEMIS feature. The necessary data for the AEMIS data base includes:

(a) time of day

(b) the AEMIS trunk groups and associated trunk network numbers

(c) the facilities

(d) the queue data

(e) the agent to functional group assignments for each load compensating package (LCP) and the active LCP

(f) the four 4-digit extension assigned to each agent terminal.

4.360 ***Time of Day:*** The time-of-day function gives the AEMIS a snapshot of the ESS switch real-time clock. The time sent to the AEMIS is the year, month, date, hours, minutes, and seconds. The AEMIS resets the PDP*-11 clock to equal this time.

4.361 ***Call Store Configuration:*** The call store configuration function provides the AEMIS with a snapshot of the ACD changeable data, namely, the active LCP, the functional group (FG) patterns of the active LCP, and the queue data. The active LCP is the current invoked LCP. The FG patterns are the FG patterns of the active LCP plus any changes made by the ACD customer. The queue data is the interflow threshold, primary outflow threshold, and the secondary outflow threshold; if the night directory num-

*Trademark

ber (DN) is call forwarded, the forwarded DN is also sent; if not forwarded, all zeros are sent.

4.362 ***Initialization or Program Store Refresh:*** Both the initialization and the program store refresh functions send the same data to the AEMIS. The distinction is the rate at which the data link orders can be loaded into the data link output buffer. For the initialization request, the maximum rate is 20 data link orders per 1 second entry; whereas the maximum rate for the program refresh is 10 data link orders per 1-second entry. The data that is sent to the AEMIS for either function is:

(a) All the trunk network numbers for each trunk group number associated with the AEMIS

(b) All of the rows of data for each functional group for all of the LCPs in the data group associated with the ACD

(c) The number of simulated facilities for each simulated facility group associated with the AEMIS

(d) All of the agent terminals in the data group and their 4-digit extension number associated with the ACD

(e) The inflow threshold, call waiting lamp threshold, primary outflow threshold A, primary threshold B, primary alternate server pool number, secondary alternate server pool number, queue size, number of queue registers, inflow queue indicator, functional group number associated with this queue, DN of this queue, base night DN of this queue, and the primary alternate server pool.

The AEMIS can also request a subset of the initialization or program store refresh data. That is, any of the individual blocks of initialization or program refresh data can be requested separately.

4.363 When interrogation requests are received by ESS switch, appropriate data is sent to the AEMIS to satisfy these requests. This data may include a copy of the current program store data and a call store configuration or some subset of this program store.

4.364 In addition to sending the AEMIS data to satisfy the interrogation requests, the ESS switch sends a continuous stream of messages describing the call processing activity of the ACD customer. In order to report events to the AEMIS minicomputer, the ESS switch keeps a record of each incoming or outgoing call over customer trunking facilities and simulated facilities group. The ESS switch also keeps track of calls terminated to and originated from the agent consoles in order to maintain a record of the agent console state.

4.365 The AEMIS messages themselves may consist of one or two 24-bit words: 23 data bits, and one parity bit. The bits are numbered from right to left (0 through 23). Bit 23 is the parity bit. Bit 22 is a maintenance bit. When the maintenance bit is zero, this indicates an ESS switch maintenance request. When bit 22 is a one, the data link message contains AEMIS data. Bits 21 through 17 in single word messages contain the operation code (SOP). Bits 21 through 17 in the first word of a double word message are always set to "11101." The operation code (DOP) is contained in bits 16 through 13. Bits 17 through 21 of the second word of a double word message are always set to "11111" as an indicator that this is the last word of a double word message. The individual SOP and DOP code messages are listed in Fig. 23.

B. Call Processing

4.366 A series of call processing AEMIS messages are generated whenever an ACD simulated facility or a dedicated ACD-ESS switch trunk becomes involved in a call.

4.367 The sequence of facility messages that are sent to the AEMIS is essentially identical whether a simulated facility or a trunk is used. The messages sent to AEMIS are as follows:

(a) Facility seizure message (SOP2 for trunks, DOP1 for simulated facilities)

(b) Facility queued (DOP2)

(c) Facility dequeued (SOP3)

(d) Facility connected (DOP0)

(e) Facility idle (SOP4).

4.368 As indicated in (a) above, the facility seizure messages are unique for trunks and simulated facilities as shown below.

(a) Bit 16 of the SOP2 message is 0 for incoming trunks and 1 for outgoing trunks. When a trunk is seized and becomes traffic busy, the SOP2 message must be sent. The only exception to this is trunk seizures for a receiver attachment delay report (RADR) test. No message is sent on a RADR seizure.

(b) Bit 16 of the second word of the DOP1 has the same function for simulated facilities.

4.369 In all facility messages a constant identifier, the facility number field (bits 14 through 0), is used throughout the call as a tag. When the facility is a trunk, the facility number field contains a trunk network number; bit 15 is 0 to indicate a trunk.

4.370 When a simulated facility is involved, the facility number field contains a simulated facility register address (bits 2 through 0 of the address is truncated in bits 14 through 0). Bit 15 is 1 to differentiate a simulated facility from a trunk. In addition, bit 14 of the facility number is always 1 to differentiate a simulated facility register from a queuing register.

4.371 In addition to the call processing facility messages, AEMIS messages are sent for various trunk maintenance states. These states may be initiated either via the TTY, the trunk and line test panel, or as a result of a hardware failure during call processing. The AEMIS maintenance messages are:

(a) Trunk disabled (SOP5)

(b) Trunk high and wet (SOP6)

(c) Trunk locked out (SOP11)

(d) Trunk active-in-service (SOP12)

(e) Trunk make busy (TMB) or carrier group alarm (CGA)-(SOP7).

CENTREX STATION REARRANGEMENTS (CSR)

A. General

4.372 The CSR feature allows centrex customers to directly access the No. 1/1A ESS switch to:

- rearrange extensions
- activate and deactivate extensions
- change certain features
- display information about extensions.

Before CSR, a centrex customer could make changes to their centrex group only through a service order, which could take several weeks. With CSR, the change takes place immediately.◀

5. MAINTENANCE SOFTWARE FUNCTIONS

5.01 In order to maintain a high degree of continuous and reliable service, the ESS switch provides duplicated equipment units, special maintenance circuits (which detect troubles and provide diagnostic access), and a comprehensive programmed system of maintenance programs. The ESS switch maintenance programs are used to detect and localize system troubles as well as control system configuration changes which may be required for maintenance purposes. Figure 35 provides an overview of ESS switch maintenance control structure.

5.02 The ESS switch maintenance software can be grouped into the following functional areas:

- Maintenance Control
- Fault Recognition (FOR)
- Diagnostics (DIAGs)
- Routine Exercises (REXs)
- Audits
- Diagnostic Results Processing.

5.03 The FOR programs attempt to recover the call processing ability of the system by establishing a working configuration of office equipment when a trouble is detected. These programs are nondeferrable and are of the highest priority in the maintenance program hierarchy. In general, FOR programs are initiated as a maintenance interrupt when a trouble is detected. The FOR runs to completion under interrupt control, after which call processing is allowed to continue undisturbed. The primary function of the FOR is to determine whether the faulty unit, as detected, has permanently affected system operations and whether substitution of a duplicate system unit is necessary to restore normal

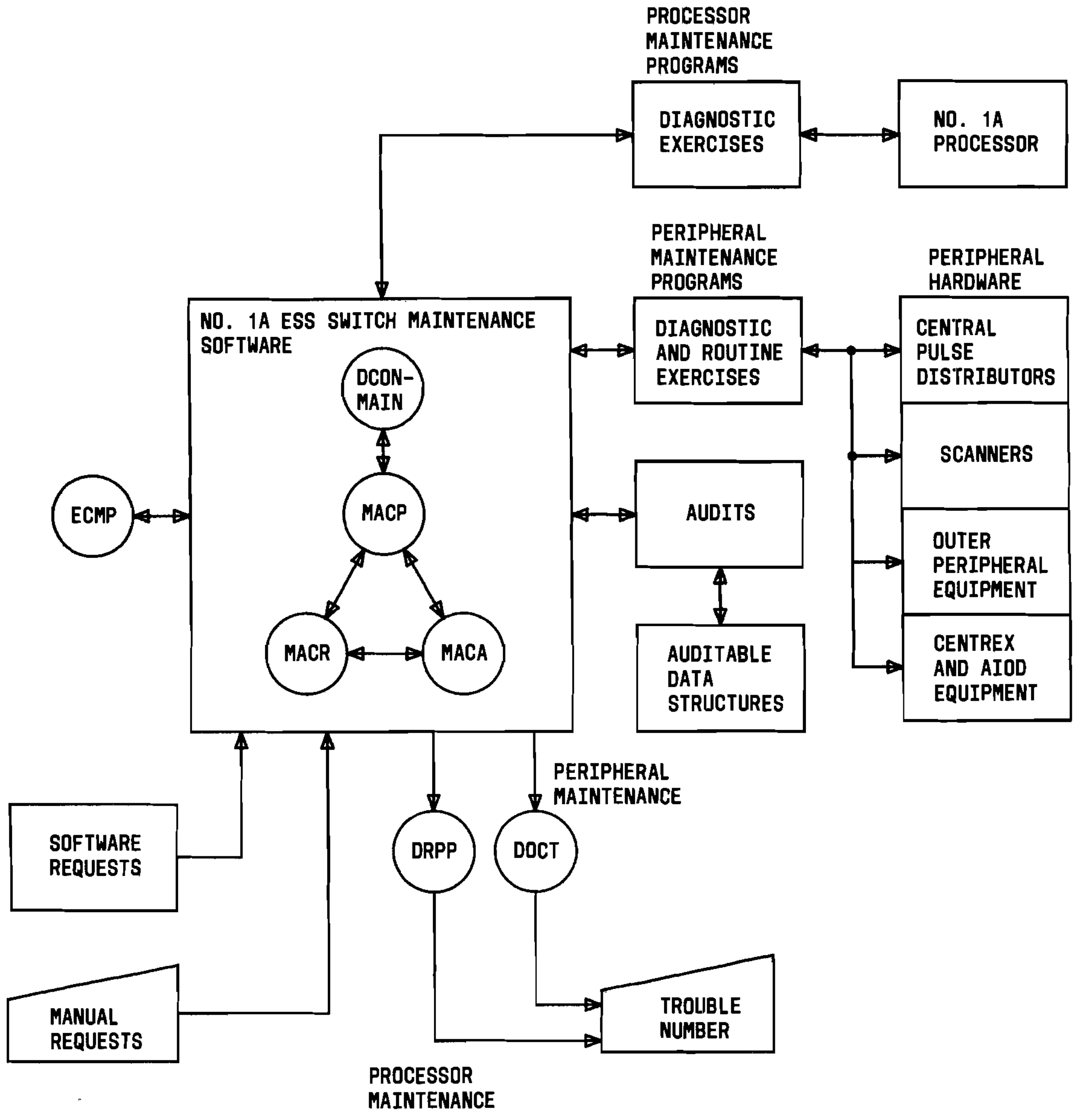

PROCESSOR MAINTENANCE PROGRAMS
DIAGNOSTIC EXERCISES
NO. 1A PROCESSOR
PERIPHERAL MAINTENANCE PROGRAMS
PERIPHERAL HARDWARE
NO. 1A ESS SWITCH MAINTENANCE SOFTWARE
DCON-MAIN
DIAGNOSTIC AND ROUTINE EXERCISES
CENTRAL PULSE DISTRIBUTORS
SCANNERS
ECMP
MACP
AUDITS
OUTER PERIPHERAL EQUIPMENT
MACR
MACA
AUDITABLE DATA STRUCTURES
CENTREX AND AIOD EQUIPMENT
PERIPHERAL MAINTENANCE
SOFTWARE REQUESTS
DRPP
DOCT
MANUAL REQUESTS
TROUBLE NUMBER
PROCESSOR MAINTENANCE

Fig. 35—No. 1A ESS Switch Maintenance Control Structure

operation. The maintenance control software has no part in the execution of FORs other than to recognize that an interrupt has occurred and to remove any related maintenance client currently in control of the maintenance scratch area. The FOR may request maintenance control to run a DIAG during normal base level maintenance.

5.04 The DIAGs are the second highest priority maintenance programs handled by maintenance control. DIAGs attempt to localize a trouble within a faulty unit down to the smallest possible number of replaceable circuit packs. A DIAG consists of a series of sequential tests which are executed against the suspected faulty equipment type. Results from the tests are converted into a TTY output message by dictionary programs. The resulting output message provides a trouble number which is used as an index into a trouble locating manual (TLM). The TLM provides a list of circuit packs which, if faulty, could have produced the trouble number.

5.05 The REX programs consist of both scheduled automatic routine exercise programs (AEX) and demand request routine exercise programs. The REX programs are designed to check for faults that might otherwise go undetected, to search for uncorrected errors, to check the trouble detection circuits, and to exercise infrequently used hardware.

5.06 Audit programs comprise a system, complete with control structure, designed to maintain the validity of stored data structures in the ESS switch memory. Audits can detect, repair, and reinitialize data elements or structures as necessary.

5.07 Diagnostic Results processing is an automated means of assimilating the so-called raw data produced by the diagnostic programs. The resultant output from the diagnostic results processing programs may consist of a suspect list of faulty equipment or an index into such a list.

MAINTENANCE SOFTWARE CONTROL PROGRAMS

A. Maintenance Control Program (MACP)

5.08 In the No. 1A switch, MACP schedules and controls the execution of deferrable base level maintenance programs, eg, hardware diagnostic and routine exercise programs as well as other nonmaintenance programs. Specifically, all paged programs are executed under MACP control. The MACP interfaces with the paging program (PAGS) to accomplish the run time loading and execution of paged programs.

5.09 The MACP consists of a group of control routines which use application dependent job tables (MTBL, MJTB) to perform job scheduling for a particular application. These job tables specify the MACP resources and client jobs and, thus, define the MACP environment for the application system. A set of macros and subroutine (MAPL) is provided by MACP which can be used by client programs for delay timing, inhibiting maintenance interrupts, etc.

B. Base Level Scheduled Maintenance

5.10 All noninterrupt maintenance in the No. 1A ESS switch peripheral (ie, not processor) area is initiated by an entry to pident MACA. This entry is scheduled as one of the class E jobs in the ECMP base level task dispenser. Pident MACR (maintenance control peripheral program) is entered to set up the correct return to the ECMP and to check for interfering MACR jobs in a multi-MAC environment. Pident MACA then checks a counter to see if trunk maintenance should be done. If not, a check is then made to see if audit work should be done. Scheduled routine audits are run in parallel with other No. 1A ESS switch maintenance. During the normal base level cycling, audits are run every third pass through this routine. If it is not time for audit work, control is passed to the job control routine in MACP. The MACP will then continue a client already in progress at the client's next segment address or, depending on the scheduling algorithm, look for other maintenance work. The MACR provides the basic control structure for peripheral maintenance as described below.

Diagnostic Requests

5.11 The MACP enters MACR when looking for diagnostic requests. Checks are made to see if MACR is currently busy in the other maintenance class. Pident MACR, as a client of MACP, runs out of the maintenance class, subclass 0 or subclass 1. Pident MACR can only run one peripheral maintenance job at a time; ie, if MACR is busy in subclass 0, a new job cannot be started in subclass 1. If upon entry, MACR finds itself busy in the "other class," then control is passed back to MACP to abort the job. If MACR is available to start the diagnostic, then MACP is given a common abort address. Should MACP abort the job, this common abort address would be given to the peripheral maintenance client in progress. Then MACR will hunt for diagnostic requests using a hunt routine. Refer to the MACR program listing for a detailed account of diagnostic request processing.

Routine Request Table Requests

5.12 The routine request tables store requests (from the maintenance personnel or other maintenance programs) to start a particular peripheral maintenance program. The routine request table (RRT) is divided into two levels: RRTA and RRTB. Manual requests are stored in RRTA; requests from other programs are stored in RRTB.

5.13 MACR provides entries for loading the RRTs. A fail return to the client is made if there are no available entries in the RRTs. If the proper entry to MACR is taken, an entry into the top slot in level B can be guaranteed; all previous entries are pushed down one entry. The lowest entry may be destroyed.

Normally, each entry loaded into level A or B is loaded following any previous entry. Requests to run jobs are answered starting at the top of each RRT level, thus providing a first-in, first-out sequence. For each job, MACR will schedule a peripheral maintenance search of the RRT from MACP.

5.14 To process a job requested in RRTA, MACP enters MACR under control of the job scheduler routines in MACP. Again checks are made to see if MACR is available. Note that the same initial checks were made for the MACSDIAG entry. As entries are processed out of RRTA, the remaining entries are moved up one position. The MACP is entered to run the job.

5.15 For job requests in RRTB, MACP again enters MACR. The initialization checks are identical to those described earlier, down to the point where the RRTB is examined for requests.

AEX Job Requests

5.16 The AEX jobs are run on a fixed schedule, eg, per half-hour (on the half-hour and hour), per hour (on the hour or half-hour), every three hours, or daily at midnight. MACR controls two classes of AEXs, ie, class II routine exercises and class III routine exercises. The class II jobs are run out of the AEX2J1 and AEX2J2 job tables. Individual jobs are turned on by corresponding bits in flag words M4FLG1 and M4FLG2, respectively. These jobs are scheduled by an entry every half-hour from the ECMP.

5.17 The jobs in the AEX2J1 table are nonhardware testing programs and routines. Their basic functions are information reporting and auditing or verification. The jobs in the AEX2J2 table are solely concerned with hardware testing.

5.18 The class III routine exercise programs are the lowest priority programs in the ESS switch peripheral maintenance software structure. These jobs, run out of the MACR AEX3JB job table, are executed during spare time when no other AEX (class II) jobs are waiting. The AEX3JB jobs are turned on by a corresponding bit in M4AEX3; these bits are also set by ECMP visits to MACR.

5.19 To start an AEX job, MACP enters MACR. As with other peripheral maintenance requests, checks are made to see if MACR is busy. If busy, the job is rescheduled for a later attempt. If MACR is available then an abort address is given to MACP.

C. Fill Maintenance

5.20 During periods of exceptionally light traffic, or when the call processing load is low, the ECMP base level task dispenser may run out of work to do. Rather than waste time looking for work that is not yet scheduled, the available time is used running audits as "filler." Audit control responds as for a scheduled entry.

AUDIT SOFTWARE

A. General

5.21 The audit program package provides an effective and rapid method of protecting the ESS switch from errors in temporary and permanently stored data structures. The audit system has access to system storage as shown in Fig. 36.

B. Audit Error Detection

5.22 The audit system is tasked with both detecting and correcting (if at all possible) software data structure errors which may occur from a great many sources. In general, an audit consists of an evaluation of a data item, ie, bit(s) word(s), etc, and based on the evaluation (actual value or data credibility), a pass/fail decision is made. If errors are detected, the audit in control or other audits may attempt to correct the faulty data. Data structures which can be audited exhibit one or more of the following properties:

(a) Constant Data: The simplest auditable memory is that which is known to contain constant information. Errors can be detected by simply comparing the backup or redundant data with the actual data.

(b) Timed Memory: Certain types of data structures are known to have short holding times relative to the length of a call. Other structures are not allowed to be in a transient state for a long period of time. Therefore, it is possible to monitor these structures and if a certain threshold time limit is exceeded, the memory is deemed to be in error. The outpulsing register is a good example of a timeable piece of software since no call should be in the outpulsing state for more than a few seconds.

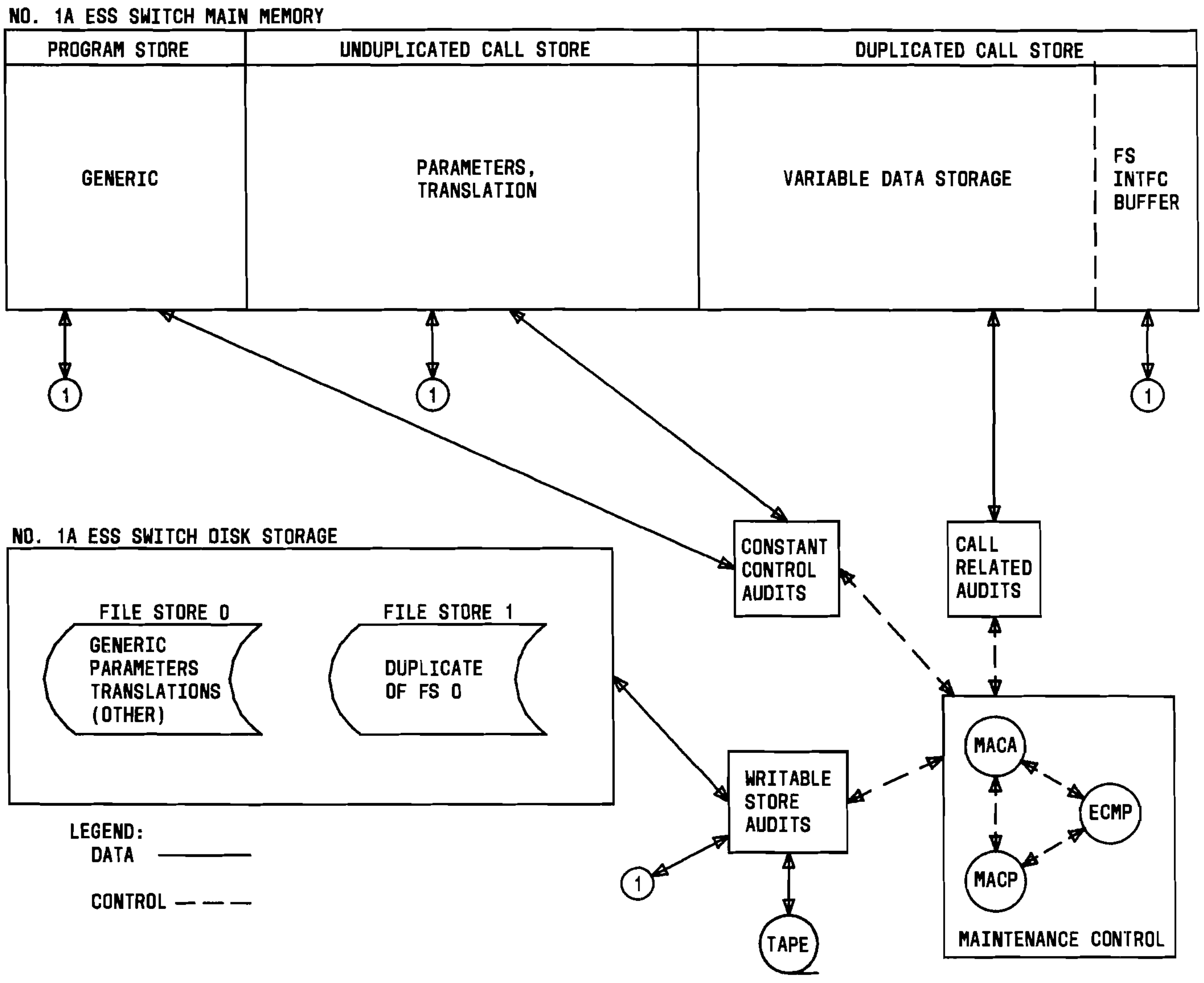


Fig. 36—No. 1A ESS Switch Audit System

(c) Redundancy: If a data structure contains redundancy, then all additional pieces of information may be compared by an audit to determine if they agree. If total agreement is not found, then the data structure is assumed to be in error. In many cases this redundancy is not essential to the normal processing of the data structure and must be added at some expense of real time or memory to provide auditability.

C. Correction Strategy

5.23 On detection of an error in the constant data, an audit will correct the erroneous information from the backup data. An error in a more sophisticated data structure or a time-out will cause all identifiable memory associated with the structure to be idled and maintenance to be requested on all identifiable hardware. In general terms, then, no attempt is made to return the incorrect structure to its proper operational state. Any incorrect guess by the audit as to the proper busy state could be fatal to system sanity. This strategy was chosen as the safest way to guarantee a valid software state.

5.24 In some rare cases where there is an overwhelming N-to-1 vote for the state of a piece of software, the one dissenting piece of information will be changed to agree with the majority.

5.25 After the error is cleared, an audit message is printed on the TTY describing the error. Other audit messages will be printed for each additional piece of equipment restored or memory idled. These messages are printed to notify the craftsperson that an error has been detected and are to be used in correcting the source of the error.

5.26 Finally, other audits are requested at high priority to examine all memory functionally related to the data structure in error.

D. Audit Subsystem Requirements

Audit Segments

5.27 An audit segment is defined to be that amount of base level time in which an audit runs without giving up control or taking a time break. The actual time allotted is a theoretical figure for maintenance programs determined by systems engineering. This figure is considered to be the amount of time in which base level call processing can be suspended without unduly influencing the system.

Short Segment

5.28 A short segment is an untimed segment that is known to run less than 2.4 ms under error-free conditions. A short segment audit usually has a very simple job whose execution time can be computed by counting cycles.

Timed Segment

5.29 Timed segment uses a timing method external to the audit to determine elapsed segment time. Once the proper segment time has passed, a flag is set by the external timing program. This flag is consulted by the audit at convenient points in the program where a real-time break would be allowed. If the flag is set, then a segment break is taken. Routines are provided to do this timing for the client audit. Timing is done using the maintenance timer. Since this timer measures elapsed real time and J level consumes 30 to 50 percent of that time, the timer must be set to a value equal to 2.4 ms plus the time consumed by the average J. By selecting an average J of 2 ms or 40 percent of all time, the audit will get less time at peak traffic and more time in an idle system, thus making segment times somewhat dynamic in response to load. Timing subroutines can set the clock with the G-level interrupt inhibited so that time-out can be detected by simply reading the clock to determine if it is zero. It is not necessary to zero the clock at the end of an audit since time-out causes no system action.

Interject Long Segments

5.30 Audits that must run for long periods of time without a real-time break must run an interject long segment. This is necessary because certain call configurations cannot tolerate delays of this length without base level action. The interject segment warns the system not to start any delay sensitive jobs and then waits six peripheral order buffer cycles to allow any delay sensitive jobs that have been started to complete. The audit runs its long segment in interject and then takes an appropriate number of "do nothing" segments to allow the system to recover call processing ability. During the peripheral order buffer cycle wait period, the normal maintenance control program entry is shut off.

> ➧*Note:* Some audits do a fixed number of "do nothing" segments, others do a variable number based on the amount of work performed by the audits.➧

Base Level Scratch

5.31 The audit system is allocated a large block of scratch area in call store.

Audit Holding Time

5.32 Audit holding time is that interval of real time in which an audit is holding the maintenance control (MAC) scratch pad. Audit cycle time is the sum of the holding times for all audits in the system. It is generally measured as the time between one pass of audit 30 and the next pass. This ensures that all audits have been run at least once. Determination of exact audit cycle and phase times is a very complex process depending on a large number of variables. Better figures are obtainable by manual timing in the field under actual conditions and even these results can vary widely from office to office, or within a single office at different times during the day. Keeping this in mind, the rest of this section attempts to formulate some guidelines for predicting audit times for a new office by comparing its parameters to base offices whose audit times are known.

Audit Cycle Time

5.33 Audits are run primarily as fill work when the main program has nothing else to do. The scheduled routine maintenance entry to audits is from main program job class E, a relatively infrequent entry as compared to the fill entry. Since most auditing is done in spare real time, and spare real time is a function of traffic in the system, then audit cycle time (ie, time required for one complete pass of all routinely scheduled audits) increases with traffic in the system.

Traffic

5.34 Individual audit holding times are also a function of traffic. Exact relationships are not clear since each audit responds differently to traffic stimulation. For instance, the map audit takes only a few cycles to process an idle path, but a few thousand to process a busy path. On the other hand a constant audit is unaffected by traffic, and an idle link list audit is inversely proportional to traffic since there are fewer idle structures to process during busy periods.

Number of Networks

5.35 Audit cycle time is almost directly proportional to the number of networks in an office. Increasing the number of networks also increases the number of trunks and lines to be audited, and it indirectly increases the amount of auxiliary memory to be audited, such as registers, peripheral order buffers, and number of call stores. By making audit cycle time a function of the size of an office, as expressed by the number of networks, it is obvious that a large office has more audit work than a small office.

Phase Times

5.36 Phase times are responsive to many of the same factors that govern audit cycle times but in different degrees. Obviously available real time is not a factor in phases because stitched audits do not run out of class E or as fill work and do not take real-time breaks for call processing. Traffic is not a variable in phases 1 or 6 but is important in a phase 4 where additional work must be done for each busy or transient connection. The number of networks is the primary factor governing the length of phase 1 with the phase 1 time being almost directly related to network size. ▶Phase 4 through 7 are comprised of signal distributor audit actions. The signal distributor actions take 18.6 seconds and are invarient, so long as the audits 6 through 72 takes less than 18.6 seconds, office size is not a factor.◀

Audit Scheduling

5.37 Audits are run by maintenance control on a demand basis if software errors have been detected in the system or if a TTY audit request has been made. The audits are run on a routine basis (class E main program entry) if no higher priority maintenance work exists in the system. This routine mode serves as the systems chief protective defense against unknown software errors. Additionally, audits run as fill work when the main program job scheduler has available real time with no scheduled work to do. All other audit functions, such as phases, demand audits, and interrupt level audits, are after the fact and serve to clean up known errors.

Request Tables

5.38 The audits are requested out of five request tables (audit control blocks):

- Phase (runs audits in a phase)
- Demand high priority
- Demand low priority
- Routine high priority
- Routine low priority.

Each request table consists of three call store words. Each bit in the request table corresponds to a single one of all the possible audits. Requests are served by scanning the tables in order of priority (highest to lowest) from right to left until a 1 is detected. Once a 1 has been found, its corresponding bit position is zeroed in all tables to clear any other requests for that audit, and its bit position is used to index a vector table to run the correct audit. The priority structure ensures that certain critical audits will have execution priority over the least critical ones.

Routine Scheduling

5.39 Audits running as fill work get a smaller and smaller percentage of real time as traffic increases. This design provides a valuable dynamic audit response to load conditions. The system is simply trading some of its audit protection for increased call capacity as needed. This technique is useful but it has the disadvantage of providing the least protection at a time when the probability of errors and vulnerability of the system is greatest. Conversely, it provides the most protection in an idle system where the error rate is very low. In addition, relying solely on the class E visits to audits greatly increases the audit cycle time while the higher traffic causes a greatly increased error propagation rate. These two diverging trends cause a consequent decrease in the probability that the audit needed to detect a specific error will run within a short enough time span of the occurrences of the error to prevent further call degradation, interrupts, or phases. This routine protection function is also shared with the less detailed and faster checks of the data validation. The data validation, since it has a fixed entry rate, is not subject to traffic variations and is thus more likely to detect catastrophic errors during peak traffic situations.

E. Audit Activity During a Phase

5.40 During a system reinitialization or "phase," audits are executed in a "stitch" mode; ie, the audits comprising the phase are stitched together one after another with no real-time breaks taken. All other central control processing activities are suspended until the phase is completed.

Audit Output Messages

5.41 The audit system provides output messages to keep the office maintenance personnel aware of audit system activities.

DIAGNOSTIC AND EXERCISE SOFTWARE

A. 1A Processor Diagnostics

5.42 The purpose of 1A processor equipment diagnostic programs is to provide a programmed test capable of verifying that a unit is free of classical faults, and if a fault exists, to assist in locating the fault to a small number of replaceable circuit packs. The diagnostic programs are entered under the following conditions:

(a) For fault location in the automatic fault recovery sequence following detection of a trouble

(b) For verifying the integrity of a growth unit before it is made available to the system

(c) For verifying the integrity of a unit before it is returned to service following a restoral request at the equipment frame

(d) As an error analysis function for locating marginal circuit packs

(e) As an automatic routine exercise function

(f) As an on-line or off-line troubleshooting tool with special options via the TTY.

5.43 Diagnostic programs for processor units have many common structural features, in that:

(a) All diagnostic programs are written in a high-level diagnostic programming language.

(b) The compiled output of the diagnostic language is in a data table.

(c) This data table is interpreted at execution time by a set of task routines linked together by a task dispenser.

Inasmuch as these and other common features exist, a common diagnostic control program, DCONMAIN in No. 1A ESS switch, oversees the individual diagnostic programs and performs many of the common functions needed by the individual unit diagnostic programs. DCONMAIN is the program store resident interface between these individual diagnostic programs and other ESS switch programs. Specifically, DCONMAIN has responsibility in the following areas:

(a) Overall control of the diagnostic programs

(b) Interfacing with MACP for execution of diagnostic requests and with PAGS for paging operations

(c) Interfacing with fault recovery and error analysis programs for prediagnostic initialization and post-diagnostic final handling

(d) Interfacing with the diagnostic results postprocessing program (DRPP) for the implementation of trouble locating procedures to identify suspected faulty circuit packs.

5.44 The 1A processor diagnostic programs provide for computer-aided diagnosis and troubleshooting of the following equipment:

(a) Central control

(b) Memory units:

(1) Program store

(2) Call store

(c) File store and the Attached Processor Interface

(d) Auxiliary data system:

(1) Data unit selector

(2) Tape unit controller

(e) Auxiliary unit bus

(f) Input/output system

(g) MCC and processor peripheral interface unit

(h) Memory buses:

(1) Call store bus

(2) Program store bus.

Diagnostic Execution Control

5.45 All requests to run a 1A processor diagnostic enter a MACP which buffers the requests and initiates them when system time and resources become available. The diagnostic programs run as MACP clients on base level. These programs execute during the segment of time allotted to MACP to perform deferred maintenance.

5.46 The general flow which is typical for processing a diagnostic request is illustrated by the following sequence of events:

(1) A request to run a diagnostic enters MACP from fault recovery programs following an interrupt, from manual control facilities (ie, TTY, MCC), or from automatic routine exercise programs.

(2) The MACP serves the diagnostic request and transfers to DCONMAIN.

(3) The DCONMAIN initializes itself, and based on the unit type under diagnosis transfers to an appropriate prediagnostic initialization routine.

(4) The prediagnostic initialization routine determines if the diagnostic can still be validly executed, prepares the unit for a diagnosis (if necessary), and returns to DCONMAIN with a go or no-go decision.

(5) If the decision is no-go, DCONMAIN does not run the diagnostic but goes to step (9) below.

(6) If the decision is go, DCONMAIN requests the paging into program store of the first set of tests (data table), any associated paged task routines, and the task dispenser.

(7) The DCONMAIN controls the execution of the diagnostic and interfaces with other programs for input/output messages, storing of test results, etc.

(8) After the diagnostic has been run, DCONMAIN activates DRPP to process any failing data, prints the diagnostic test results, and sets flags indicating the test results.

(9) The DCONMAIN then transfers to the error analysis program which records the test results and returns to DCONMAIN.

(10) The DCONMAIN then transfers to a final handler program where the decision is made whether or not to restore the unit to service. The final handler will end the MACP job or return to DCONMAIN. If a return is made to DCONMAIN, DCONMAIN completes the execution of the job by returning to MACP.

(11) The DCONMAIN is reentered by MACP to set up and initiate summary processing by DRPP. The DRPP will generate and print the pack list output message and return to DCONMAIN. DCONMAIN then returns to MACP to terminate this job.

Diagnostic Program Structure

5.47 The 1A processor diagnostic programs are data-table-driven programs controlled by DCONMAIN. These programs are composed of three distinct program types: a task dispenser, diagnostic data table, and task routines. These are all paged programs and are declared to the switching assembly program as follows:

(a) CPSECT (Control Program Section): The task dispenser section which includes the vector table and high usage task routines (these are the XXDG00 pident [per unit control programs]).

(b) PGSECT (Program Section): The data table, (these are the XXDG01, 02, etc, pidents).

(c) SUBROUTINES: Task routines that are not a part of a CPSECT.

5.48 The task dispenser is a section of code which reads an index word in the data table and transfers to the appropriate task routine based on that index word. Entries to the task dispenser are via a transfer vector table at the start of the task dispenser. The 1A processor task dispensers must return to DCONMAIN to end each segment within 2.5 ms after being entered by DCONMAIN.

5.49 The data table is a series of blocks, each block consisting of an index word and zero or more data words. The index provides the task dispenser the means to locate the task routine which tests the unit. The data words are used by the task routines to assemble orders for the test.

5.50 The task routines are subroutines which are transferred to by the task dispenser based on the index contained in the data table. These routines apply the tests to applicable units under diagnosis according to the data from the data table. For test evaluation and raw data storage, the routines transfer to DCONMAIN.

5.51 The general structure of the diagnostic tests consists of three levels. The basic characteristics of the three levels of this modular test structure are described as follows:

(a) Test: A test applies a combination of inputs to a small block of circuitry, compares resulting outputs with expected no-fault results, and records a pass/fail indication.

(b) Test Segment: A test segment is a collection of tests to be run in sequence with no real-time break.

(c) Test Phase: A test phase is a collection of test segments that tests portions of a processor unit. If a failure is detected, the diagnostic may be terminated if the accumulated test results can pin down a fault in the area tested. The diagnostic may also terminate on a detected test failure if further testing requires that the circuits tested up to this point be working properly.

B. ESS Switch Peripheral Diagnostic and Exercise

5.52 The peripheral diagnostic and exercise program package is designed to localize peripheral equipment faults, reconfigure associated peripheral units, and provide status information concerning the peripheral system. Diagnostic and exercise programs are provided for the following peripheral equipment:

- Network, network controller, and signal distributor
- Scanner and scanner answer bus
- Central pulse distributor and buses
- Peripheral unit bus.

Peripheral Unit Diagnostic Programs

5.53 The peripheral unit diagnostic programs are used to localize faults within the peripheral unit to a small number of circuits. The initiation of these programs is normally done by way of the FOR programs or manually from the maintenance TTY. The diagnostic request is scheduled via a maintenance control block request administered by MACR or by setting the appropriate request bit in the status word of the unit.

5.54 The diagnostic programs under MACR control can be divided into control operations and functional testing. The control operations perform the following:

(a) Determine whether or not the requested diagnostic can be performed

(b) Establish the appropriate system configuration for diagnosis

(c) Initialize the unit to be diagnosed

(d) Process the test results in a form suitable for input to the diagnostic results processing program which generates a trouble number

(e) Generate TTY messages indicating the diagnostic results

(f) Establish a working system configuration depending on the diagnostic results.

5.55 The peripheral equipment testing performed by the diagnostic programs consists of a fixed sequence of tests. The test results are compared with expected results or by monitoring strategic points within the unit.

Centrex and AIOD Diagnostics

5.56 Centrex diagnostic programs are designed to locate problems with centrex data links, consoles, and the AIOD equipment. These programs can send data to and accept key data from a centrex console and test whether the data is correctly sent and received. Diagnostics are run every night if the console is on night service, and may also be executed by TTY request, or if centrex data errors are detected. The AIOD diagnostic program can test an AIOD receiver and associated automatic number identification (ANI) circuits. It is normally executed only upon TTY request, but may also be entered if an AIOD fault is recognized.

Peripheral Unit Exercise Programs

5.57 The peripheral unit exercise programs are also administered by MACR on a base level, low priority basis to:

(a) Check for faults that might otherwise go undetected

(b) Supplement trouble detection circuits

(c) Check trouble detection circuits

(d) Exercise infrequently used hardware.

5.58 Exercise programs are classified in the following categories:

(a) Scheduled automatic routine exercises which are run at prescheduled intervals of time

(b) Demand exercises which are run at the request of another program or by maintenance personnel via the TTY.

5.59 Diagnostic test results may be processed by the dictionary trouble number program (DOCT). The resulting trouble number can be used to index a suspect list of faulty equipment in the trouble locating manual, as described below.

DIAGNOSTIC RESULTS POST-PROCESSING SOFTWARE

A. ESS Switch Peripheral

5.60 The maintenance personnel in an ESS switch office use TLMs to interpret diagnostic printouts from the TTY. These TLMs provide an ordered list of 12-digit decimal numbers with one or more circuit packs associated with each number. After the diagnostic programs have diagnosed a faulty unit, the DOCT program reduces the raw data of the diagnostic results to a 12-digit decimal number. This number, referred to as a trouble number, appears as part of the TTY output message identifying the system component and unit. The maintenance personnel can then select the proper TLM according to the system and unit specified by the printout, and look up the trouble number. The circuit packs associated with the given trouble number may then be determined.

5.61 An additional set of phase trouble numbers will also be printed out with the overall trouble number. These numbers provide a backup, although more difficult, means of fault resolution and will only be used if the procedure designed for the overall trouble number is unsuccessful.

5.62 Each diagnostic program independently translating its results into a trouble number would be redundant; therefore, the subroutine DOCT is provided. In addition to trouble number generation, the subroutine DOCT also allows its clients, the diagnostic programs, to request some special services (eg, raw data may be printed for examination by the maintenance personnel). Special patterns of apparent failures may be interpreted as All Tests Pass and special actions taken to compute trouble numbers in cases where a unit cannot be fully tested because an associated bus is out of service.

5.63 ***Number Generation:*** Raw data diagnostic results may be over 5000 bits long. The reduction of this long number to a 12-digit decimal number is accomplished in two stages. The first is called bin loading; it reduces the raw data to a number that can be stored in 20 or less call store locations. The second stage is a set of three scrambling routines that operate on the bin of 20 (or less) locations to produce three

4-digit decimal numbers. The set of three 4-digit numbers is the trouble number.

5.64 ***Phase Number Generation:*** The diagnostic tests are made up of groups of tests called phases. The phase number generation is done in the same manner as the number generation (but only each failing phase of data is used), and a 12-digit trouble number is produced for the phase.

5.65 ***Bin Loading:*** The bin loading routine searches the raw data for failures (1s in a field of 0s). When it finds a failure, it computes its position in the overall string of raw data. It now has a number, the address of the failure. The address of the first test of the first phase of a diagnosis is 0. In the beginning, the answer bin (which will be the input to the scrambling routine) contains all 0s. The bin loading routine examines the address of the first failure. If bit 0 of the address is 1, the routine increments the first location of the bin. If not, it increments the eleventh. If bit 1 is a 1, it increments the second bin location; if not, it increments the twelfth. The routine continues in the above manner until all ten pairs of bin locations have been properly adjusted. It then finds the next 1 in the raw data and repeats the bin loading operation. In the calculation of phase trouble numbers, an additional bin is used (PHBN). At the end of each failing phase, the bin is scrambled and the resulting 12-digit number is printed under the appropriate phase heading.

5.66 ***Scrambling:*** At the end of each phase (of phase number generations) and at the end of the last phase (for overall number generation), the bin is delivered to the scrambling routine, which performs a series of additions of successive bin locations, with a rotate operation performed on the sum after each addition. The summing is done three times, each with a different rotation of the partial sums. The three results are the final trouble number.

5.67 These processing routines are simply a means of transforming large binary diagnostic results into small decimal numbers in such a way as to minimize the probability that two different diagnostic results will yield the same 12-digit decimal number.

5.68 The DOCT also performs the following special services for clients when so requested:

(a) Prints raw data results

(b) Generates special trouble numbers when communication buses are out of service

(c) Permits diagnostic restarts

(d) Checks for special all tests pass situations

(e) Provides extra scratch memory

(f) Generates 10-cell trouble numbers.

B. 1A Processor

5.69 The processing of 1A processor diagnostic results is performed by the DRPP. The purpose of DRPP is to provide an on-line facility which provides a human interface for each of several methods of automatic raw data analysis, called automatic trouble locating procedures. The main output of this process is an ordered list of suspected faulty equipment locations, more commonly called the "pack list." This pack list is the first line maintenance tool available to the craftsperson to aid in the repair of faulty frames.

5.70 The DRPP, more commonly called the trouble locating procedure (TLP) program, is the collection of diagnostic results post-processing programs including both the common DRPP programs and the frame dependent interface programs.

5.71 The common DRPP programs contain those routines which are applicable to both 1A processor and application system TLP processing. Some of these routines are required to be main memory resident; whereas, those which have a low occupancy and high main memory requirement reside in file store.

5.72 Frame dependent interface programs (FDIPs) are unit dependent; ie, one exists for each applicable 1A processor and application system unit type. These all reside in file store.

5.73 All application dependent portions of the TLP program are contained in the application dependent FDIP pidents. Linkage to the application dependent FDIP program sections (PGSECTs) are handled by an application pident which receives control from the diagnostic control program (DCON) to perform all linkage-required processing.

5.74 The TLP program implements three TLP methods, each of which may apply for a specific group of unit tapes.

- The behavioral TLP

- The pattern analysis TLP
- The connectivity TLP.

Each TLP method processes the raw data in a different manner and the data base data used by each TLP is of a different format. Although the methods employed by the various TLPs may differ, the common program flow which controls the gathering of the raw data and the generation of the pack list is identical for all TLPs.

5.75 For diagnostics executing under DCON control and interfacing with the TLP program using the normal DCON interfaces, the typical generation of a pack list takes place in five general steps:

(1) Raw data collection (DCON interface function)

(2) Raw data analysis

(3) Locate and monitor

(4) Index generation

(5) Index translation.

The generation of the pack list proceeds as described below.

5.76 The diagnostic program applies tests to the suspected faulty frame and passes each diagnostic test result along with the expected test result to DCON. The DCON program compares the actual and expected test results to determine if the test has passed or failed. The DCON program passes any failing raw data to the TLP program and prints a message on the TTY indicating the tests that have failed.

5.77 The raw data passed by DCON is analyzed by the raw data processor portion of the TLP program, and the resultant summary data is placed in file store where it is held for later processing. This summary data is referred to as a TLPFILE and the collection of TLPFILEs is called the TLPQUEUE. An output message is printed on the TTY indicating the summary data that has been generated, and its disposition (ie, whether it was placed in the TLPQUEUE).

5.78 The file store queue is necessary to temporarily hold the summary data while the TLP data base tape is positioned to generate the pack list from the summary data. Depending on the speed of the 1A processor tape drives and on the relative position of the required data on the tape, the positioning process may take several minutes. Instead of holding up other diagnostics for this length of time, the TLPFILE is created and the diagnostic ends normally.

5.79 The data file used to generate the pack list from the summary data is on auxiliary data system tape referred to as the TLP data base tape. The data on the tape is organized in groups of blocks, each group containing all the data files required to process a pack list for a particular unit. Therefore the TLP data base tape must be positioned at the start of the group associated with the particular unit whose diagnostic produced the summary data.

5.80 This positioning process, called the locate step, is controlled by a portion of the TLP program which is resident in main memory. Initiation is by the raw data processor program requesting the initial 1-second MACP entry for this program prior to returning control to DCON for termination of diagnostic processing for the current diagnostic MACP client.

5.81 As the locate step is being stepped up, a TTY message is printed to indicate that the summary data is being processed. The locate steps, operating independent of any diagnostic MACP client, positions the TLP tape to the tape file required to process the summary data. When the tape has been positioned, the locate steps initiate a TLP MACP client to start the summary processor.

5.82 The summary processor uses the summary data in the TLPFILE and data from the TLP tape to generate and print the ordered pack list. All packs for a given unit on the TLP tape are referenced by a 12-bit index, so the generation of the pack list is a 2-step process. The first step, index generation, generates the list of indexes representing the faulty packs. The second step, index translation, then translates these indexes to produce the final pack list which is output on the TTY. Once the pack list has been successfully produced, the TLPFILE is removed from the TLPQUEUE and the next TLPFILE is processed.

5.83 The craftsperson may intervene in the TLP processing using TLP utility functions.

Through TTY input messages, access to the TLPQUEUE is possible to check status, clear one or all of the TLPFILEs from the TLPQUEUE, inhibit or allow the placement of data into the TLPQUEUE, manually generate a TLPFILE, or completely initialize the TLP program. The TLP functions also permit the craftsperson to switch the active and inhibited states of the TLP program.

TRUNK AND SERVICE CIRCUIT MAINTENANCE SOFTWARE

A. General

5.84 The overall objectives of trunk and service circuit maintenance programs are to remove faulty units from service, pinpoint detected troubles within a faulty unit, and alert maintenance personnel of trunk and service circuit status. The trunk and service circuit maintenance programs rely on several other subsystems to aid in performing the functions previously noted. (See Fig. 37.)

B. Trouble Detection

5.85 Trunk and service circuit maintenance action may be requested under the following circumstances:

- Any difficulty in establishing a customer's call

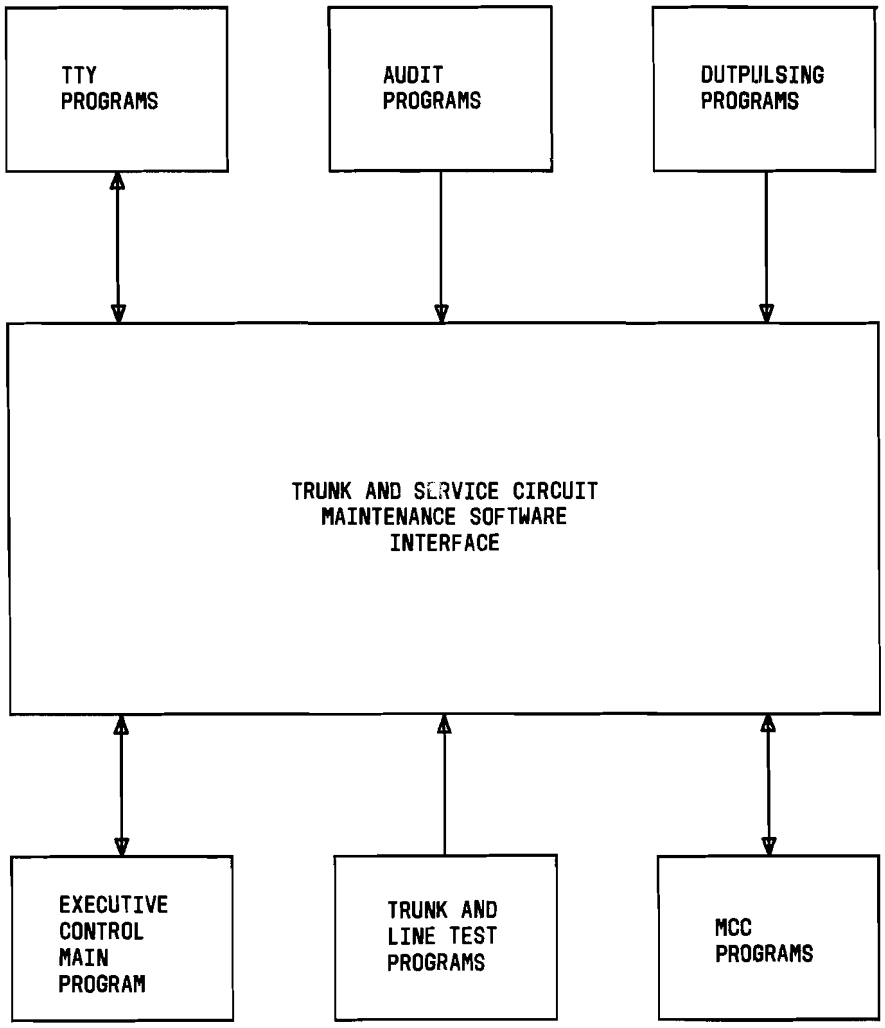

TTY PROGRAMS
AUDIT PROGRAMS
OUTPULSING PROGRAMS
TRUNK AND SERVICE CIRCUIT MAINTENANCE SOFTWARE INTERFACE
EXECUTIVE CONTROL MAIN PROGRAM
TRUNK AND LINE TEST PROGRAMS
MCC PROGRAMS

Fig. 37—Trunk and Service Circuit Maintenance Software Interface

- During automatic progression testing (APT)
- Manual request.

5.86 When problems arise during call processing activities, the trunks involved are placed on a maintenance list for testing. Some types of problems which may be detected with trunks and service circuit failures are:

- Incomplete outpulsing to a distant office
- Network continuity check failures
- Peripheral order buffer execution failures
- Transceiver time-outs
- Internal circuit check failures.

5.87 When many trunk and service circuit failures occur, the maintenance list may become full. In this case failing circuits are not tested at this time but are returned to service since a minimum number of trunks must be in service at all times.

5.88 Trunks on the maintenance list are tested using normal trunk diagnostics. If a trunk fails the test twice, a message is printed for the maintenance personnel and the trunk or service circuit is removed from service.

5.89 Besides testing these circuits when problems arise, trunk and service circuits are routinely tested on a flexible schedule during nonpeak hours. This automatic progression testing is done intermittently throughout the day, and a full cycle of testing is typically completed once a week. When faulty circuits are found, they are placed on the maintenance list and tested as previously described.

5.90 Also, when maintenance personnel deem it necessary, they may test individual circuits or groups of circuits. In this case, raw data may or may not be returned to the maintenance personnel for analysis.

C. Testing

5.91 Once a trunk or service circuit is placed on the maintenance list, it is tested via normal diagnostic procedures. The trunk and line test programs are used to request diagnosis of the circuits and place selected circuits on various lists for further processing.

5.92 The major functions of the diagnostics are fault detection and the generation of failure data used to locate faults. Diagnostics are comprised of routines that are driven by data tables. The diagnostic programs are resident on disk in No. 1A ESS switch and are paged into main memory when executed. The diagnostics initialize the trunk and service circuits into their various states and check their reaction to certain situations.

5.93 Error analysis programs are consulted to compare the present faults with a past history of faults to obtain a data base for future analysis. Data collected by error analysis programs include the kinds of faults encountered and how often they occur.

D. Trouble Reporting

5.94 Once a trunk or service circuit has been detected as faulty and diagnostic testing has pinpointed the errors, the maintenance personnel must be informed. Teletypewriter messages are printed to allow personnel to locate detected troubles. This allows the maintenance personnel to correct the faulty circuit and return the circuit to service. In all, the trunk and service circuit maintenance programs enable the ESS switch to have a minimum number of circuits unavailable or unfit for service.

TRUNK AND LINE TEST SOFTWARE

A. General

5.95 The trunk and line test software is comprised of programs which automatically conduct a sequence of tests on trunks and lines. These tests may be activated manually from a TTY and/or automatically by the No. 1/1A processor during routine maintenance activities. Messages to maintenance personnel are printed to inform them of the tests progress and results. Besides interfacing with TTY software, the trunk and line test programs work with several software subsystems to provide reliable trunk and line maintenance. (See Fig. 38.)

5.96 The trunk and line test software include the automatic line insulation test program, incoming trunk test program, station ringer test program, line termination denied program, and through

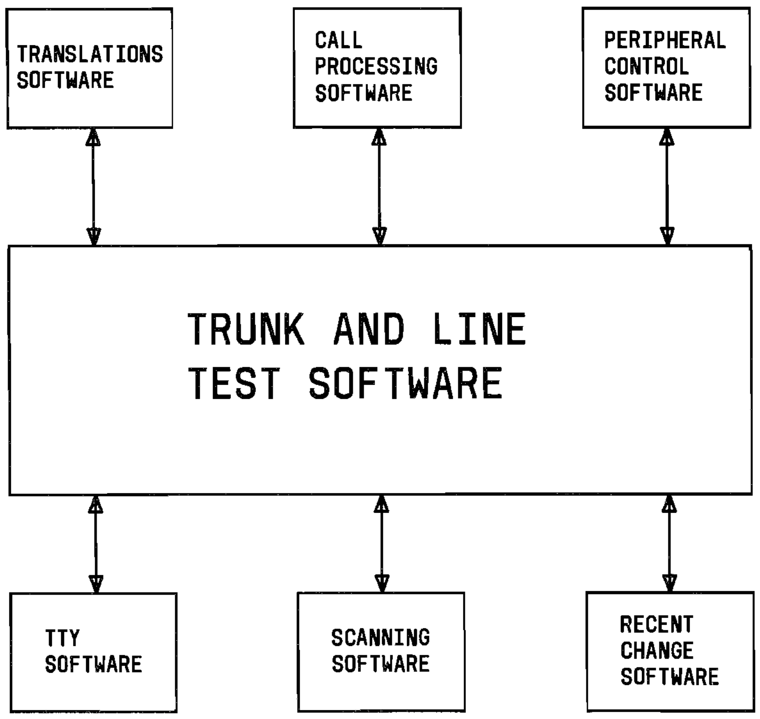


Fig. 38—Trunk and Line Test Software

balance test facility program. These programs and associated tests are discussed in the following paragraphs.

B. Automatic Line Insulation Test (ALIT)

5.97 The ALIT provides software to permit the automatic testing of line insulation values for all idle lines in an ESS switch office, except PBX lines.

5.98 Testing is initiated by requests from the MCC or may be started at a specific time and day by program control. ALIT includes instructions to enable various hardware tests to be performed with varying sensitivity. Sensitivity of the test is set to one of four ranges by adjusting the proper control digit. Other control digits are adjusted to select starting and stopping information and which tests to perform, if not all.

5.99 Normal progression testing begins with the lowest directory number in an office and proceeds through the office until the highest directory number is tested. An ALIT register is used to store the lowest directory number and the register is incremented as testing progresses.

5.100 When an idle line is found, a connection is made between the line and the ALIT test circuit. The tests are made and if successful the process is repeated on the next line. If a failure is encountered, TTY messages are printed to alert maintenance personnel and the testing is stopped.

C. Incoming Trunk Test Termination (ITTT)

5.101 Upon demand from a distant office, this program provides terminations for incoming trunk testing. An ITTT is responsible for all actions required by the far-end office to complete the test. These actions include making the appropriate connections through the office to the proper test circuit, placing the trunk under test into the proper state for testing, providing test signals over the trunk under test to the originating office, and initiating supervision for the length of the call by transferring control to proper supervisory routines.

5.102 An ITTT provides test terminations for eight incoming trunk tests:

Code test 100, 102, 103, 104, 105, 107, 108 charge test, and synchronous line test.

D. Station Ringer Test (SRTT)

5.103 The SRTT program provides the software necessary to allow maintenance personnel to perform three tests from the customer's premise. These are:

- TOUCH-TONE dialing test
- Party ground test
- Ringer test.

5.104 The SRTT program is responsible for four functions in relation to these tests. These are:

(a) Connect the calling line to a station ringer test circuit

(b) Test the TOUCH-TONE dialing and signal the result

(c) Check and indicate ground identification

(d) Connect the line to a ringing circuit to test the station.

5.105 The SRTT program is activated when the ESS switch receives a predetermined code. The SRTT connects the line to a TOUCH-TONE service test receiver via the station ringer test circuit and returns dial tone. Upon receipt of the second dial tone, the maintenance personnel dials a sequence of digits, used by SRTT to test the TOUCH-TONE service facility. The SRTT returns a double burst of high tone for a successful test and a single burst for a failure.

5.106 Any time after receipt of the second dial tone, maintenance personnel may initiate the party ground test by flashing the switchhook. The SRTT then tests the ground connections and returns coded signals to identify the type of ground connection found. After flashing, the ringer may be tested by going on-hook. SRTT will connect the line to a ringing circuit to apply ringing current. If a problem is present with the station on-hook, a coded signal is applied instead of ringing current.

E. Through Balance Test Facility (TBTF)

5.107 This test activity is initiated by a TTY request message. The TBTF checks to see if the trunk specified by the message has been cleared for a through balance test. If it has, the state of the trunk is determined. If the trunk is busy the test is abandoned and an output message is printed. If the trunk is idle, a search is made for idle test facilities. When a test circuit becomes available, TBTF initiates a connection between the test circuitry and the trunk under test.

5.108 Once the connection has been successfully set up, the power must be removed to allow maintenance personnel to insert a test tool. If power is not removed, TBTF will not allow the test.

5.109 Power is restored when the test tool is in place and ready for testing. The TBTF places the trunk under test in the proper testing state and makes the final test connections.

5.110 After the tests are completed, maintenance personnel may request a test on the next consecutive trunk. The TBTF will take appropriate actions to initiate the test and the testing procedure will be repeated.

F. Line Termination Denied Program (PLUG)

5.111 When a subscriber's line has been denied termination, calls to the line are routed to a trouble intercept. The PLUG program processes all requests to initiate or cancel this condition.

5.112 Requests to deny line termination may only be initiated manually from the recent change TTY. A recent change message is used to change the line equipment number and directory number translation words to reflect the denied service condition and to reroute incoming calls.

5.113 When a fault is found in a subscriber line, it may be desirable to deny terminations to that line. When an idle line is found faulty, PLUG will place it on a line termination denied list. When a busy line is found faulty, PLUG will place the line on a high and wet list and indicate its condition.

5.114 Canceling the line termination denied condition can be done three ways. When a line that has been denied termination originates, PLUG takes action to remove the line from the list. (The ability to originate indicates that the fault has been cleared.) Also this condition can be cleared via a recent change message. Furthermore, when a line, which was previously placed on the high and wet list, is found to be in good condition, the permanent signal partial dial programs request PLUG to grant the line terminating ability.

DATA MAPPING SOFTWARE

A. General

5.115 This section describes the functional operation of the data mapping control and linking program (DMAPAPPL). This program is responsible for the movement and preservation of data structures containing variable data in the No. 1A ESS switch application of the 1A processor during a system update.

5.116 The major software subsystems with which the data mapping program interfaces (Fig. 39) are as follows:

(a) File Store Administration Program: This program contains standard routines which are used by the data mapping program for all disk (file store) input/output requests.

(b) Input/Output Control Program: This program contains standard routines which are used by data mapping to receive TTY input from maintenance personnel and to print status and error messages encountered during the update (mapping).

(c) Maintenance Control Program: Since the data mapping program is a client of the peripheral maintenance control program, routines in maintenance control are used by data mapping for segmenting the data mapping operation into appropriate time slots.

(d) Audit Programs: The translation system audit is used by data mapping to compress and map the temporary recent change area in core. The system audit program is used to reduce register-carried stable calls to a nonregister state.

B. Functional Description

5.117 The introduction of a new generic or new parameters into an existing ESS switch office requires, in most cases, a rearrangement of the data structures which are resident in the call store, program store, and file store. This rearrangement normally involves relocating existing structures from their current addresses to new addresses in the same or in different stores.

5.118 The data structures which are resident in the program store, simplex call store, and file store normally contain the fixed data that is included as part of the update. These data structures are preserved directly by the system update program. No additional action is required by any other program (including the data mapping program) in the retention of these data structures during and after an update.

5.119 Call dependent data and other variable data structures contained in the duplicated call store which must also be preserved during an update include:

- Path memory information needed to maintain an existing call in the talking state
- Temporary recent change data
- Trunk maintenance lists.

5.120 The core copy of the temporary recent change area is mapped by a subroutine in the translation system audit program during core-to-core mapping. The disk backup copy is updated after the memory reinitialization phase is run following the update. Data structures to be mapped are:

- Temporary Recent Changes
- Path Memory for Lines
- Trunk-to-Trunk Memory
- Path Memory for Trunks
- Traffic Scratch
- General Purpose Traffic Accumulators
- Day, Month, and Year Counter
- Traffic Counters
- Plant Measurements Daily Counter
- Input/Output Status Tables
- Central Pulse Distributor Status Tables
- Peripheral Unit Bus Status Tables

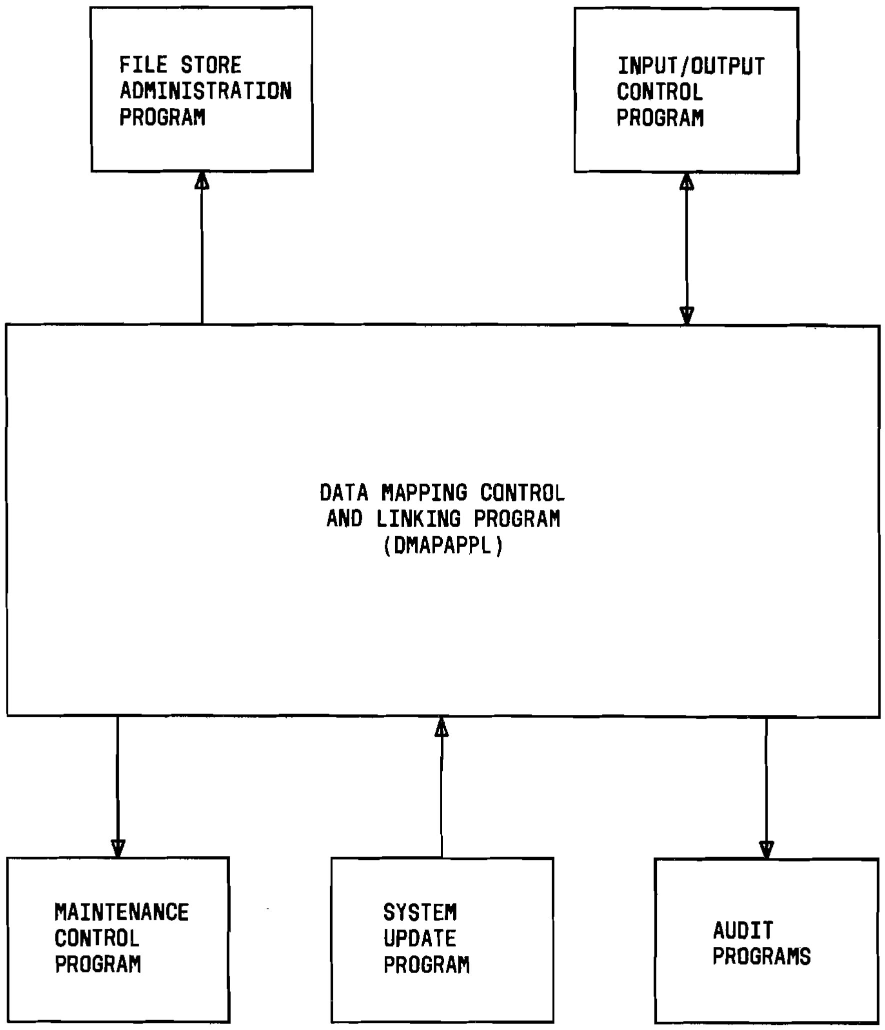

FILE STORE ADMINISTRATION PROGRAM
INPUT/OUTPUT CONTROL PROGRAM
DATA MAPPING CONTROL AND LINKING PROGRAM (DMAPAPPL)
MAINTENANCE CONTROL PROGRAM
SYSTEM UPDATE PROGRAM
AUDIT PROGRAMS

Fig. 39—Data Mapping Interfaces With Other Subsystems

- Scanner Answer Bus Status Tables
- Central Pulse Distributor Bus Status Table
- Pointers to the Recent Change Rollback Area.

5.121 These data cannot be saved by a common program, such as the system update program, since the identity of these data structures depends largely on the following:

(a) The application of the 1A processor

(b) The nature of the generic programs involved in the update

(c) The nature of the memory recovery phase run after the update.

5.122 In the No. 1A ESS switch, scheduled changes to the generic program or the office parameters are introduced to an existing office by means of a full or a partial update. A full update implies that all data in either or both program stores and the parameter area of unduplicated call store are being replaced with new system data. A partial update will result in only selected data in the parameter or generic areas being replaced with new data.

5.123 The data mapping program is concerned with that segment of the system update process which involves the preservation of transient call store data.

Order of System Update

5.124 The steps followed by the system update and data mapping programs during a full update are as follows:

(1) Copy the tape(s) containing the new generic and/or office parameter data to a file store community that was manually configured for update.

(2) Verify that all call store K-codes to be duplicated by the new copy of the parameter data are in fact duplicated.

(3) Map file store data with a new copy of the data mapping program.

(4) Map transient call store data.

(5) Run a phase 4 to initiate the periphery.

(6) Begin call processing with the new generic and/or parameter data.

5.125 The above order of events involved in a system update minimizes the complexity of recovery from an interrupt during the update process. This procedure also has the advantage of having no loss in call processing if a mapping failure occurs prior to Step 4. A failure after Step 4 may lead to the loss of both transient and talking state calls.

Update Procedure

5.126 The process for a generic update (Fig. 40) begins with maintenance personnel securing an available tape unit controller for the update function. The tape unit controller should be set for read only by inputting the appropriate TTY message.

5.127 A rover program store is then seized and configured for use as a buffer area for data mapping and system update by the appropriate input message. (Since maintaining a dedicated area of core for infrequent use is uneconomical, a rover program store is utilized.)

5.128 A file store community is next selected for update by operating the appropriate file store key on the MCC common panel. This action will prevent the use of that file store community by programs other than those involved in the update process.

5.129 The contents of the update tape is then copied into the selected file store community by a TTY input message. At this point, maintenance personnel can request, via the TTY, to bypass the data mapping segment of the update (system default is to perform data mapping).

5.130 Following the integration of the file store with the data from the update tape, the system update program will print a message indicating that the file store tape is completed. A new tape or the update of core should be started.

5.131 When all update tapes have been inputted to the file store community, the system update program prepares the following new system maps:

(a) Core to disk map

(b) Identification (id) tag to file store map.

5.132 The old and new system maps are checked for ID overlaps and for disk descriptor block consistency. A failure in any of these checks will result in a request to either provide additional tape input or to abort the update. A successful pass in each of the above checks and the absence of a TTY request to inhibit data mapping results in control being transferred to the disk mapping program to perform file store mapping and to build tables for core mapping.

5.133 The old control routine in the disk mapping program locates, from the system maps prepared by the system update program, the new copy of the disk mapping program and the new copy of the parameter data assembler (PDA) on the updated file store. The new disk mapping program and PDA are then pumped into the rover program store that has been configured for use as a scratch area.

5.134 After the pump of the new data mapping program and PDA into the rover program store, the old data mapping program then transfers program control to the new data mapping program to map the necessary file store structures. File store mapping consists of the copying of the required data from their locations on the nonupdated file store to

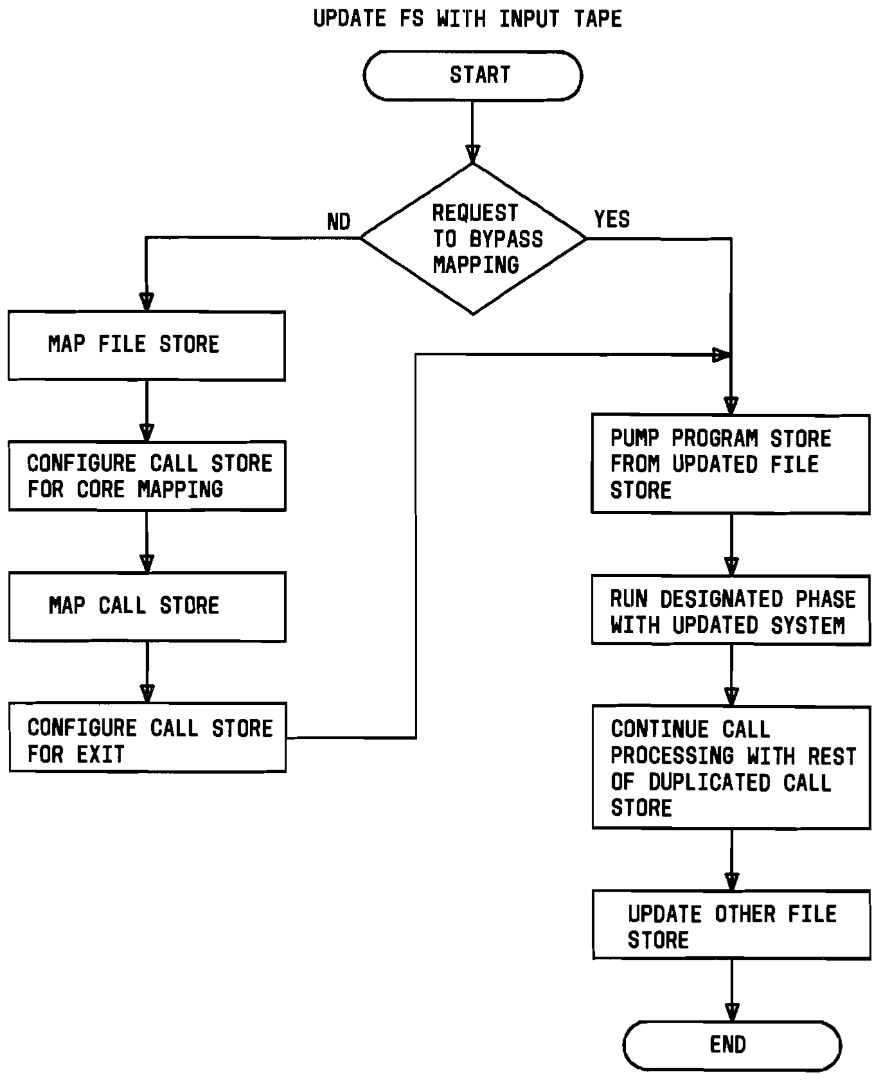

UPDATE FS WITH INPUT TAPE
START
REQUEST TO BYPASS MAPPING
NO
YES
MAP FILE STORE
CONFIGURE CALL STORE FOR CORE MAPPING
MAP CALL STORE
CONFIGURE CALL STORE FOR EXIT
PUMP PROGRAM STORE FROM UPDATED FILE STORE
RUN DESIGNATED PHASE WITH UPDATED SYSTEM
CONTINUE CALL PROCESSING WITH REST OF DUPLICATED CALL STORE
UPDATE OTHER FILE STORE
END

Fig. 40—Functional Flowchart of Update Procedure With Emphasis on Mapping

a buffer area in the rover program store. The data is then modified, as required, and copied to the updated file store.

5.135 The data mapping program returns program control to the system update program upon completing file store mapping which then sets up the system for core mapping by splitting the duplicated call stores into two halves.

Configuration of Duplicated Call Store

5.136 The duplicated call stores on the active bus are maintained in the normal mode while those on the standby bus are reconfigured to the active bus, set in the maintenance mode, and initialized to zero.

5.137 The set of duplicated call stores that are now in the normal mode can be accessed by normal reads and writes while those in the maintenance mode can only be accessed by maintenance orders. This will enable the data mapping program to address its reads and writes to particular member numbers in the duplicated call store community.

5.138 H-level and J-level interrupts are now inhibited in order to prevent call processing programs from updating transient data structures concurrently with the mapping of the same structures. G-level interrupts are also inhibited to prevent routine maintenance programs and the generic utility program from either updating transient data concurrent with mapping or the triggering of recovery actions which may also abort the update.

Actions by Old Data Mapping Program

5.139 The system update program will transfer program control to the old data mapping program upon the completion of system preparation. The old data mapping program when entered at this time will perform the following functions:

(a) Unpest J-level interrupts and set the J-level modify instruction address so that a count of J-level interrupts may be made to ensure that the B-level timer may be reset before it expires. The data mapping program counts a predetermined number of J-level interrupts before it transfers control to the system update program to reset the long timer. If the data mapping program fails to transfer to the system update program in time to reset the timer, the timer will expire, triggering an automatic processor configuration which will abort the update.

(b) Transfer program control to the system audit programs to reduce register-carried stable calls to a nonregister state in order that they may be saved by mapping path memory. These types of calls include hotel/motel, temporary transfers, and simulated facilities.

Transfer to New Data Mapping Program

5.140 Transfer of program control is next made to the new data mapping program in the rover program store to map transient data. The old and the new data mapping programs are then scanned to determine which structures to map and which structures to skip. If mapping is required, a transfer is made to the appropriate mapping algorithm for the actual mapping.

5.141 Program control is passed at the completion of core mapping from the new data mapping program to the system update program to pump the program stores, translation stores, and call store 0. The original standby stores in the maintenance mode which now contain the mapped data are made active and the originally active stores are taken out of service before the pump. The phase which is specified on the tape header of the update tape is next run to complete the recovery. At some time after this phase is run, the call store fault recognition programs are called on a time-shared basis to reconfigure a duplicated call store community.

SYSTEM PERFORMANCE SOFTWARE

A. General

5.142 The system performance software consists of program operations designed to indicate the ability of the system to perform its call processing functions.

B. System Performance Software Interface

5.143 The interface between the system performance software and other ESS switch software systems is illustrated in Fig. 41.

5.144 The system operational control software provides scheduled entries to the system performance software. The system performance indicator program (SYPI) is entered every 10 seconds to collect data and perform tests on the displayed system performance indicator located on the No. 1A system status panel. The dial tone speed test program (DTST) is entered every 4 seconds only if the DTST feature has been activated either manually or automatically. An entry is made every 4 seconds to the RADR program to test the delay in receiver attachment.

5.145 All of the system performance programs provide interface with the TTY software for manual interfacing with the TTY.

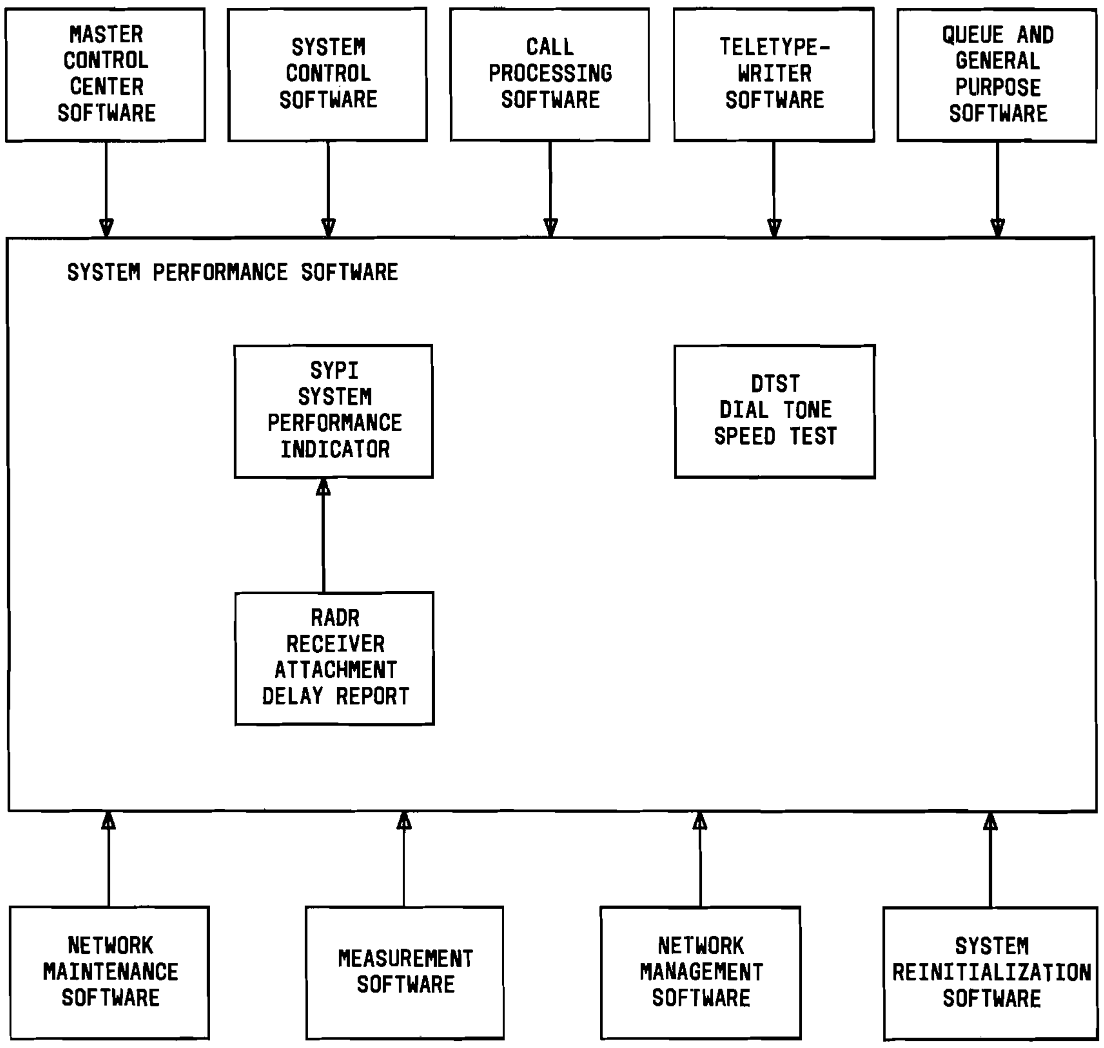
C. System Performance Software Function

System Performance Indicator Program (SYPI)

5.146 The system performance indicators are located on the No. 1A system status panel. The purpose of these is to visually alert the craftsperson that the system processing capabilities are decreasing. The information provided by SYPI allows the craftsperson to take corrective measures before the system performance deteriorates to a lower level.

5.147 The status of the following resources are displayed by TEST STATUS lamps:

- Call Registers
- Hoppers


MASTER CONTROL CENTER SOFTWARE
SYSTEM CONTROL SOFTWARE
CALL PROCESSING SOFTWARE
TELETYPE-WRITER SOFTWARE
QUEUE AND GENERAL PURPOSE SOFTWARE
SYSTEM PERFORMANCE SOFTWARE
SYPI SYSTEM PERFORMANCE INDICATOR
DTST DIAL TONE SPEED TEST
RADR RECEIVER ATTACHMENT DELAY REPORT
NETWORK MAINTENANCE SOFTWARE
MEASUREMENT SOFTWARE
NETWORK MANAGEMENT SOFTWARE
SYSTEM REINITIALIZATION SOFTWARE

➧Fig. 41—System Performance Software Interface➧

- Queues
- Service Circuits
- Line and Network
- Data Transfer
- Abrupt Traffic Change.

5.148 The SYPI program applies a statistical test to call registers based on holding time. The average holding time is defined as the ratio of the sum of the active elements to the number of seizures. A moving average and a moving variance are computed based on the average holding time. If the sample is very small, no statistical analysis is performed. A safety factor of 4 is normally applied if the sample is sufficiently large. If the value falls beyond the safety value, the test is considered a failure and the lamp is lighted amber. Also, whenever there are requests for the facility but no seizure is made, the lamp will be lighted amber.

5.149 The lines and networks are checked for excessive failures by SYPI. The check periodically computes the ratio of the weighted sum of the faults constituting the particular failure type to the total traffic per network observed during the interval. If this ratio exceeds a predefined threshold, then the TEST STATUS—LINE AND NETWORK lamp is lighted amber.

5.150 Statistical tests are applied periodically in order to alert the craftsperson via the panel status lamps whenever an excessive number of failures occur.

5.151 The statistical tests for line and network are done once every minute. Based upon this criteria, the TEST STATUS—LINE AND NETWORK lamp is lighted amber whenever an excessive number of failures, in comparison to the traffic, occurs in the system. The lamp is lighted green only if the statistical checks are successful.

5.152 The TEST STATUS—DATA TRANSFER lamp reflects results of the test on the disk and TTY equipment. The tests are executed once every 10 seconds by SYPI.

5.153 A statistical test is performed to verify the availability of the OMRs. A failure is recorded if there is either an excessive number of OMR seizure failures or a lack of idle OMRs. In either case, SYPI requests the appropriate TTY audit (no more than once every 5 minutes).

5.154 The TEST STATUS—DATA TRANSFER lamp is lighted amber if any of the preceding tests fail. The lamp is lighted green only if all tests pass continuously a fixed number of times.

5.155 Every 10 seconds SYPI applies a statistical test on certain traffic conditions. The test is similar to that applied to call registers and service circuits, differing in precision maintained in calculations. If the value falls beyond the given tolerance interval, the TEST STATUS—ABRUPT TRAFFIC CHANGE lamp is lighted amber.

5.156 Light-emitting diodes (LEDs) on the panel display system activity associated with:

- Originating calls
- Incoming calls
- Intraoffice calls
- Tandem calls
- Processor occupancy
- Manual selection (miscellaneous).

5.157 In general, the number of LEDs lighted green indicate the value of the item being displayed as a percent of the threshold (100 percent) value. If the value of the item falls below 10 percent of the threshold value, the column identification lamp at the bottom of the bar graph is lighted white to indicate a change in scale. The change implies that the percent reading for that bar graph must be multiplied by 0.1 to obtain the true reading. For these low values, each LED represents increments of 1 percent. If, on the other hand, the item being displayed exceeds the threshold value, the LED at the top of the bar graph is lighted red in addition to the 10 green lighted LEDs.

5.158 Bar graphs, ORIG. CALLS, INCOMING CALLS, INTRA OFF CALLS, and TANDEM CALLS are updated only once each minute. The PROCESSOR OCCUPANCY bar graph is updated once every 10 seconds.

5.159 The PROCESSOR OCCUPANCY bar graph value is calculated by system operational control software once every 10 seconds. This value is the percent of real time spent by the processor on nonrelinquishable and call processing functions. The bar graph reflects the time spent on accomplishing the basic objective of the system, which is to process calls.

5.160 The MANUAL SELECTION bar graph is normally not lighted.

Dial Tone Speed Test Program (DTST)

5.161 The purpose of the DTST program is to provide measurement of dial tone service. It applies a test to measure the interval of time separating customer off-hook from receipt of dial tone. If dial tone is furnished within 3 seconds, a success is recorded. If the 3-second test was a failure and at the end of a 11-second period dial tone has not been furnished, the test is scored as an extended failure.

> ***Note:*** The dial tone speed test function is activated by a TTY input message.

Receiver Attachment Delay Report (RADR) Program

5.162 The RADR program generates, maintains, and administers incoming test calls designed to measure the amount of delay incoming calls are experiencing in getting a connection to a receiver.

Activation and deactivation are via TTY input messages.

5.163 Every 4 seconds, the RADR feature is activated. A test call is generated on a randomly selected incoming trunk. Attempts to establish a connection which take more than 3 seconds between the incoming test call and a receiver are termed delays.

5.164 The program is entered every 4 seconds from system operational control software to determine test results of the previous test call, if one occurred. A check is first made to see if RADR has been inhibited. If so, control is returned to system operational control software; otherwise, another check is made to determine if a test was performed during the last 4-second interval. If so, traffic counts associated with the test call are administered. After processing the test results, including the setting of the TOC02 flag if the threshold value was exceeded, control is returned to generate a trunk network number for the next RADR test.

5.165 Entry is made every 100 ms from system operational control software to generate a suitable trunk network number. If a satisfactory trunk network number is not found or until the 4-second entry to simulate the seizure occurs, a total of 120 trunk network numbers are considered before a test call is skipped.

5.166 Every 30 seconds an entry is made from system operational control software to update the signal distributor points on the network management indicator circuit for RADR failure percentage display.

5.167 Hourly, an audit of the up-down failure counters and result registers is performed. Only those registers of valid receiver types are audited. The purpose of this audit routine guarantees that the up-down counter containing the number of test failures in the last 50 tests are correct.

5.168 Every 5 minutes, entry is made from the network management program to determine if any threshold values have been exceeded in the last 5-minute interval. If so, a request is made for printing an NM17 output message. The exception indicators are zeroed and control is returned to the main program.

MASTER CONTROL CENTER SOFTWARE

A. General

5.169 The master control center software contains programs which administer the input and output functions associated with the following master control center control and display panels:

- 1A Control and Display Panel
- Trunk and Line Test Panel
- Supplementary Trunk Test Panel
- Auxiliary Test Frame
- Manual Test Trunk Panel.

5.170 The master control center group of programs monitors the state of keys, lamps, and switches via flip-flops and scan points in the processor peripheral interface frame. These flip-flops and scan points are arranged in two matrices: the control and display matrix, and the scanner and signal distributor matrix. The recognition of a state change results in the reporting of the change to the proper client program that performs MCC-associated display action.

B. Master Control Center Software Interface

5.171 Manual actions performed by the craftsperson at the MCC and the control panels are detected by the common control and monitor program (MCCM) via the matrices. The state change information is reported to the appropriate program.

5.172 The software interface between the master control center software and other system software is depicted in Fig. 42.

5.173 The system control software enters the MCCM program every 100 ms to verify switch changes and perform 1-second scans. This entry also controls the 1-minute work, if any, such as flashing lamps and printing messages pertaining to power status.

5.174 The MCCM program is entered from the fault recovery software when:

(a) Out-of-service lamp is to be lighted

(b) Out-of-service lamp is to be extinguished

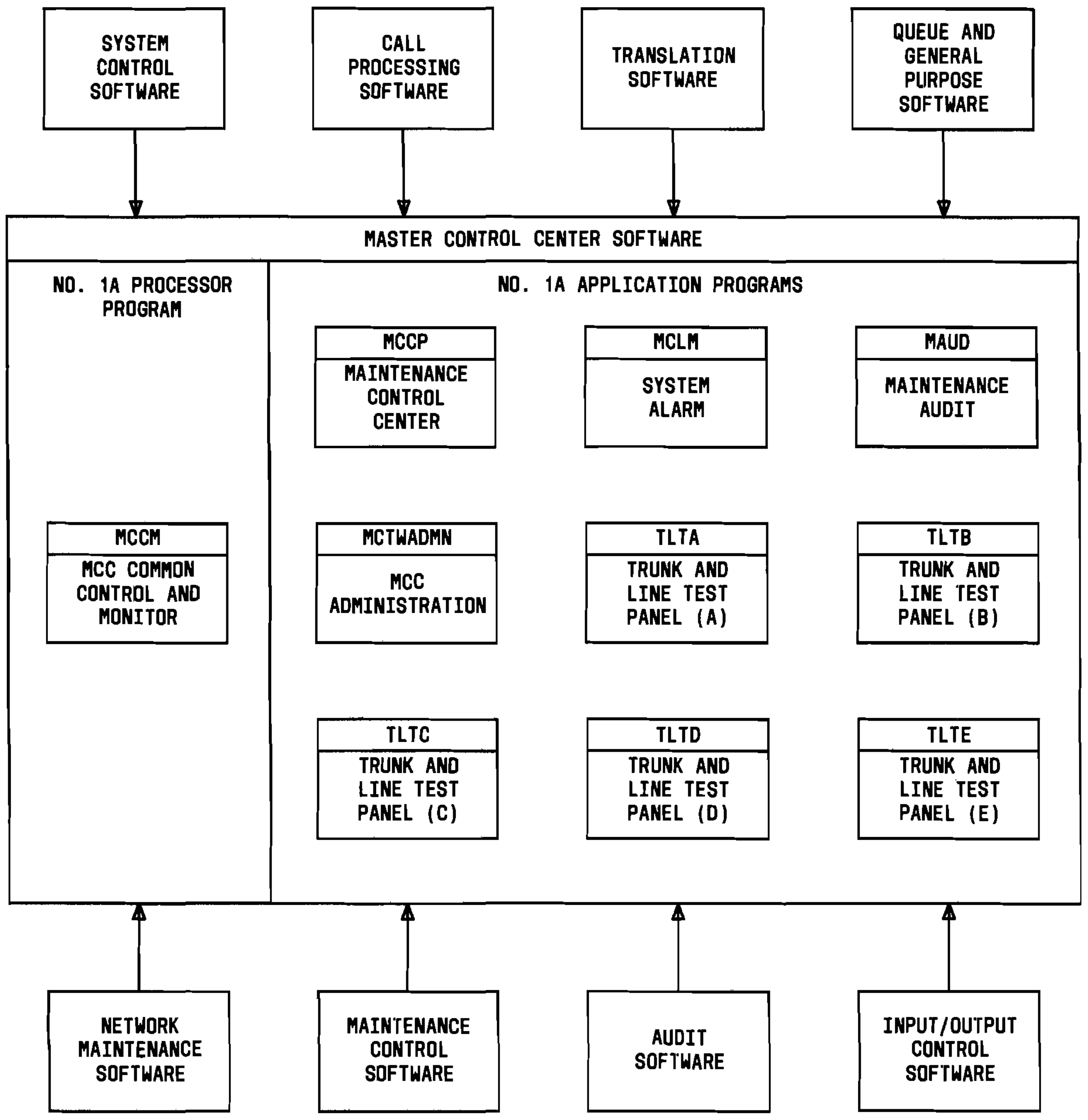

SYSTEM CONTROL SOFTWARE
CALL PROCESSING SOFTWARE
TRANSLATION SOFTWARE
QUEUE AND GENERAL PURPOSE SOFTWARE
MASTER CONTROL CENTER SOFTWARE
NO. 1A PROCESSOR PROGRAM
NO. 1A APPLICATION PROGRAMS
MCCP
MAINTENANCE CONTROL CENTER
MCLM
SYSTEM ALARM
MAUD
MAINTENANCE AUDIT
MCCM
MCC COMMON CONTROL AND MONITOR
MCTWADMN
MCC ADMINISTRATION
TLTA
TRUNK AND LINE TEST PANEL (A)
TLTB
TRUNK AND LINE TEST PANEL (B)
TLTC
TRUNK AND LINE TEST PANEL (C)
TLTD
TRUNK AND LINE TEST PANEL (D)
TLTE
TRUNK AND LINE TEST PANEL (E)
NETWORK MAINTENANCE SOFTWARE
MAINTENANCE CONTROL SOFTWARE
AUDIT SOFTWARE
INPUT/OUTPUT CONTROL SOFTWARE

Fig. 42—Master Control Center Software Interface

(c) Unit is to be restored.

5.175 The MCCM program interfaces with the application programs via the process application transfer table. The MCCM program interfaces the application programs to obtain the following information:

- Overload data
- Translations data
- Scan point data.

5.176 The system control software enters the maintenance control center program (MCCP) every second for updating the lamps on the panels.

5.177 The call processing software is interfaced to perform the routines which will process the test call, such as digit analysis and ringing.

5.178 The translation software is interfaced to provide translations on the test call requested at the panel.

5.179 The queue and general purpose software is interfaced by the master control software whenever a change of state is detected on the test panel key.

5.180 Whenever a failure is detected on a signal distributor by the network maintenance software, the master control software is interfaced to provide a TTY printout.

5.181 The audit software interfaces with the master control software to provide scheduled audits.

C. Master Control Center Software Function

5.182 The MCCM program is organized into two functions for the administration of key requests, lighting and extinguishing of lamps, and operation of switches. These functions are initiated once every 100 ms by the system control software. The operation of keys on the master control console panels are detected by a periodic scan of the control and display matrix. The recognition of a state change is reported to the client program.

5.183 Scan points and signal distributor points are periodically scanned for state of changes in switches. If fault recovery action is required, the change of state is reported to the appropriate fault recovery program. The MCCM program administers the lighting and extinguishing of the switches and out-of-service lamps.

5.184 The MCCM provides TTY output messages when there is a power alarm, power off, and power restoral.

5.185 When a diagnostic has been completed by the fault recovery software, MCCM is reentered to determine if the unit can be restored. If the diagnostic is not completed within 7 minutes, MCCM extinguishes the ACK lamp on the panel and prints a time-out message to inform the craftsperson that the acknowledgment has timed out but the request is still in progress.

5.186 The MCC lamp routines detect changes in the control and display matrix and update the corresponding item in memory. After memory is updated and if the MCC is in service, the data is set up for the peripheral order. The order is then sent to the client program to perform the lighting function.

5.187 When a key change is detected by the MCCM periodic scan of the processor peripheral interface control and display matrix, a transfer is made to the client program to serve the key request. After the key request is processed, the client returns to MCCM, and the scan action is continued.

5.188 The MCCP program performs the administrative actions on the keys and lamps on the No. 1A system status panel. It provides TTY output messages which support the key and lamp operations.

5.189 System alarms are initiated and reported by the system alarm program (MALM).

5.190 The maintenance audit program (MAUD) performs audits which ensure that the supervisory programs report the changes of nontrunk ferrods.

5.191 The administrative functions performed by the MCC administration program (MCTWADMN) are:

(a) Service routines: Perform application function for the MCCM program.

(b) MCC control and display: Makes reports of status changes in the control and display matrix and processes certain key requests on the system status panel.

(c) TTY input message response: Responds to TTY input messages pertaining to program control flags associated with the system status panel.

GROWTH SOFTWARE

A. General Purpose

5.192 Within any given ESS switch central office, the wire center may have a unique set of engineered equipment. When the office grows, the amount of equipment changes. Since it would be expensive to rewrite the program every time an additional frame of equipment is added, the basic program (generic) is designed to treat all information about quantities of office equipment as data which can be changed as the office grows. Growth programs provide the required operations to change this information as the office grows. These programs maintain uninterrupted telephone service during relocation of new equipment and other system upgrading which either replaces or disables normally operating equipment. Growth programs also provide operations necessary during the cutover of a central office. Refer to Fig. 43 for an illustration of the interfacing of the growth software subsystems.

B. Network Growth Program

Primary Functions

5.193 The primary functions of the network growth program (NETG) are as follows.

(a) To respond to certain input messages from the maintenance TTY by making the specified switching network frame maintenance busy

(b) To respond to other TTY requests by restoring to service a specified network frame previously made busy

(c) To print out on the maintenance TTY hourly, or on demand, the identification and busy/idle status of those frames of the switching network which are affected by make-busy messages.

Principles of Operation

5.194 In response to a frame make-busy input message, NETG will busy-out all the network B

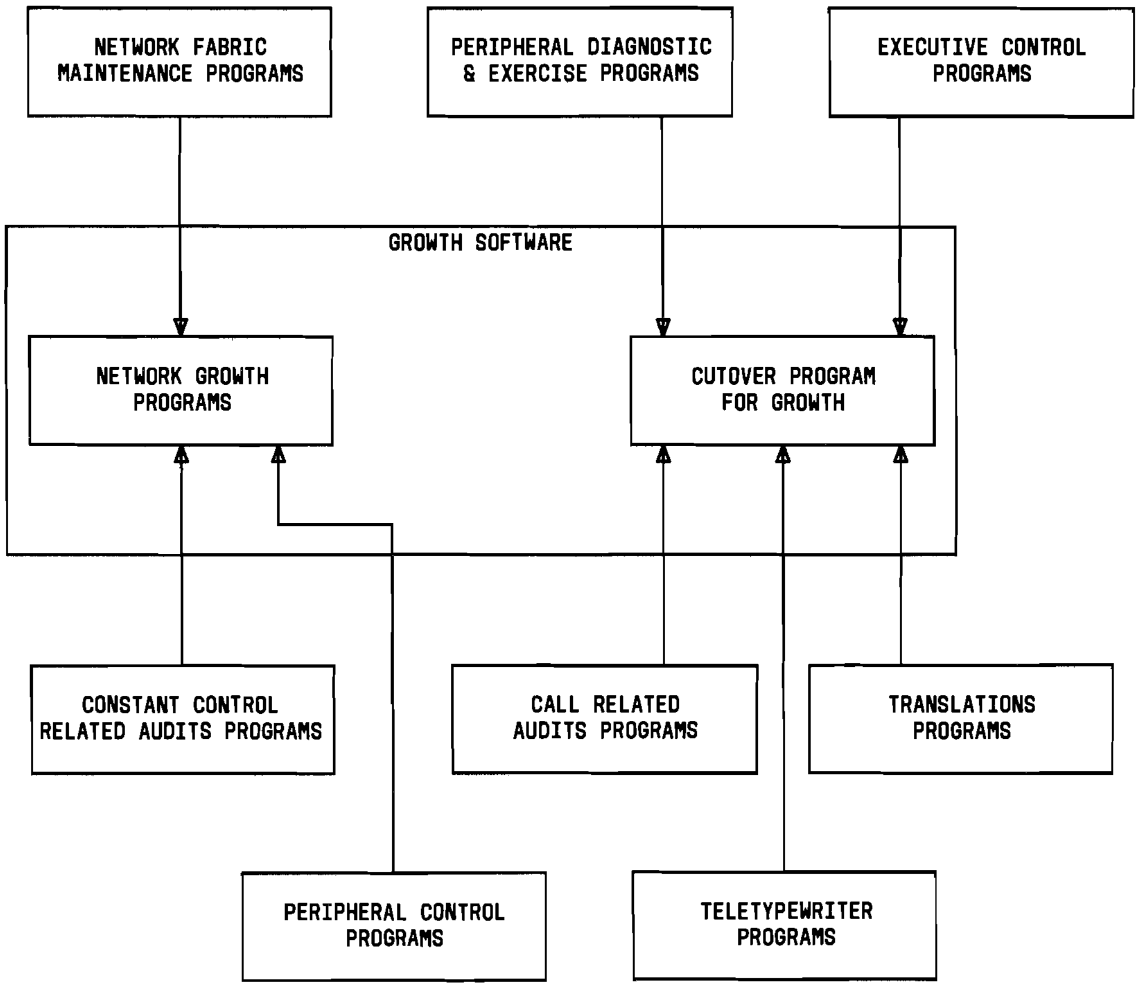

NETWORK FABRIC MAINTENANCE PROGRAMS
PERIPHERAL DIAGNOSTIC & EXERCISE PROGRAMS
EXECUTIVE CONTROL PROGRAMS
GROWTH SOFTWARE
NETWORK GROWTH PROGRAMS
CUTOVER PROGRAM FOR GROWTH
CONSTANT CONTROL RELATED AUDITS PROGRAMS
CALL RELATED AUDITS PROGRAMS
TRANSLATIONS PROGRAMS
PERIPHERAL CONTROL PROGRAMS
TELETYPEWRITER PROGRAMS

Fig. 43—Growth Software Subsystems Interface

links associated with the specified frame. When a TTY message is used to make busy a network frame, no network connections can be made through the specified frame after the input message has been accepted, although all calls already connected through the specified frame are not affected.

5.195 When a connection between any two network terminals is to be established, the network hunt program (peripheral control) examines a flag word which is set when any frame in the office is maintenance busy. If the flag is not set, the patch can be hunted in the normal way. However, if a frame is out of service and it is associated with one of the networks under consideration by the network hunt program, the appropriate routine in NETG will modify the A-, B-link busy/idle word to make busy all the B links associated with the busy frame. If the call involved a line to trunk path, and either the line switch frame or trunk switch frame associated with the line or trunk was maintenance busy, the path would be blocked. If a junctor switch frame on one of the networks was maintenance busy, then the call may not be blocked since paths may be available through other junctor switch frames.

Input/Output Program Control

5.196 Via a TTY input message, the network fabric maintenance program determines that a frame is to be removed from service and then transfers program control to one of four routines in NETG, depending upon whether the input request is to make busy one trunk switch frame, one line switch bay, one line junctor switch frame, or one trunk junctor switch frame. If the input message is valid and specifies a frame present in the office, a message signifying acceptance of the message and execution of the action requested will be printed on the TTY. If the message is invalid or specifies a frame not present in the office, an appropriate message signifying rejection of the message will be printed. Any frame which has been made busy can be restored to service by typing a TTY message. The particular message will result in a transfer to one of four routines in NETG, depending upon whether the input request is to idle one trunk switch frame, one line switch bay, one line junctor switch frame, or one trunk junctor switch frame. As before, acceptance or rejection of the message is shown. A TTY message will be printed in response to a request for the status of the network status via the TTY. The message will show, for any network frame affected by a make-busy message, the busy/idle status of all the switch blocks on the frame.

Audit of Out-of-Service Tables

5.197 The NETG program contains its own audit routine which serves three basic functions as follows:

(a) To check the validity of the contents of the flag word, the network indicators, and the out-of-service tables, correct when necessary and print error messages to help trace problems

(b) To check the out-of-service data and print out the status information when requested via a TTY message as well as hourly on request from network fabric maintenance

(c) To check the out-of-service data after the processing of a TTY message to restore to service. The response to a message to restore to service is such that only the contents of the out-of-service tables are changed; the network indicators and the flag word are not touched. This updating job is left to the audit routine. Any discrepancies encountered by the audit will be cleared up and error messages will result.

There are two entry points for the audit routine. The first entry point is for audit 6, whereby idle messages are handled. This entry is used to print a TTY message indicating the status of any network frame, whereas the other entry is used to print hourly and other status messages. Basically, the audit routine restores to the idle state any data which contains conflicting information.

C. Cutover Program for Growth

Primary Functions

5.198 At times it is necessary to transfer groups of subscriber lines from one central office to another. The primary instance of line transfer occurs when a new central office replaces an existing office. In order to maintain uninterrupted telephone service when such a transfer occurs, subscriber lines must be connected to both the old and the new offices (where the old office is the one being transferred from, and the new office is the one being transferred to) for a period prior to cutover in order that line connections to the new office can be tested while switching functions for these lines are performed by the old office. These lines must be kept functionally isolated from the new office and serviced by the old. The cutover program for growth (SACT) provides this isolation. It also provides the mechanism for effecting immediate change of state of lines from isolated (inactive) to supervised (active), or vice versa. The SACT routine provides the capability of accessing isolated lines from that certain test facilities. The cutover program provides the office with the capability of reverting from a post-cut state (state of office after cutover has occurred) to the precut state (state of office before cutover has occurred). This capability is provided to ensure service to lines in the event of trouble indications occurring immediately after cutover. After all translation information in the old office for transferred lines is removed, the cutover program is inactivated.

Cutover Procedure

5.199 The transfer of lines is accomplished by typing a precut message and throwing the master control console key CUT 0 to the proper position. When both offices involved in the transfer have the cutover program for growth, the line transfer can be reversed, if necessary, by resetting the CUT 0 key to the ON position and typing a post-cut message. An inquiry message can be used to determine the status of various indicators. The response tells whether the cutover program is activated or not and indicates office state (precut or post-cut) and state of the cut keys.

5.200 Directory numbers are in groups of 1,000 for translation information. Thus when an office is being replaced, it is desirable to switch all lines simultaneously.

Program Control

5.201 Program control is passed to a SACT routine for handling ESS switch-CUT messages. This routine simply records receipt of the message, updates the message indicator, and obtains status information if requested.

5.202 The cutover audit of the program assures that inactive lines are kept isolated from the switching machine by maintaining cutoff contacts of all lines in the office in the proper state—open for inactive lines, closed for active lines—according to translation information received. The audit has two modes: routine and demand. The routine mode can be entered via a TTY message or maintenance control during routine audit cycle. Peripheral order buffers are used to hold orders to open or close cutoff contacts. The demand mode of the cutover audit can be entered on a high priority basis via a TTY message requesting that line transfer be performed.

5.203 As soon as audit scratch pad is available, the cutover program is entered from one of the call related audits (SADT) for the purpose of actually processing an ESS switch-CUT message. The function of this routine is to put all of the line cutoffs in a precut or post-cut status, depending upon the message. Therefore, it performs all the hardware transactions for transition from precut to post-cut or vice versa.

5.204 To make line transfer independent of the order in which the message is entered and CUT 0 key thrown, an E-level routine is entered to check for their occurrence. When a precut or post-cut message is received, this routine checks the state of the CUT 0 key. If it is in the appropriate position according to the message received, ie, the key position and the most current message agree, the demand mode of the cutover audit is requested on a high priority and the office state is reversed.

5.205 The network matrix exercise program (NMMX) exercises the relay matrix in the line switch frames by setting up a half path to a line in the cutover state. It is necessary for NMMX to know whether to leave the line cutoff opened or closed. Thus at this point program control is passed to a routine in SACT to determine whether to close the line cutoff when it is exercising the matrix.

5.206 Another entry is entered when an OFF-HOOK is detected on a precutover line. This routine sends out a peripheral order buffer to open the network path or the cutoff, since if the cutoff had really been open the OFF-HOOK would have been undetected.

GENERIC UTILITY SOFTWARE

A. General

5.207 The generic utility software provides the craftsperson with full access to data within the system for use in support of system maintenance. The data access functions include:

(a) Printing of data associated with peripheral order buffers, call registers (CRs), and signal distributor and scanner controller enable words

(b) Data transfer (copy, dump, load)

(c) Dataset (create, save, access)

(d) Freeze (freeze block of data)

(e) Overwrite (make permanent data changes).

B. Generic Utility Software Interface

5.208 Figure 44 illustrates the interface between the generic utility software and other system software.

5.209 The 1A processor generic utility program (GULP) interfaces with other 1A processor common systems software and with the 1A application software via the applications transfer table.

5.210 Program interfacing occurs between the 1A common system program and the 1A application program when peripheral units are involved. The GULP program passes the utility messages pertaining to the peripheral units and builds an intermediate buffer table. The address of this buffer is passed to the application program.

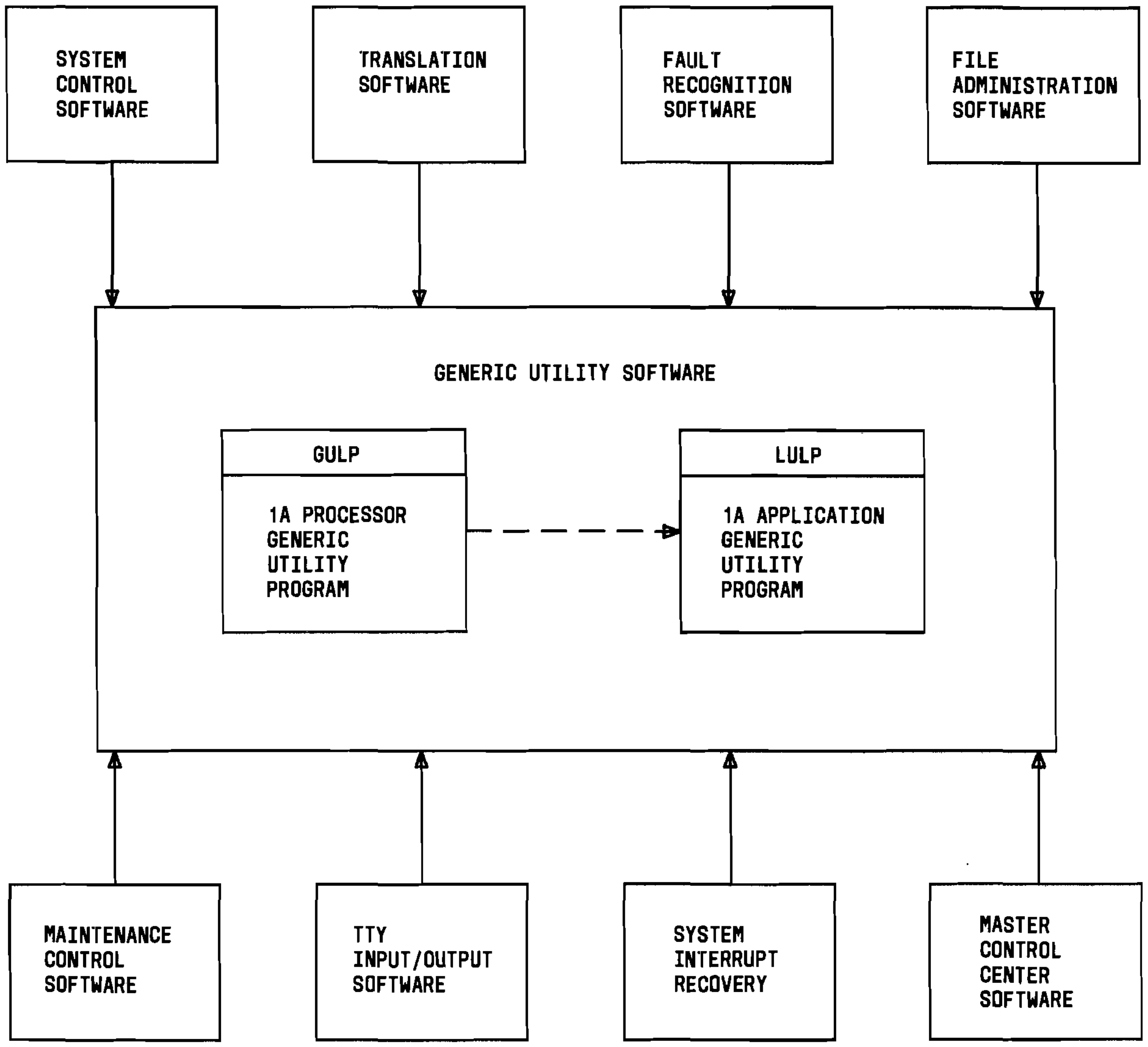

SYSTEM CONTROL SOFTWARE
TRANSLATION SOFTWARE
FAULT RECOGNITION SOFTWARE
FILE ADMINISTRATION SOFTWARE
GENERIC UTILITY SOFTWARE
GULP
1A PROCESSOR GENERIC UTILITY PROGRAM
LULP
1A APPLICATION GENERIC UTILITY PROGRAM
MAINTENANCE CONTROL SOFTWARE
TTY INPUT/OUTPUT SOFTWARE
SYSTEM INTERRUPT RECOVERY
MASTER CONTROL CENTER SOFTWARE

Fig. 44—Generic Utility Software Interface

5.211 The functions provided by the interfaced 1A processor common systems software include:

- Off-line configuration changes
- Transfer of data to and from file stores
- Copy functions
- Place out-of-service files in update mode
- Paging and scheduling
- Display status of MCC matrix
- Tape update.

5.212 Whenever a peripheral utility function is to be performed either on base level or G level, an entry to the peripheral utility execution routines is made via the transfer table.

5.213 The local generic utility program (LULPUTIL) is entered from GULP when a utility request is initiated to obtain a 5-ms entry from the input/output control program for computing the time allowed for utilities over the next 5-ms period.

5.214 Other application software interfaces with LULP to provide:

- Translations
- Memory addresses.

C. Generic Utility Software Function

5.215 The GULP is a manually initiated program accessed via the input/output control software which requests the maintenance control software to page in the utility programs from file store. The GULP receives 3-ms segments of time from the maintenance programs for execution.

5.216 The functions of GULP are performed on G level (each function depends on G-level interrupt). The estimated time required to execute all the utility functions are controlled by a utility flag. When utility flags are active, the PROGRAM OFF-NORMAL lamp on the MCC panel is lighted.

5.217 The 1A processor generic utility program contains six related routines.

- Main memory resident
- Paged message processor
- Utility paged delayed verb execution
- Utility paged immediate verb execution
- Paged data set procedure
- Paged overwrite procedure.

5.218 An automatic overwrite mechanism is provided in the event an overwrite causes problems that are severe enough to cause a phase of memory initialization. On low-level phases, overwrites of main memory in test state are removed; the overwrite procedure is not terminated. On higher level phases, all overwrites are removed and the overwrite procedure is terminated.

5.219 The LULPUTIL program is modularly designed into four program units:

- G-LEVEL TIME CNTL
- PARSER
- DUMP EXECUTION
- PRINTER.

5.220 The G-LEVEL TIME CNTL unit performs three basic actions:

- Computes operating time for utilities
- Computes the number of cycles available
- Cancels the low priority J-level program entries.

5.221 The PARSER unit passes the dump messages pertaining to the peripheral order buffers, CRs, and enable words in the output buffer.

5.222 The DUMP program unit locates and stores the data for the peripheral order buffers, CRs, and enable words in the output buffer.

5.223 The PRINTER program unit administers the printout of the data.

RINGING, TONE, AND RECORDED ANNOUNCEMENT MAINTENANCE SOFTWARE

A. General Purpose

5.224 The primary purpose of these programs is to monitor and test equipment associated with ringing, tone, and recorded announcements. Tests provided by the ringing and tone plant maintenance programs ensure the system is always provided with ringing current, dial tone, audible ringing tone, high tone, low tone, call waiting tone, busy tone, intrusion, receiver-off-the-hook tone (ROH), and NU-TONE tone. The supervisory scan program provides tests for trouble-detecting capability on a daily basis and, upon detecting a fault, requests a TTY message and a major alarm. The recorded announcement machine in the central office contains various tones and announcements which are used to inform customers and operators of various conditions encountered during the process of a call. Any of these facilities is accessed via tone or announcement circuits which are located on the universal trunk frame. This facility contains specific scan points which are monitored and processed by the recorded announcement machine program (RAMP) which provides required input to the TTY program for a printout of the fault(s) located within the facilities. Refer to Fig. 45 for an illustration of the ringing, tone, and recorded announcement maintenance software subsystem interface with other software subsystems.

B. Ringing and Tone Maintenance

Equipment Interface

5.225 Being fully duplicated, the ringing and tone plant contains two ringing generators, two tone sources, and two solid-state interrupters, thereby providing an active circuit and an idle circuit. When under program control, all units are powered; however, manual control may be requested by means of keys at the equipment locations. This action overrides the program control, removes power from

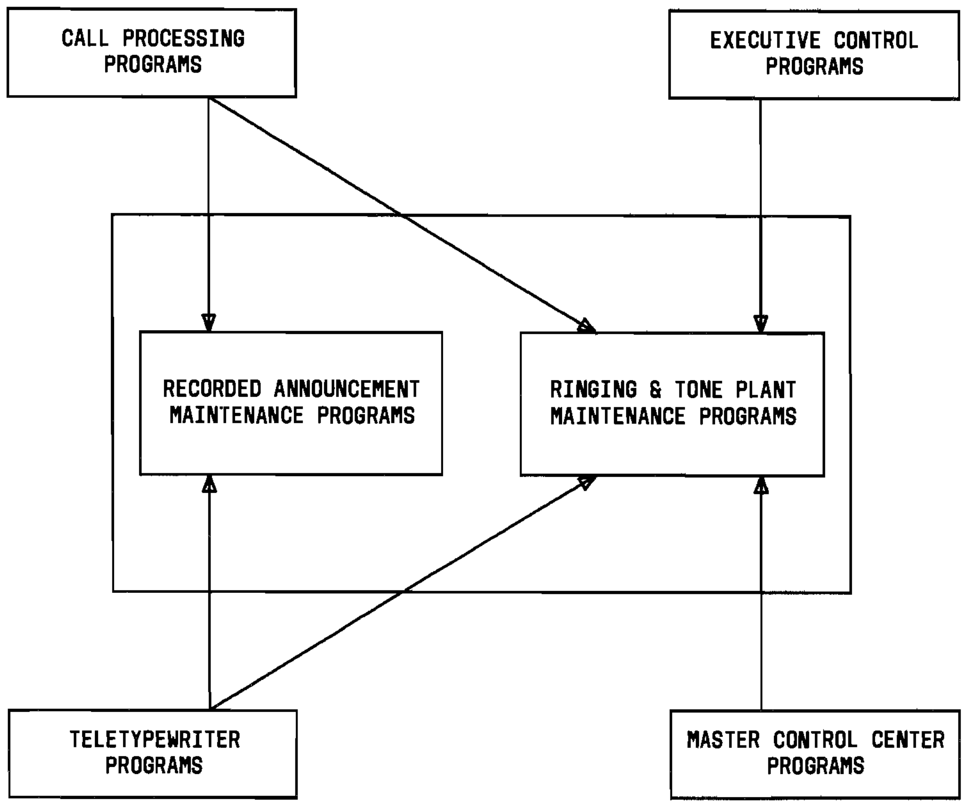

CALL PROCESSING PROGRAMS
EXECUTIVE CONTROL PROGRAMS
RECORDED ANNOUNCEMENT MAINTENANCE PROGRAMS
RINGING & TONE PLANT MAINTENANCE PROGRAMS
TELETYPEWRITER PROGRAMS
MASTER CONTROL CENTER PROGRAMS

Fig. 45—Ringing, Tone, and Recorded Announcement Maintenance Software Interface

all units of one circuit, applies power for all units in the other circuit, and connects it to the distribution network.

5.226 Solid state logic provides the timing for the solid state interrupter which interrupts the ringing current. The interrupt for tone uses solid state logic to operate relays which, in turn, interrupt the tone current. Monitors exist that check the output of each ringing generator and one "low output" monitor that checks each of the tone amplifiers of a circuit. To indicate the ringing phase, ferrods are associated with each solid-state interrupter. In the diagnostic program, relays may be operated to simulate trouble conditions. Load transfer relays are wired in an "up-check down-check" fashion so that if any contact (of a certain set of relays) fails to make or break when appropriate, an indication will be made via certain ferrods. To power an idle interrupter, one transfer relay is released and the other one is operated. The ground cross detection circuit monitors the distribution network of the milliwatt tone supply and the loop checker generator. In order to test these monitors, a trouble condition is simulated by operating a relay.

Ringing and Tone Plant Monitor and Exercises Program

5.227 Program control of the monitoring facilities is provided by the ringing and tone plant monitor and exercises program (TOMK). The primary function of TOMK is to ensure that the system is always provided with ringing current and various tones.

5.228 In order to check the speed of the interrupter, this program is entered every second. On every entry, the interrupter ferrods are scanned and the ringing phase is determined. After a transfer of load, the ringing phase pointer and interrupter are synchronized before attempting to match ringing phases. The matching of ringing phases consists of comparing a scan result with a program store data word, consisting of what the scan should be for that particular phase. When a scan result and the data word match, a pointer is incremented to point to the data word associated with the next ringing phase. The program is entered again one second later and the scan result and the data word are compared. If a mismatch occurs a counter is incremented. An interrupter is marked in trouble after three consecutive mismatches after a failure to resynchronize. The program and interrupter are synchronized when the ringing phase is determined by scanning the ferrods and then updating the pointer to the data word corresponding to the next ringing phase.

5.229 When one of the ringing and tone plant monitors has detected a trouble in the ringing generator, solid-state interrupter on tone source, a TTY message indicating the type of unit and the unit number is generated.

5.230 Whenever the supervisory scan program detects a loss of output from the ROH tone generator, or a ground cross in the distribution network of the milliwatt tone source, loop check generator or the ringing and tone plant, program control is passed to a routine in TOMK for the purpose of setting up the pool phrase for a TTY error message. A major alarm is sounded and a trouble message indicating the unit in trouble is printed.

5.231 Another entry to TOMK is responsible for switching the ringing and tone units from active to idle or vice versa. In the transfer of circuits, there exists an entry which indicates the signal distributor action per translation and zeros the signal distributor timing counter. Then the program enters a routine to perform the requested signal distributor action for the transfer of circuits.

5.232 In addition to an entry being provided for the diagnostic program, a final entry is provided for the audit of the ringing and tone plant. During an emergency action phase, sometime after all of the relays in the ringing and tone plant have been released, the audit segment of the program is entered; however, the audit can also be requested with a TTY input message.

Ringing and Tone Plant Diagnostic Program

5.233 The purpose of the ringing and tone plant diagnostic program (TODA) is to test the trouble-detecting capability of the monitors in the ringing and tone plant and the ground cross detection circuit on a daily basis. In general, there are two daily main program entries to TODA. One entry is to test the ringing and tone plant and the other is to test the ground cross detection circuit. Testing of the monitors or transfer of the loads can also be requested via TTY entries to the program.

5.234 When program control is passed to TODA in order to test the ringing and tone plant, the monitors of the idle circuit are checked first. Relays are operated to introduce trouble and the ferrods associated with the unit in trouble are then scanned to determine if the monitor detected the trouble. The scan result is compared with a program store data word. The data is what the scan result should be. If all of the tests are passed, then the load is transferred and a TTY message is printed indicating the monitor is all right. The monitors of the other circuit are then checked. If a test is failed, the load will not be transferred to the failing circuit and a TTY message indicating the unit with the faulty monitor and the faulty monitor itself is printed.

5.235 Since a trouble introduced in the tone source is not immediately detectable by the monitors, noncall register timing is required to obtain the necessary delays. If a register is unavailable, an appropriate TTY message is printed and the test is terminated. If a test is terminated with the introduced trouble still in the circuit, a TTY trouble message is printed. The trouble can then be removed via a TTY input message. Any of the tests that are performed during the daily diagnostic can be requested via the TTY entry to the program.

C. Recorded Announcement Maintenance

Equipment Interface

5.236 For the recorded announcement function, the recorded announcement frame (RAF) interfaces with the ESS switch via service circuits (announcement trunks). The recorded messages are distributed to the switching networks via the tone or recorded announcement circuit, audible ring and recorded announcement circuit, and permanent signal-partial dial holding circuit. Access to the service circuits is via regular route index, trunk group, and trunk network number translations.

5.237 The trunk group number(s) assigned per channel is stored in the RAF unit type 23 auxiliary block which is required for each equipped RAF to store hardware assignment information. A 19-word block is provided in call store to process scan point reports and to control up to 16 equipped RAFs. The RAF control block contains three words to administer timing and one state word (for RAF status and control) for each equipped RAF. Each RAF control word contains such data as channel request number (CHRN), RAF state item, 6-second cycle counter, out-of-service indicator, recorded tone indicator (RCTN), relay activity bit (RACT), and transition state indicator (TRNS).

5.238 The individual RAF channels are in either a reproduce, record, or maintenance state as indicated by the various assigned scan points. Based on the scan point changes received, the central processor busies or idles the trunk group associated with each announcement channel and controls the channel state (in service or out of service) via the assigned recorded announcement signal distributor points. The central processor responds to the IAM-IDLE input message used for the RAF and its announcement channels to place a RAF channel in service, reinitialize the memory for one RAF, or to reinitialize the entire RAF memory. Associated with the RAF and its announcement channels are various output messages which indicate the status of the trunk groups.

Recorded Announcement Machine Program

5.239 Program control of the recorded announcement facilities is provided by RAMP. Depending upon the status of the various scan points associated with the RAF and its announcement circuits, this program is entered externally at various points in time.

5.240 Depending upon the various conditions or states of the scan points, RAMP is entered via the common systems recorded announcement frame input analysis program (CRFI). Upon checking the state of the frame and the channel request number, the program stores the state word, frame member number, and an end code. If this information is invalid, the program prints a TTY message which indicates an error in translation and then transfers to an initialization routine. However, if this information indicates an operator recording or a 1000-Hz tone recording, the program transfers to a routine which completes the necessary action for that type of scan point. If there is a change on the scan point associated with the voice alarm, the program sets the out-of-service bit for that channel, prints a voice alarm message, and busies the trunk group associated with that channel. Upon detecting a change in the channel request scan point, the program transfers to a routine which completes the necessary action for that type of scan point.

5.241 The RAMP is also entered via a TTY request to idle particular trunk groups associated with the indicated recorded announcement channel. Upon receiving this request, the program checks for the RAF member number. If the RAF member number is 99, then the entire recorded announcement memory is initialized; otherwise, the program transfers to a subroutine which checks for a common systems RAF (CSRAF). Upon finding a CSRAF number, a check is made to determine if that feature is loaded in that office. If so, RAMP transfers to the CRFI program to initialize the CSRAF memory; if not, an appropriate TTY message is printed. If the member number is associated with a RAF, then a check of the unit type auxiliary block is made. If this block is built correctly and the channel request number is valid, then an appropriate TTY message is printed. Next, a check is made of the status of the out-of-service bit for that channel. If the out of service is set, the program resets it and prints a TTY message indicating the trunk groups associated with that recorded announcement channel are idle. The recorded announcement channel may be returned to service by releasing the appropriate key.

5.242 When audit 58 (scan point audit) from the maintenance audit program (MAUD) reaches a RAF scanner row (nontrunk program index 2 or 3), program control is again passed to RAMP, which accomplishes the following actions. The program first checks the frame member number to determine whether a RAF or a CSRAF is detected. If a CSRAF member number is encountered, then the program transfers to CRFI. If a RAF is detected; then the state of the frame is checked. If the RAF is found in the transition state (item TRNS set) at the time the audit is requested, a check is made to ensure that a valid real-time break is occurring, since during a real-time break the RAF cannot be audited. If item TRNS is not valid, then it is reset if necessary and the audit continues. If there is a valid real-time break, then the scan pointer and counter (in MAUD) are updated and program control is returned to MAUD. If the RAF is not in the transition state, the T1 and T2 bits for the channel request, voice alarm, record tone, and operator recording scan points are initialized according to the appropriate status reflected in call store. Then the scan point pointer and counter are updated and program control is returned to MAUD.

5.243 In order to check if any trunk group is unnecessarily left out of service, RAMP is entered every hour from the maintenance control peripheral program (MACR). In this routine, each of the 16 RAFs is checked to determine if any out-of-service bits for that frame are set. For those frames which have at least one out-of-service bit set, the frame member number and the corresponding out-of-service bits set are stored in a memory scratch. An output message indicating which trunk group is out of service is then printed via the TTY.

TELETYPEWRITER SOFTWARE

A. General

5.244 The TTY software serves as the input/output intermediaries between the ESS switch and the craftsperson.

5.245 Both input and output TTY messages are controlled by the TTY software. Input messages are manually initiated at the TTY terminal by the craftsperson, whereas output messages are generated by a client program. The TTY input and output messages, a medium of communication between the craftsperson and the ESS switch, are used for:

- Reporting status of system
- Requesting system action
- Updating customer service and translation information
- Reporting traffic data.

B. TTY Software Interface

5.246 Figure 46 illustrates the software interface between the TTY software and the TTY input/output control and client programs.

C. TTY Translation Input/Output Program (TTYM)

5.247 Whenever an input message is typed and has been identified by the input/output software, TTYM is entered to perform a translation of the input message. The input translation routine parses the message according to certain syntax rules into message fields and parameters. The parsing procedure also separates the message keywords and arguments, converting each keyword into a numeric code which is then used internally to represent the keyword. The translation routine identifies the message verb in the input message directory tables. The translated arguments are then stored in a block of memory referred to as the input message data area.

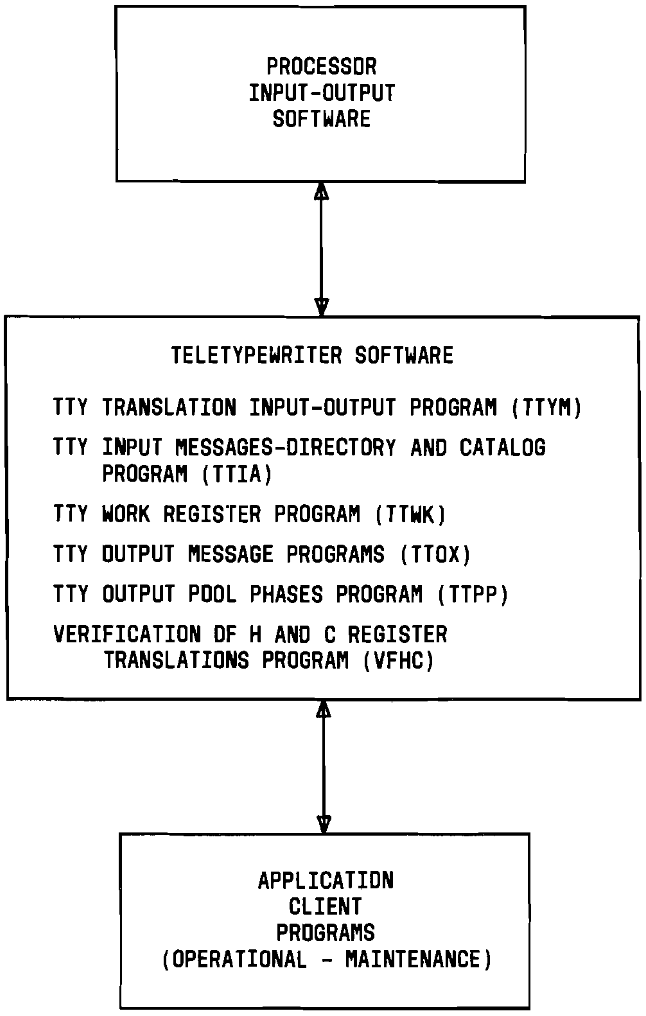

PROCESSOR
INPUT-OUTPUT
SOFTWARE
TELETYPEWRITER SOFTWARE
TTY TRANSLATION INPUT-OUTPUT PROGRAM (TTYM)
TTY INPUT MESSAGES-DIRECTORY AND CATALOG PROGRAM (TTIA)
TTY WORK REGISTER PROGRAM (TTWK)
TTY OUTPUT MESSAGE PROGRAMS (TTOX)
TTY OUTPUT POOL PHASES PROGRAM (TTPP)
VERIFICATION OF H AND C REGISTER TRANSLATIONS PROGRAM (VFHC)
APPLICATION
CLIENT
PROGRAMS
(OPERATIONAL - MAINTENANCE)

Fig. 46—TTY Software Interface

5.248 The TTYM program gives the start address of the message data area and the identity of the input/output channel to the client program. The client program is allowed a full base level program segment in which to process the input message. After the client program completes processing the input request, it releases the message data area memory and returns an appropriate acknowledgment to the input translator via the top of the return stack.

TTY Input Messages—Directory and Catalog Program (TTIA)

5.249 The TTIA program consists primarily of the data required to identify and translate the TTY input messages. Its structure consists of two program units:

(a) Input Message Directory Table, and

(b) Input Message Catalog.

Input Message Directory Table

5.250 The input message directory table (one entry per input message) provides the information needed to identify the TTY input message. Each table entry contains an encoding of the input message and a pointer to the respective input message catalog entry. This table is utilized by the TTY input message identification routine.

Input Message Catalog

5.251 The input message catalog unit provides the information needed to translate and store input message data, and initiates the appropriate client program. This catalog is utilized by the TTY input message translation routine.

Teletypewriter Work Register Program (TTWK)

5.252 The TTWK program is divided into two separate functions:

- Administration of TTY buffer registers (TBRs) and TTY work registers (TWRs)
- Processing of TTY input messages.

Administration of TWR

5.253 An entry is made to TTWK once every second if the associated control flag is active to initiate the TTY input message client program. The program first checks the TWR to determine if it is in an active state; that is, the client program is in progress. If the TWR is inactive, the TTY input message TBR entry is located, the data word is stored, and the client program is initiated. However, if both the TWR and TBR are inactive when the entry is made, the control flag is cleared and the program returns control to the system control program.

5.254 If the client program to be initiated by the TTY input message is inactive, TTWK checks the interrupt maintenance flag before initiating the client to process the message. In the case of a maintenance interrupt while a TTY client program is in progress, TTWK deactivates the TWR, releases the maintenance TTY, and returns the NA acknowledgment. The acknowledgment is a signal to the craftsperson that the input message has been aborted.

5.255 The TWRs buffer only one input message at a time. When TTWK is entered at its loading routine and the TWR is active, transfer is made to the translation program. However, if the TWR is available for loading the message, the data is loaded and the maintenance TTY is suspended until the client program has completed its task. The control flag is set before returning control to the translation program.

5.256 The TWR is also used to buffer an output message generated by the client program.

Administration of TBR

5.257 The TBR is loaded with the client program start address, one word of translated data, and the terminal member number. This action results in the client program being activated.

Input Message Administration

5.258 The following requests are processed by TTWK:

(a) Read scanner row or scan point: The output message contains an image of the scanner row in memory.

(b) Signal distributor operation: The message causes operation of a signal distributor point.

(c) Central pulse distributor operation: The message changes the status of the designated bipolar central pulse distributor scan points.

(d) Verify centrex digits: The message reformats and identifies centrex digits.

TTY Output Message Programs (TTOx)

5.259 The TTY output message programs contain the information needed to format and translate the data for the output messages. The output translations are performed according to translation units specified in the corresponding catalog entry. The translated data is stored in the empty output message registers. When a translation is completed, the registers containing only the raw output data are idled and the translated output message is placed on the output link list.

5.260 The TTY output message programs are listed in Table C.

TTY Output Pool Phases Program (TTPP)

5.261 The TTPP program contains the pool phases used by the TTY output messages. The pool table consists of 256 phases, each indexed by a phase number. It provides client programs with the capability of specifying various sets of American standard code for international interchange characters for printing.

Verification of H and C Register Translations (VFHC)

5.262 The VFHC program performs actions which verifies the H (busy hour traffic schedule) and C (continuous hour traffic schedule) traffic register data.

5.263 Three input options are provided to the craftsperson for obtaining the data printout. They are:

(a) List: Lists all the type measurement code, equipment, group, or office count number.

(b) Search: Identifies a specific type of measurement code, equipment, group, or office count number.

(c) Abort: Aborts the output listing requested by either the list or search options.

5.264 The entry into the program is via the verification TTY input message. If the input message parameters are invalid, a no good (NG) output message is returned to the craftsperson. If valid, the input parameters are stored in the T-scratch memory locations.

5.265 If the list option is requested, and there is not a higher priority request being serviced, the message header for the output message is printed. Each traffic measurement primary translation word is recent hunted. The type measurement code, equipment, group or office count number, and the fast scan indication are moved to the output buffer. An output

TABLE C

OUTPUT MESSAGES

TTY OUTPUT MESSAGE PROGRAM	OUTPUT MESSAGE CATALOG
TTOA	0–63
TTOB	64–127
TTOC	138–191
TTOD	192–255
TTOE	256–319
TTOF	320–383
TTOG	384–447
TTOH	448–511
TTOI	512–575
TTOJ	576–639

segment of one line is printed. A print routine is then called to print additional segments until all have been printed.

5.266 If the search option is requested, a sequential search of the register is made to locate the specific type of traffic. If found, a TTY output message is returned which verifies the translation data.

LIBRARY AND PAGED PROGRAM CONTROL SOFTWARE

A. Library Control Software

General

5.267 The library control software for No. 1A ESS switch (Fig. 47) administers the storage, loading, and execution of special purpose nongeneric programs in a generic environment. The library client programs are initially stored on magnetic tape providing an inexpensive means of bulk storage. From tape, the library programs may be loaded into a special file store area allocated for the semipermanent storage of library programs. The library control system provides for retrieving and controlling the execution of the library programs (ie, clients) from either tape or file store as conditions warrant.

5.268 There are two basic types of library client programs: those designed to be executed out of a rover program store with a K-code = 36 (referred to as program store 36) and those designed for paged execution (from file store only) out of the program store paging area. Program store 36 programs may be loaded for execution from either tape or file store. Paged programs are loaded for execution from file store only. Even though file store is loaded with both program store 36 type and paged type library programs from tape (using program store 36 as a buffer), no provision is made for the execution of a paged type program from tape.

Library Program Packaging

5.269 The basic execution entity loaded into the 1A processor by the library control system is the library package. A library package consists of from one to eight library programs; each program consists of from one to eight tasks (ie, subprograms). Note

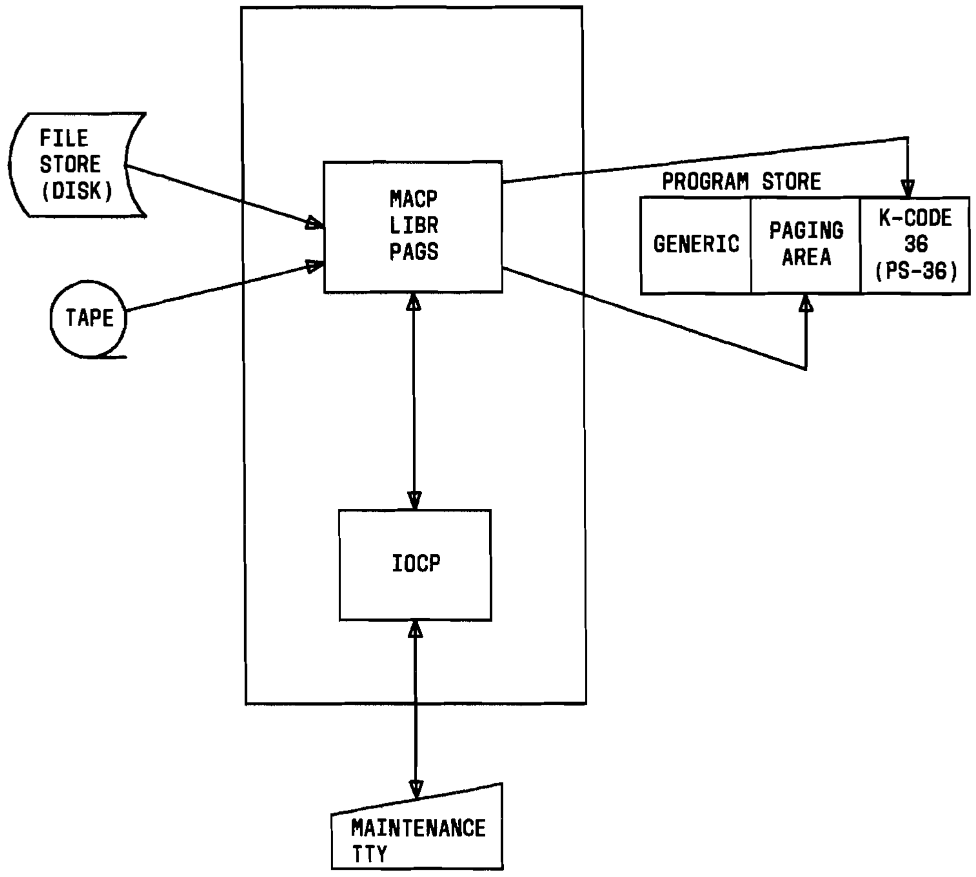


Fig. 47—No. 1A ESS Switch Library and Paging Control

that a library program may consist of one or more assembly units (ie, pidents).

5.270 Program store 36 packages are restricted to no more than 65,536 words (the size of a 64K program store). Paged library program packages have a practically unbounded size, since the size limiting factor is likely to be the amount of file store area allocated for library program storage in any particular office.

5.271 Library program packages are initially introduced into a 1A processor from a tape containing as many packages as necessary. Each package is always uniquely named (with respect to other package names on the same tape) even though the same program name may appear in more than one package. This uniqueness of package names is required so that the specification of a package name, program name, and task number will always clearly identify only one library client. A given library tape may have a mixed assortment of both program store 36 type and paged type packages. However, a package type must be homogeneous with respect to program type; eg, a program store 36 type package may not contain other than program store 36 type programs.

Library Program Loading

5.272 The library load function of the library control program (LIBR) provides three basic load mechanizations:

- Tape to program store 36
- Tape to file store
- File store to program store 36.

5.273 The loading of a program store 36 program from tape occurs under control of pident LIBR and is the simplest type of load. It involves copying the memory image of the program store 36 package from tape to memory, performing validity checks, and allocating scratch space to the program store 36 programs from the program store 36 area that is not occupied by the programs.

5.274 Before library programs can be executed from file store, the desired subset of library program packages (program store 36 and/or paged type) must be copied into file store from tape. Loading a library package from tape to file store is also accomplished under control of LIBR. Program store 36 is used for buffering the transfer, requiring that no library programs be active. The loading process performs memory management of the file store area for storage of up to 16 library packages. The size of each package is rounded up to the next multiple of 1024 words; this facilitates recovery from file store mutilation and aids in memory management. The file store library area is partitioned into two sections. The first area contains 16 catalogs used to locate and describe the library packages. The second area contains the actual text of the library package. Each package is stored at a 1024-word (BINK) boundary to aid recovery from mutilation. If the writable store audit (SAWS) detects a mutilated BINK, it is only necessary to reinitialize the area belonging to the library package containing the bad BINK. When the load file store from tape operation is requested, and if a contiguous area in file store can be found (using the first fit algorithm), the package is copied into file store and a catalog entry for it is created. Once the file store has been initialized with a set of library programs, loading program store 36 from file store is much the same as loading program store 36 from tape.

5.275 The LIBR program also supports several file store related administrative functions. Packing of the file store library area can be done to remove external fragmentation and create the largest possible contiguous area of free space. Library packages can be removed from the file store library area to make room for new packages. To guide in the packaging and removal functions, a memory map of the file store library area can be obtained upon manual query of the system.

Library Program Execution Facilities

5.276 The ability to manually initiate and terminate library programs is provided by the library system by interfacing with the MACP and PAGS. With the appropriate input messages, a library package, a program within that package, and a task within that program can be selected for initiation or termination. The library system processes this input information for validity and then communicates the desired action to MACP.

5.277 A program store 36 type library program is brought into a spare program store which has automatically been configured to K-code 36. The direct execution of the program store 36 programs is performed by MACP as if they were main memory resident generic programs. The MACP provides for multiexecution of clients within a program store 36 package, the number of programs allowed to run at one time being application dependent. Library clients execute in the MACP library class. If the spare program store is needed for generic integrity, it may be seized by the maintenance programs on interrupt level, thereby terminating the library function.

5.278 For paged library programs, LIBR locates (on file store) and provides linkages between the paged clients and MACP/PAGS. At execution time, PAGS loads the paged library program into one of the paging areas in generic program store. The paged client is also executed in the MACP library class.

5.279 An input facility for passing data to the executing library client programs is provided. This facility allows maintenance personnel to pass up to 16 words of data along with miscellaneous control information to a library client by using the appropriate input messages.

Generic Integrity

5.280 Program store 36 resident library clients are often required to interface with generic programs for a variety of reasons. Frequently when transferring from a generic routine back into a library client (ie, from generic program store to program store 36), it is desirable to verify the validity of the library system before returning control to program store 36. Returns to program store 36 are controlled by the library control common traps administrator program (LIBRTRP1) and the No. 1A application traps in generic identification and compatibility tables pident (PGID).

5.281 A library client is allowed to transfer to a generic routine without return trapping if the time duration is to be short, ie, no real-time breaks, delayed returns, or time-consuming data manipulations, etc. If the generic routine is to have control for a relatively long period of time, returns to program store 36 are via the traps.

5.282 The library trap contains unique trap points for independent returns to program store 36. The first function of a trap is to retrieve the return address of the library client in progress and to set up failure and wait addresses. Checks are also made to determine the compatibility and sanity of the program store 36 program (ie, library client). If these checks should fail, the return to program store 36 is "trapped" and control is handed to the trap failure address. When the trap checks reveal that a program store 36 hashing is in progress, control is transferred to a wait address from which later attempts may be made to return through the traps back to program store 36.

5.283 In addition to trapping and compatibility functions, pident PGID performs generic identification as a response to the "who are you" input message. Pident PGID is entered to identify the generic program number and issue number.

B. Paging Program Supervisor (Pident PAGSUPER)

General

5.284 The PAGSUPER pident provides all of the run-time functions necessary for the execution of paged programs. A paged client must be a MACP client. Page loading is accomplished with frequent transfers between the executive interface, page loading sequence, and disk administration (pident DKAD) interface, program units of PAGSUPER. The PAGS program actions consist of paging into main memory the required portions of the program: a page directory table, partial page, and, if necessary, CPSECT and subroutines. The PAGS also corrects the contents of the linkage vector tables in the paged program to reflect main memory addresses or relative addresses for this page. Tables so affected include the INTSUB table in the CPSECT, the subroutine directory table and any subroutine private vector tables. The PAGS operates on base level only. It is a MACP client for purposes of beginning to process a paging request. However, most of its work is completed as a result of returns from DKAD or entries from executive control. The PAGS is naturally segmented by its interfaces with file store; it takes time breaks while awaiting file store request completion. During the INTSUB modification, the computation may continue over more than one time segment. The PAGS program performs time segmenting for this case. The MACP passes one paging request per entry to PAGS; however, PAGS may be working on more than one request at a time.

5.285 When MACP determines that a paged client program is to be run, it enters PAGSUPER. The PAGS program blocks the execution of the client until its paging is complete by placing it in a special wait state. When paging is complete, the client is started.

5.286 Data passed by MACP is used to determine the paging area for the client. After determining that the paging request is legal, PAGS initiates page loading.

5.287 For the remainder of this segment, PAGS sets up and issues the request for the page directory entry from file store. The state vector reflects this on return to MACP. Since PAGS is limited in its number of outstanding file store requests, the program may simply set the state vector to initiate the page directory entry request on entry to PAGS after a time break.

5.288 The next entry to PAGS comes from either DKAD (file store request complete) or executive control (file store request not initiated). The executive control program enters PAGS and requests that the page directory entry be retrieved from file store. The program also verifies that DKAD is still processing all outstanding page directory entry requests (and other file store requests) using the FIND_SPECIFIC macro interface with DKAD.

Partial Page and CPSECT Loading

5.289 Upon completion return from DKAD, PAGS begins partial page loading. This takes place in several steps. The PAGS waits for the completion of each file store request before continuing to the next step. When file store requests are not available, they will be reissued. First, the partial page specified in the page directory entry is requested from file store. Using information in the partial page, a CPSECT may be requested from file store. Upon completion of CPSECT actions, the subroutine directory table is modified to reflect the main memory addresses of the paged programs. If a program has no subroutines, the state vector indicates end of load.

5.290 Subroutine directory table modification takes place upon entry from executive control. First, the CPSECT internal vector table addresses are modified to reflect the main memory addresses of the subroutines. This modification may require more than one time segment to complete. Upon completion, the subroutine directory table is modified to reflect the relative addresses where the subroutines were paged into main memory.

Subroutine Loading

5.291 Each subroutine is then paged into main memory. The PAGS program issues multiple file store requests, requesting as many subroutines as it can at that time. A PAGS entry from executive control monitors the status of loading. Upon completion return from DKAD for the last outstanding request, the state vector indicates subroutine loading complete.

5.292 A separate entry from executive control modifies the subroutine private vector table entries to reflect paged program main memory addresses.

End of Load

5.293 The PAGS end-of-load actions consist of removing the wait state blockage of program execution, set up by MACP on entry to PAGS. This indicates to MACP that the client may be started.

C. Miscellaneous Run-Time Loading Functions

User Paging Information Routines Program Unit

5.294 The PAGS performs the run-time function necessary to provide information about the page loading state to the client program. The function of the GETCPADR macro is to obtain the CPSECT address. The GETPGADR macro obtains the address and size of the paging area. The GETPGNUM macro obtains the current page number. The return from each of these functions is to the client program.

5.295 The routine PAGSGFRA is called to obtain the address and size of the free or unused space on the current page. Client programs may utilize this area of the page for temporary memory storage. The routine is called using the standard CALL_SUB interface.

5.296 To accommodate the disk-paged mode of library operation, the PAGS/ALOC routine is called by MACP to allocate a paging area for the library class. If the request is successful, the subclass paging area pointer table is changed to contain the address of the allocated paging area. The PAGSDLOC routine is called by MACP to deallocate a paging area for the library class. If the request is successful, the subclass paging area pointer table entry is cleared.

Errors Program Unit

5.297 The PAGS also provides a complete set of error diagnostics through TTY messages to operations personnel to clarify any error that may occur while it is in the process of loading a page for a client. These routines are contained in the errors program units, and are accessed through the error exits from routines in other PAGSUPER peripheral units. When errors are detected during page loading or linking, an applications program is called via a processor applications transfer table (PATT) entry to generate an error message on the TTY. The detected error is corrected, if possible, and an audit routine is requested to check for further errors.

Audit Program Unit

5.298 The audit program unit provides an entry for periodic checks on the self-consistency of the MACP assigned control memory blocks, and the load states of the various clients. Should a control memory block be found mutilated, it is rebuilt using independent information gained by searching the MACP class tables.

Link and Return to Paged Program Unit

5.299 The link and return to paged program unit is called by the client program to pass or return control to another paged program section. Control is passed directly to a paged program that is on the current page except that a segment break may be taken, if requested. When the target address is in a paged program but not on the current page, the MACP control memory block is made ready and page loading begins. The page loading sequence, as described in run-time loading, will be executed.

5.300 Entry to this program unit to perform linking functions to another program section is via the LKPGSECT macro and contains a return address. The RTNSECT macro is used to perform a return function to a paged program section. The return address is the address saved by a previous LKPGSECT macro call.

NETWORK MAINTENANCE SOFTWARE

A. General

5.301 The network maintenance software provides for testing and maintaining the network links and switches.

B. Software Interface

5.302 Figure 48 illustrates the interface of the network maintenance software with other system software programs.

C. Network Maintenance Software Function

Network Maintenance Action Program (NMFL)

5.303 The ESS switching network consists of links and switches through which talking or signaling paths are established. The peripheral orders that control the links and switches are loaded in a buffer called the peripheral order buffer. Periodically, at every fifth J-level interrupt, the peripheral order execution program sends the orders to the appropriate unit. If a failure condition occurs during the execution of a peripheral order, network fault recognition program (NMRF) is entered and processes the peripheral order buffer that contains the failed order.

5.304 ➧The NMRF program deactivates the peripheral order buffer which contains the failure and places on the maintenance unexpected result list (MURL). The NMFL program identifies the failing peripheral order buffer and places it on the MURL. The NMFL program unloads the MURL and series of checks, depending upon the type of failure as indicated by scan point results, are performed.➧

5.305 In the case of a signal distributor failure, all trunk network numbers associated with the failure are matched with all the trunk network numbers associated with signal distributor orders. If a match is found, a diagnosis of the associated trunk is requested and the program returns to the client failure address. If a match is not found, the program transfers to the client failure address without a trunk diagnostic request.

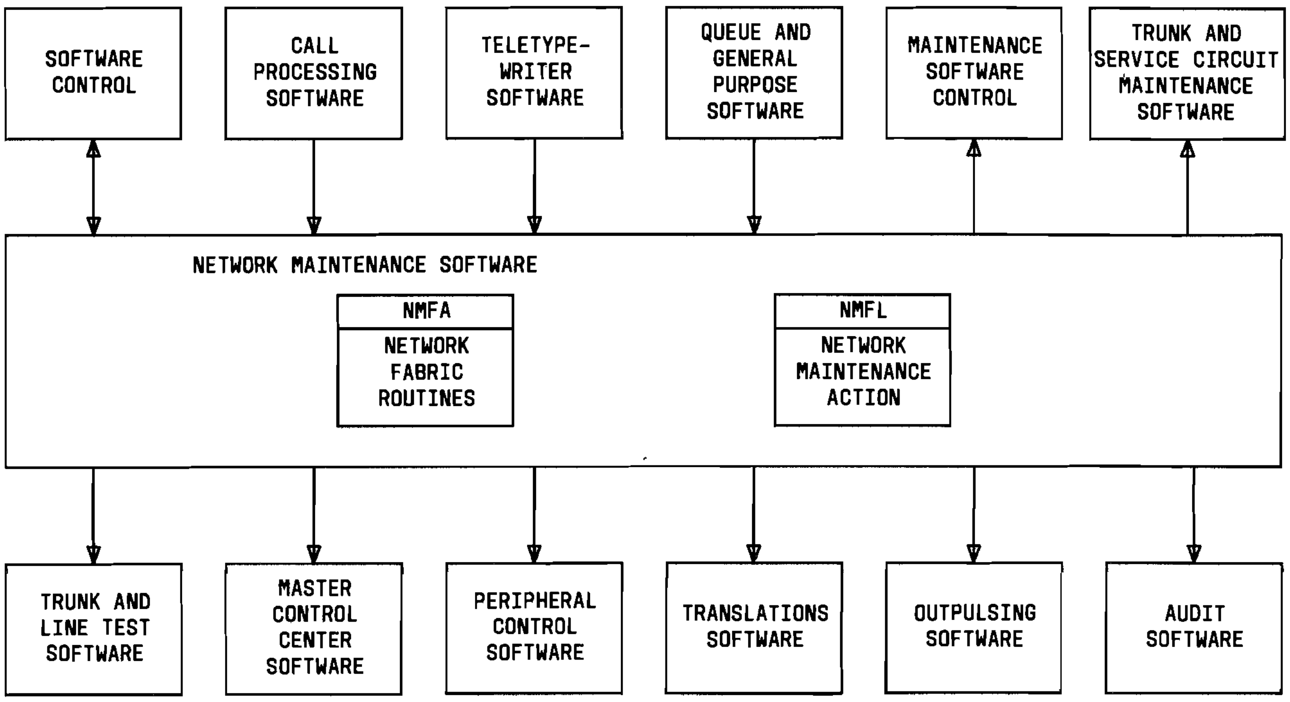

SOFTWARE CONTROL
CALL PROCESSING SOFTWARE
TELETYPE-WRITER SOFTWARE
QUEUE AND GENERAL PURPOSE SOFTWARE
MAINTENANCE SOFTWARE CONTROL
TRUNK AND SERVICE CIRCUIT MAINTENANCE SOFTWARE
NETWORK MAINTENANCE SOFTWARE
NMFA
NETWORK FABRIC ROUTINES
NMFL
NETWORK MAINTENANCE ACTION
TRUNK AND LINE TEST SOFTWARE
MASTER CONTROL CENTER SOFTWARE
PERIPHERAL CONTROL SOFTWARE
TRANSLATIONS SOFTWARE
OUTPULSING SOFTWARE
AUDIT SOFTWARE

Fig. 48—Network Maintenance Software Interface

5.306 During the processing of the peripheral order buffer which contains the failure, NMFL performs the following actions:

(a) Identification of the failure activator

(b) In-depth testing, further isolation, or retrial on per-path and per-call peripheral order buffer maintenance test failures

(c) Generation of TTY output messages for notifying office maintenance personnel of the failure

(d) Error analysis of network switches involved in certain types of peripheral order buffer failures

(e) Isolation of failure trunks, and request of automatic diagnosis

(f) Isolation of a failing line and certain failures of automatic line tests

(g) Restoral of nonfailing trunks

(h) Restoral of supervision to all lines

(i) Idling of peripheral order buffer.

Network Fabric Routines Program (NMFA)

5.307 The NMFA program provides the craftsperson the capability to remove portions of the network fabric from service and establish partial paths in the network for testing, repairing, and replacing faulty crosspoints in the network fabric.

5.308 The NMFA program performs the following actions in response to TTY input messages:

(a) Lists incoming trunks that are assigned to an identified trunk switch frame and grid

(b) Makes busy and removes the links from service by marking the associated bits of information busy in the network map

(c) Restores links which have been previously taken out of service

(d) Performs tests on suspected crosspoints and switches in order to establish a partial path in the network frame

(e) Displays the present condition of the link or switch

(f) Prevents F-level interrupts from occurring when removing a concentrator from a line switch frame.

FAULT RECOGNITION SOFTWARE

A. General

5.309 The fault recognition software is implemented when a system F-level interrupt occurs because of a failure in the network peripheral units. The fault recognition programs first determine which peripheral unit is at fault and then reestablishes an operational configuration that will allow the continuation of the interrupted call or maintenance procedure. A diagnostic is requested to be performed on the failed unit at a later time.

5.310 The peripheral units checked by the fault recognition software are the central pulse distributor, signal distributor, peripheral unit bus, scanner controller, network and signal distributor controller, and automatic number identification (ANI) circuit.

5.311 The peripheral units are connected to central control via two bus systems, central pulse distributor bus, and peripheral unit bus. The central pulse distributor bus provides the communication path between the central control and the central pulse distributor community. The central pulse distributor bus consists of the central pulse distributor execute bus, central pulse distributor echo bus, central pulse distributor enable bus, and central pulse distributor enable verify bus. The peripheral unit bus provides the communication path between the central control and the peripheral unit community. The peripheral unit bus consists of the peripheral unit address bus and scanner answer bus.

5.312 A peripheral order processing failure will generate an F-level interrupt. The failure types are given below.

(a) All-Seems-Well: The all-seems-well message is returned to the central control from the addressed peripheral unit. When a peripheral unit is addressed the all-seems-well response signifies that the unit is being properly addressed via a valid name, matched mode, valid parity, and expected received data. The central control accepts the all-seems-well reply, or lack thereof, from each reply bus and records the message in the all-seems-well flip-flop.

(b) Answer Parity: The central control selects the parity result recorded in the answer parity flip-flop associated with the peripheral unit reply bus and checks the result against the parity generated over the data received. The results of the check determine whether proper or improper parity was received.

(c) Autonomous Peripheral Unit: Processing and maintenance work is performed autonomously by the autonomous peripheral unit. If trouble occurs within the unit, it is reported to the central control. The failure is recorded in an associated flip-flop resulting in an F-level interrupt.

(d) Peripheral Sequencer: If during the execution of an order by the peripheral sequencer and a trouble is detected, a sequencer failure is recorded in the failure summary flip-flop.

B. Fault Recognition Software Interface

5.313 The interface between the fault recognition software and other system software is illustrated in Fig. 49.

C. Fault Recognition Software Function

Central Pulse Distributor Fault Recognition (CPFR) Program

5.314 Whenever a system F-level interrupt occurs and the system interrupt recovery software has determined that the failure occurred within the peripheral units, CPFR is entered. If the peripheral unit failure was due to an actual fault, CPFR retries the order in an attempt to isolate the source of trouble by using various configurations of the central

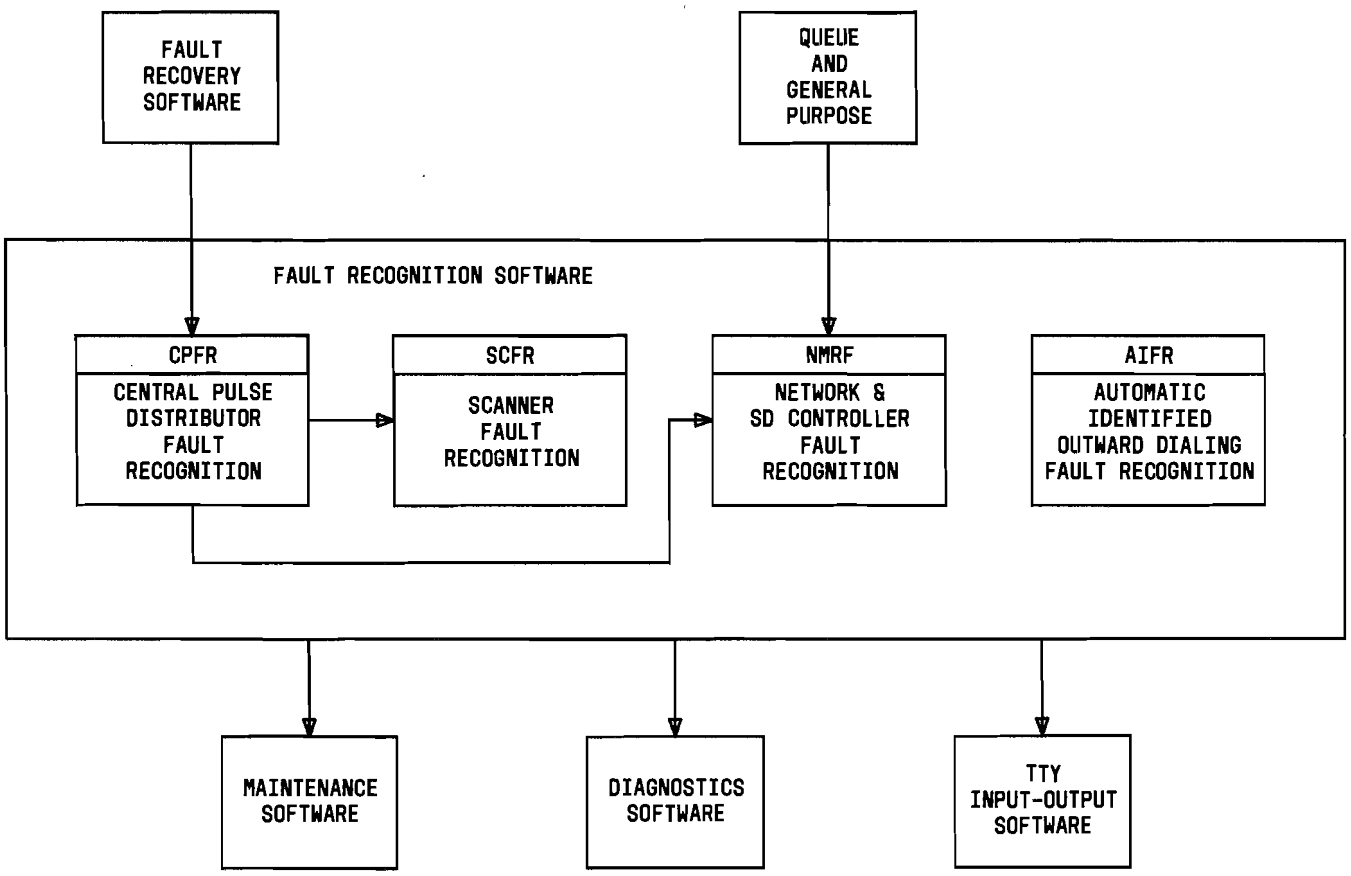

FAULT RECOVERY SOFTWARE
QUEUE AND GENERAL PURPOSE
FAULT RECOGNITION SOFTWARE
CPFR
CENTRAL PULSE DISTRIBUTOR FAULT RECOGNITION
SCFR
SCANNER FAULT RECOGNITION
NMRF
NETWORK & SD CONTROLLER FAULT RECOGNITION
AIFR
AUTOMATIC IDENTIFIED OUTWARD DIALING FAULT RECOGNITION
MAINTENANCE SOFTWARE
DIAGNOSTICS SOFTWARE
TTY INPUT-OUTPUT SOFTWARE

Fig. 49—Fault Recognition Software Interface

pulse distributors and peripheral unit buses. When a good configuration is found, the network routing information is updated and a diagnosis of the faulty unit is requested.

5.315 If the F-level interrupt occurred during supervisory scanning, the scanner enable route is changed and a new bus and controller are selected in an attempt to isolate the trouble. If a scan order caused the failure, a program transfer is made to the network fault recognition program (NMRF).

5.316 Whenever a scanner fault is isolated by the scanner fault recognition program (SCFR), CPFR is entered to perform tests on the central pulse distributor and peripheral unit bus. If the central pulse distributor is found to be at fault, a diagnostic is requested, and the central pulse distributor lamp on the MCC is lighted. If the peripheral unit bus is faulty, it is taken out of service, and its failure is recorded.

Scanner Fault Recognition Program (SCFR)

5.317 The SCFR program is entered from CPFR whenever either an all-seems-well failure message or an enable verify mismatch has been detected. The SCFR program retries the peripheral unit order on all possible routes. If the peripheral unit order is then successful, the route is updated, associated secondary lamps on the MCC are lighted, and a program transfer is made to the maintenance software. However, if there are no routes available to the addressed scanner, a major alarm is sounded, and a TTY output message is printed.

Network Fault Recognition Program (NMRF)

5.318 The basic function of NMRF is to maintain a network configuration in which peripheral unit orders that are addressed to the network and signal distributor controllers are successfully executed.

5.319 At the beginning of each network cycle, NMRF scans the F-scan points of the network, signal distributor, and master scanner controllers. If the scanning function detects a change-of-state in the F-scan points, a failure order is recorded in the peripheral order buffer. The failing order is then processed during the next network cycle. Also, if a controller cannot complete a peripheral unit order within one network cycle, a failure is recorded.

5.320 The NMRF program services TTY input messages that:

- Activate or deactivate the F-level, F-scan, and error snaps
- Inhibit maintenance of frame
- Restore maintenance of frame
- Print F-level, F-scan, and error snaps.

Automatic Identified Outward Dialing Fault Recognition Program (AIFR)

5.321 The AIFR program is entered whenever a trouble has been detected during the operation of the ANI circuit. If the trouble is due to a circuit failure, the program requests a diagnostic of the circuit, and provides a TTY output message to assist the craftsperson in locating the trouble.

5.322 If the originating station is not identified, the program idles the receiver.

5.323 The AIFR services TTY input messages that:

- List ANIs out of service
- Remove ANIs from service
- Restore ANIs to service
- Restore maintenance to receiver
- Remove maintenance from receiver
- Print daily counters on ANIs
- Control failure information in printout.

SYSTEM UPDATE SOFTWARE

A. General

5.324 The No. 1A ESS switch update software updates the system generic with either a full update (introduction of a complete new issue of generic and/or office dependent data).

B. ➧ System Update Software Interface

5.325 Figure 50 illustrates a block diagram of the system update program.

C. System Update Software Function

5.326 For a file store update, a previously designated set of file stores is taken out of service

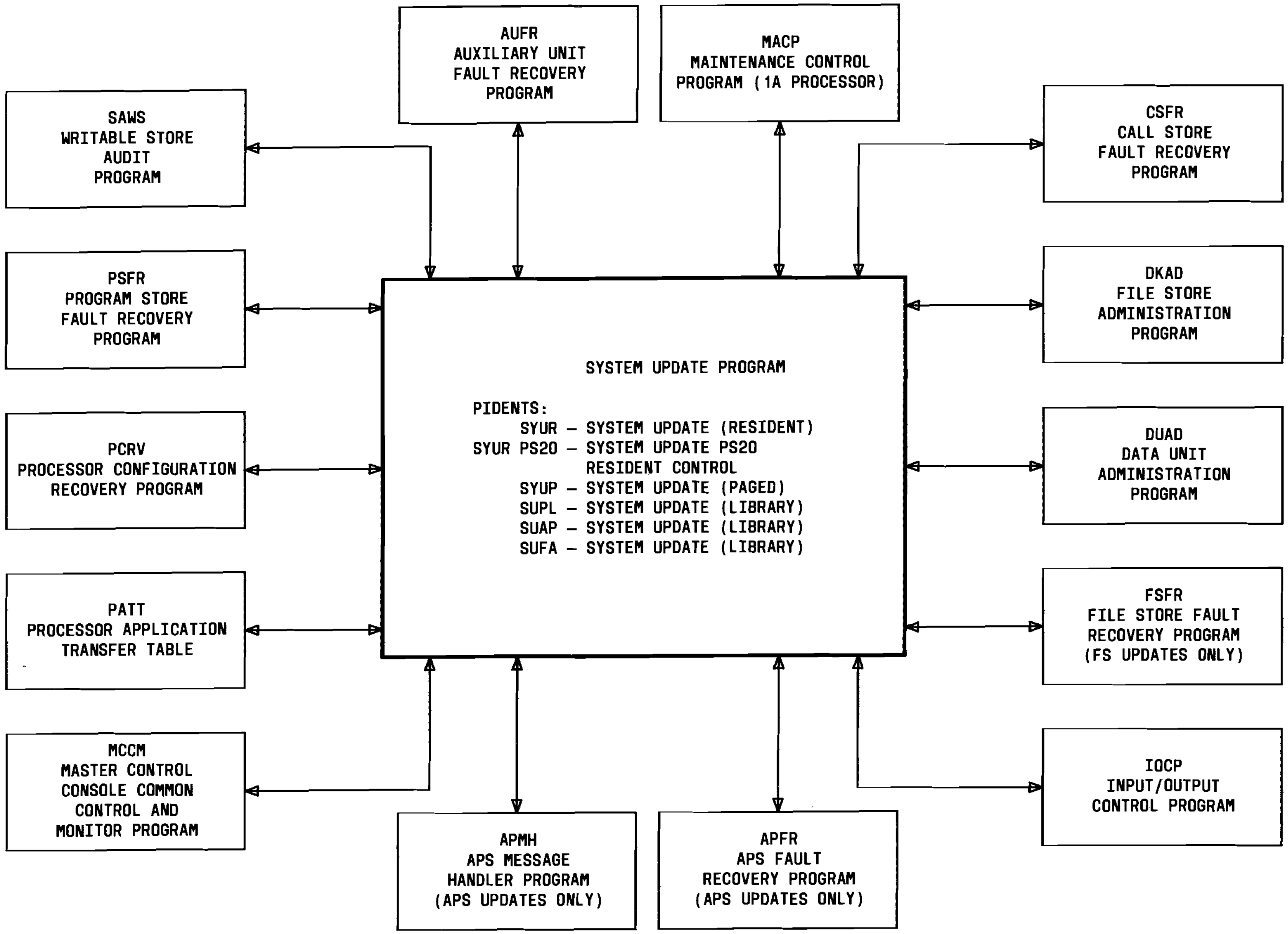


Fig. 50—System Update Program—Program Interface Block Diagram

and updated on a segmented basis prior to the interruption of normal call processing. This minimizes system disturbance during introduction of an entire new issue of generic program or office data. For the Attached Processor System (APS) update, no units are taken out of service. Instead, the APS update programs send an APS message to the file manager interface (FMI) which "unlocks" the UPDATE file on the 3B disk, copying the NORMAL file into it. The FMI then uses the UPDATE file to store the update data. The system update program, being an MACP client, must operate in the 3-ms segment allotted by MACP. When the update procedure begins, a check is made to see if any other data changing programs are in progress. If so, the update will stop. If not, a lockout mechanism will be set up to prevent those programs from running. Examples of programs affected are the recent change program, generic utility overwrite, tape writing program, or the system audit of stores using tape. For a file store update, this lockout is necessary because any changes made after the file stores are split will be lost when the updated copies are returned to service. After all data has been read in from the update tape(s), any mapping of transient data is performed. Then, the file stores are switched so that the updated files are in service and the obsolete files are out of service, the new data is pumped from the updated file store copies, and then a previously designated phase of memory reinitialization is started. For an APS update, the files are not switched but access is switched. After data mapping is done, access is switched so that normal reads and writes go to the UPDATE file.

D. Full Update

5.327 The library program SUPL is used for full file store updates. The library programs SUAP and SUFA are used for full APS updates. A full update, since it entails a pump from the updated file store, requires considerably more time than partial update (or point load). The following paragraphs present a description of each step of a full update.

Preliminary Verification

5.328 A full update entails a generic update and/or an office dependent data (generated by the office data assembler) update. The tape header must be verified to ensure the user has mounted the correct tape. Since the headers for generic and office dependent data tapes are slightly different, the verification procedure for both tapes is given.

(a) ***Generic Update:*** To verify a tape for a generic update, the following checks are made.

(1) The tape is a generic tape.

(2) The tape is classified as a system reinitialization tape (tape format is consistent with that which is expected by SUPL, SUAP, or SUFA).

(3) The generic identification data matches that which is inputted on the TTY.

(b) ***Office Dependent Data Update:*** To verify a tape for an office dependent data update, the following checks are made:

(1) The tape is an office dependent data tape.

(2) The tape is classified as a system reinitialization tape (tape format is consistent with that which is expected by SUPL, SUAP, or SUFA).

(3) The date and time the tape was written matches the date and time inputted on the TTY.

(4) The office identification code and the generic identification on the tape match that stored in the system.

5.329 In addition to verifying a tape update, SUFA, which can only be used for a full generic update, must do extra initialization. This is necessary because SUFA must be able to access the 3B processor via the attached processor interface (API) while running in a file store generic. These additional requirements are:

(a) All auxiliary unit bus 0 file stores must be out of service. The auxiliary unit bus 0 is used to access the 3B processor.

(b) All auxiliary unit bus 1 file stores must be in service to permit normal system activity to continue.

(c) The file store 0&2 key at the MCC update panel must be selected to ensure that the file stores are used to pump memory during an abort.

(d) Translations must be changed to indicate that API 0 is in AUB port 0.

(e) The API 0 is restored unconditionally.

(f) The FMI state is initialized to normal.

5.330 A mismatch in any of the above checks will cause an immediate termination of the MACP job, and an error message is generated and printed. If all tape header information matches correctly, SUPL checks to see if a file store has been manually selected for update. If not, a message is printed and the program lapses into a wait mode for 1 minute. At the end of 1 minute, if a file store has not been chosen, another message is printed and the update is terminated. If a file store selection is made before the 60 seconds expire, an acceptance message is printed. For an APS update, SUFA makes a check of the file store 0&2 key at the MCC update panel, or SUAP makes a check for the NORMAL key.

E. Data Transfer

Merge Data File

5.331 The merge data file on a full update tape contains the main memory-to-file store map, the IDtag-to-file store (ID2FS) map, hash sum head table, and hash sum tables. Each of these structures is incomplete in that they contain entries pertaining only to the particular type of tape. That is, a generic tape would have generic data range blocks in the ID2FS map plus range blocks for all Datapool defined file store areas, only generic hash sums in the hash tables, etc. The merge data file must therefore be buffered until an indication is received that all tapes have been read in, so that all data required for the generation of tables is available. The remainder of the data needs no buffering before being placed in the corresponding file store. The merge data file is saved in a designated main memory buffer area of the rover program store for later use.

Update Data Blocks

5.332 Following the merge data file on the tape is the remainder of the update data. This data is placed in files of 32 BINKs (BINK=binary K= 1024). The data may be placed directly on the appropriate file store without change. Each file store is taken out of service only when data is encountered on the tape which must be placed in that file store. Thus, changes in system configuration are made only as necessary and are delayed as long as possible. Before any file stores are removed from service, all recent change activities are inhibited. During an APS update, the entire UPDATE file is unlocked so data encountered on the tape can be written directly to the UPDATE file.

F. Building System Maps and Tables

5.333 After the last data file has been read in, the tape is rewound to the beginning-of-tape mark, and a message indicating completion of work on that tape is printed. If another tape is to be included at this point in the full update, it must be mounted and the corresponding message entered. If there are no more tapes to be included in the update, the actual main memory overwrite may be initiated or a test merge may be requested, verifying the compatibility of the tapes. In any case, system update enters a wait mode which will time out in 30 minutes if a message is not entered. At the end of 10 minutes, the tape completion message is printed again. If no instructions are received in 20 minutes, another message is printed. If after 30 minutes from the end of processing of the current tape no input message has been received by the update program, the time-out abort message is printed and the update is terminated. However, if the system is committed to the update before the 30-minute time-out, the wait mode is ended and the processing of the merge data files begins.

5.334 Before building system maps, the update program must verify that all components are available. Any data which has not been entered via tape is extracted from the current system maps and assumed to be correct. Since SUFA is used for file store to APS retrofits, generic and all office dependent data tapes must be inputted. If only a generic tape were read in for a file store to file store update, SUPL or SUAP would extract from the respective tables (currently in the system) all file store descriptor blocks, range blocks, and hash sums not entered from the tape. Then, new maps and hash sum tables are created by merging the data which was entered from tape with that which was extracted from current system maps. Verification includes assuring that range blocks do not overlap in the ID2FS map and that file store descriptor blocks are consistent in the CS2FS map. These updated maps and tables are then written onto the associated file store for a file store update. For an APS update, the updated maps and tables are written by SUFA and SUAP to the UPDATE file on the 3B disk.

G. Data Mapping

5.335 Data mapping is used to copy transient data blocks (nonpermanent data) from current locations to locations compatible with the updated issue of data. Data mapping is accomplished in two steps:

(a) Mapping which may be completed using segment breaks can be run concurrently with system processing. This can be done without interruption of service.

(b) Mapping which will not tolerate segment breaks must be performed after normal system processing is halted.

5.336 Since file store resident data requires a relatively long time to move and since it is not per-call data, it is moved concurrently with call processing. In addition, the mapping program needs from file store can be fetched during this interval. Before the mapping can begin (for SUPL and SUFA), system update causes all queued file store write requests to be canceled and inhibits any other write requests. Before file store mapping can begin (for SUAP), SUAP inhibits all writes to the NORMAL file and sends a message to the FMI which the FMI returns when all write jobs to the NORMAL 1A_FILE are finished. Reads are permitted throughout the duration of file store mapping. After the necessary file store data mapping is completed, system update then inhibits all call processing and begins main memory mapping.

5.337 Before beginning the main memory data mapping, all file store requests are inhibited and any queued requests are canceled. In addition, all interrupts are prevent error source transmitted (pested) and will remain pested until after the system comes up on the new issue of data. Special data mapping control blocks and mapping algorithms are provided to preserve, over a system update procedure, the critical common Datapool location. Code is also provided which protects against executing requests for erroneous data mapping algorithms.

5.338 To facilitate transient main memory data mapping (and "gentle" recovery in case of mapping failure), the update program (either SUPL, SUAP, or SUFA) will configure all duplicate call stores on the active bus in the maintenance mode. Mapping may then be performed by moving data from the normal mode stores to maintenance mode stores. This scheme permits system call data in normal mode stores to be saved. If mapping fails, the mapping routine returns successfully to the update program, the mapped stores are switched to the normal mode, and the "old" stores are switched to the maintenance mode.

5.339 The system update program library provides inhibit mapping flags for both main memory and file store mapping functions. The default state of these flags permits mapping. Application programs control mapping functions by setting or resetting these flags.

H. Main Memory Overwrite

The system update program listings may use the terms "PS0" and "K-code 20" which equate to the term "program store block 0" used in this section. Likewise, the PR term "CSN" equates to the term "highest number call store" used in this section.

5.340 A full update is accomplished by pumping those stores which were affected by the update. Special care must be used in dealing with program store block 0 and the highest number call store, since these are fractionalized and have system maps and tables located in them.

5.341 For the file store, the APS update, or the APS-APS update, SUFA or SUAP uses interprocessor commands to pump program store 0. Also SUFA or SUAP sends and completes the command protocol. Since all interrupts are pested, the sanity timer must be monitored by SUAP or SUFA to avoid a premature time-out. Sufficient time is allowed for progam store block 0 to be filled. In case of errors or other difficulties in which the time allowance is exceeded, a B-level interrupt will occur with the processor configuration state counter indicating that a pump is required. For the file store to file store update, service routines which are found in program store block 0 cannot be used since all of program store block 0 will be overwritten with the new issue. The system update program library must therefore set up the fill request and check for completion or errors. Since all interrupts are pested, the sanity timer must be monitored by SUPL to avoid a premature time-out. Sufficient time is allowed for program store block 0 to be filled. In case of errors or other difficulties in which the time allowance is exceeded, a B-level interrupt will occur with the processor configuration state counter indicating that a pump is required.

5.342 For the file store update, the pump of program store block 0 is complicated because the maintenance file store data request blocks (DRBs) are located in program store block 0. To avoid overwriting the DRBs, SUPL must request a pump from the start of program store block 0 to the start of the DRBs in one segment, and from the end of the DRBs to the end of the store in another segment. To avoid problems caused by having the location of the DRBs moved from one version of Datapool to another, SUPL requires that the location of the maintenance DRBs always be in the same location in all versions of Datapool.

5.343 Since the new generic or office dependent data may require more main memory storage than the previous issues, the main memory communities must be configured before the system can be pumped. To bootstrap the main memory communities, new data found in the highest number call store must be available. Therefore, after program store block 0 has been pumped, the highest number call store is pumped. Normal DKAD program requests are used at this point since program store block 0 has been pumped. Abnormal answer dispensing must be used, however, since the executive control program is not cycling.

5.344 After the highest number call store is pumped, both program store and call store communities are bootstrapped according to data in the highest number call store. For the file store update, the remainder of the stores affected by the update are then filled from the update file stores. For the APS update, the remainder of the stores are filled from the UPDATE file using normal access requests.

I. Running on New Issue

5.345 After all stores are pumped, the system is now ready to run on the new issue of data. Since a phase will not return to the update program (SUPL, SUFA, or SUAP), several items must be taken care of before the phase. The general buffer table is released, lock-out programs are released, and a transfer is made via the PATT to the phase programs.

5.346 When the phase completes, the system begins normal processing with the new version of data. A processor configuration involving a pump at this point would bring back the old version. Assuming the system functions normally, tests are now manually run to verify that the new update is successful.

J. Copying File Store Mates

5.347 ***SUPL:*** If the MCC update key for selection of a file store for update is not altered from the initial selection, a processor configuration pump is made from the file stores containing the old issue of data. This occurs even though those files had been placed out of service. If the manual update key selection is switched after all stores are pumped, a processor configuration pump would be made from the file stores containing the new data. After confirming that the new issue of software is functioning correctly, the set of file stores containing the old issue of data must be updated and returned to service. This is accomplished by entering a restore file store message at the maintenance TTY. The file store restore routing verifies the integrity of the hardware which has been out of service. The restore calls a deferred fault recognition routine which copies data from the in-service file stores to their corresponding mates before they are placed back into service.

5.348 ***SUAP:*** If the MCC NORMAL key is not altered from its initial selection, a processor configuration pump is made from the NORMAL file which contains the old issue of data. This occurs even though the access was switched. If the UPDATE key is selected, the processor configuration pump is made from the UPDATE file which contains the new issue of data. After confirming that the new issue of software is functioning correctly, the 1A_FILE version must be renamed so that the NORMAL file contains the new data and the UPDATE file contains the old data. The UPDATE file must then be locked. This is done via the commit TTY message which causes a message to be sent to the FMI to do the rename and lock. If the update aborted or if a backout was done, another commit message can be used to just lock the UPDATE file.

5.349 ***SUFA:*** If the file store 0&2 key on the MCC panel is not altered from its initial selection, a processor configuration pump is made from the file stores which contain the old data. If the key selection is switched, ie, file store 1&3 selected, the processor configuration pump is made from the UPDATE file on the 3B disk which contains the new data. After confirming that the new issue of software is functioning correctly, the commit TTY message can be used.◀

WRITE PROTECT SOFTWARE

A. General

5.350 The write protect software prevents accidental overwrites from occurring within the protected main memory of the 1A ESS switch.

B. Write Protect Software Interface

5.351 Figure 51 is a block diagram depicting the write protect software interface.

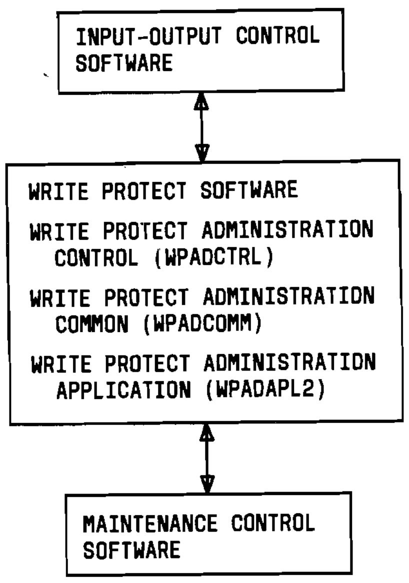

INPUT-OUTPUT CONTROL SOFTWARE
WRITE PROTECT SOFTWARE
WRITE PROTECT ADMINISTRATION CONTROL (WPADCTRL)
WRITE PROTECT ADMINISTRATION COMMON (WPADCOMM)
WRITE PROTECT ADMINISTRATION APPLICATION (WPADAPL2)
MAINTENANCE CONTROL SOFTWARE

Fig. 51—Write Protect Software Interface

C. Write Protect Software Function

5.352 The write protect software consists of three separate programs:

(a) Write protect administration control (WPADCTRL)

(b) Write protect administration common (WPADCOMM)

(c) Write protect administration application (WPADAPL2).

Write Protect Administration Control (WPADCTRL)

5.353 Pident WPADCTRL is the control program for the write protect system and is treated as a system maintenance job. It is divided into four basic routines:

- Initialization
- Service
- Maintenance
- Audit.

Initialization

5.354 The write protect mechanism is initialized either via the TTY or from any memory zeroing phase which requires a restart of the write protect program.

5.355 When a request is made for a write protect restart, all write protect jobs which are in progress or that have been requested are canceled. The 1A processor maintenance control software interfaces on base level to initialize the write protect process. A static map is built and the write protect write-read RAM is pumped.

5.356 The static map building occurs during the initialization phase where information defining the write protected addresses are extracted from the core-to-disk map and merge from the common and application maps to form a static map for all the main memory. Only those identified tags which appear in both the common and application map will appear in the static map. This map has a 1-to-1 correspondence with the write protect RAMs in the stores when there are no active clients and all protected memory is locked.

5.357 The static map consists of 1024 words. Its location in memory is restricted in that it resides in a back-up call store.

Service

5.358 The service routines service TTY client requests, provide status on the write protect system, and open locked memory blocks.

Maintenance

5.359 The functions of the maintenance routines are to inhibit writings when there is a write protect violation and inhibit audits from generating an audit message when the client memory is unlocked.

Audit

5.360 Pident WPADCTRL has a self-contained audit system. The routine is entered once every second from the maintenance software. A portion of the RAM is compared with the static map and the common associated counters to assure continuity between the hardware and software status. The status of the write protect inhibit flip-flops are compared with the call store and program store inhibit status words. The client counters are verified to assure the relocking of memory. The TTY output messages are printed to indicate the invalid RAMs and counters.

Write Protect Administration Common (WPADCOMM)

5.361 Pident WPADCOMM is a set of data tables containing common maps of client identification tags and the start and end addresses of the protected main memory. This data is used in the building of the static maps.

Write Protect Administration Application (WPADAPL2)

5.362 Pident WPADAPL2 is a set of data tables containing the application maps for the client tags and the start and end addresses of the protected main memory. This data is used in the building of the static maps.

6. INTERRUPT RECOVERY SOFTWARE FUNCTIONS

6.01 Comprehensive maintenance software is required to meet the system reliability objective of an average of less than 2 minutes per year of outage from all causes. The processor depends on integrating maintenance software with the hardware to (1) quickly recognize a fault condition, (2) isolate and configure around the faulty subsystem, (3) diagnose the faulty unit without interfering with normal processor functions, and (4) assist the maintenance personnel in locating and correcting the fault. The function and interrelationships of seven recovery programs (interrupt recovery software, system reinitialization, hardware recovery, emergency mode, processor recovery, error analysis, and software initialization) are given here.

INTERRUPT RECOVERY SOFTWARE CONTROL

A. General

6.02 When the processor is interrupted by an error condition, the processor central control performs a hardwired transfer from its current program address to the appropriate entry point in the software interrupt control structure. The corresponding fault recovery program then configures a working system of hardware and returns control to normal base level processing. These programs may also schedule deferred testing (fault recovery or diagnostics) on implicated units.

6.03 The ESS switch is designed to minimize the effect of maintenance activities on call processing. System maintenance features include:

- Duplication of key units
- Switchable spares for some units
- Matching of signals between duplicated units
- Unique error indicators within the equipment
- Software controlled access into various units for hardware testing
- Communication checks between units
- A multilevel maintenance interrupt structure
- System reconfiguration mechanisms.

6.04 Program initiated communications between the central control and other units are protected by parity signals and response verifications called all-seems-well (ASW) signals. The parity signals accompany the communication from the central control to the unit and the ASW signal is returned to central control if the communication is successful. In several instances an ASW failure signal is also returned. The return of the ASW signal indicates that parity checks and additional internal checks by the unit were successful. The return of an ASW failure signal indicates that some unit or a bus system detected an error. In addition, if the unit is to return data to central control, the unit may send parity to accompany the returned data. A failure of any of the checks may result in a system interrupt.

B. Central Control Features

6.05 Communication paths between central controls (active and standby) are used for matching of operations, for passing of error signals, and for testing. Central control communications with other units over call store buses, program store buses, and auxiliary unit buses are matched. The peripheral unit reply parity is matched between central controls. Certain other internal points in the central controls are also matched. Detection of a mismatch results in an interrupt of the normal system control.

6.06 The central controls normally operate in step, but perform checks separately. For lower priority interrupts (F through J) the central control, which detects an error and initiates an interrupt or some other actions, cross-couples a signal to the other central control in order to keep it in step. For the higher priority interrupts (A through E), the standby central control is stopped but signals are still cross-coupled.

6.07 It is also possible for one central control to test the other under program control. For these tests, there is an access bus between the central controls. The bus provides the capability for the active central control to address 128 different maintenance access locations in the out-of-service central control for read or write operations. There is a 24-bit read bus and a 24-bit write bus between the data buffer registers of the two central controls in order to allow sending and receiving of data to accomplish the tests.

C. Interrupt Hierarchy

6.08 The interrupt structure (Table D) is a hierarchy of interrupt levels that are entered according to the severity of the problem encountered by the system. Manual actions to keep the system in operation are given the highest priority (A level).

6.09 The sources of the nine interrupt levels are summarized in Table D. The interrupt levels have been placed into four categories. The H- and J-level interrupts are used for input/output related tasks. The J-level interrupt is activated every 5 milliseconds by a system clock. Normally, the J-level tasks are executed and program control is returned to base level before the next 5-millisecond interrupt occurs. However, if the J-level tasks are not completed at the next 5-millisecond interrupt, an H-level interrupt occurs. The high priority J-level tasks are then completed before a return is made to J level.

6.10 A higher priority interrupt can interrupt a lower priority interrupt which is in progress. The only exception is the B-level interrupt which can interrupt an A level as well as all lower priority interrupts. For example, a J-level interrupt may be interrupted by a D-level interrupt. The D-level interrupt can, similarly, be interrupted by a B-level interrupt. Interrupt levels D, E, F, G, H, and J and the generate control pulse (GCP) source for B level may be inhibited during system recovery.

D. General Approach to All Interrupts

6.11 The relationship between interrupt level processing and normal system processing is illustrated in Fig. 52. During normal base level processing the bulk of the ESS switch programs, both call processing and maintenance, is executed.

6.12 An interrupt may occur at any time during system operation. The hardware interrupt sequencers in central control may be triggered either manually (for A or B level) or automatically by hardware. As a part of the hardware sequencer action, a wired transfer (not under program control) is made to the appropriate entry point in the processor system interrupt recovery program (SIRE) for the No. 1A. Then SIRE stores a basic set of data that may be useful for restarting base level processing after an interrupt recovery. This data is stored at memory locations (interrupt bins) that are assigned to each interrupt level.

6.13 After SIRE has stored the required data in the appropriate interrupt bin, a program transfer is made to an interrupt associated filter program. This program determines the primary source of the interrupt (see Table D) and the units involved. After the basic source of the interrupt is resolved, the appropriate fault recovery programs are entered.

6.14 The fault recovery programs are designed to isolate faulty units or subsystem rather than to identify replaceable components. These programs recognize and isolate most call-affecting faults during a single interrupt interval.

A-Level Interrupt

6.15 The A-level interrupt is manually initiated from the MCC. There are two ways to manually initiate an A-level interrupt:

TABLE D

1A PROCESSOR INTERRUPT SOURCES

TYPE OF INTERRUPT	LEVEL OF INTERRUPT	SOURCE OF INTERRUPT
System Configuration	A	Activated Manually From Master Control Console (MCC)
	B	Processor Configuration Sequencer
		Program Request to Switch Active Central Controls
Fault Detection	B	Generate Control Pulse (GCP) Failure
	C	Mismatch of Data Between Central Controls
	D	Failure to Access Call Store
		Failure to Access Auxiliary Unit
		Protected Address Range Write Violation
		Underflow/Overflow of the Program Stack Counter
		Transfer to an Address Outside the Program Store Range Without the Call Store Program Flip-Flop Set
	E	Failure to Access Program Store
	F	Peripheral Unit Failure
Testing	G	Interval Timing or Match Testing
Processing	H	Interrupt J Level After 5 or 10 ms
	J	Interrupt Periodically After 5 or 10 ms

- Manual interrupt program requests
- Override control.

6.16 First, for manual interrupt program requests (such as a memory initialization phase), the manual activation of an MCC key causes a hardwire transfer to the A-level interrupt entry to SIRE. See Fig. 53 for a block diagram of A-level processing and program flow interfaces.

6.17 Second, if a manual processor configuration circuit activation is initiated from the MCC override control panel, there are two possible valid overrides. The override may select a new active central control or a basic processor which consists of a new active central control, program store bus, auxiliary unit bus, and program store 0. If the override involves only a new active central control, then the activation of the override causes a hardware transfer to the A-level entry to SIRE. If the manual override selects a basic processor, then a hardware initiated "pump" of program store 0 is performed before program control is transferred to the processor configuration recovery program instead of SIRE.

6.18 Third, if the processor configuration has been unsuccessful (ie, file data and program store data are multilated or certain catastrophic hardware faults have occurred), then a manual system reinitialization may be activated from the MCC system reinitialization panel. The system reinitialization allows the program to be copied from tape or a data link (instead of file store) prior to attempting a processor configuration recovery.

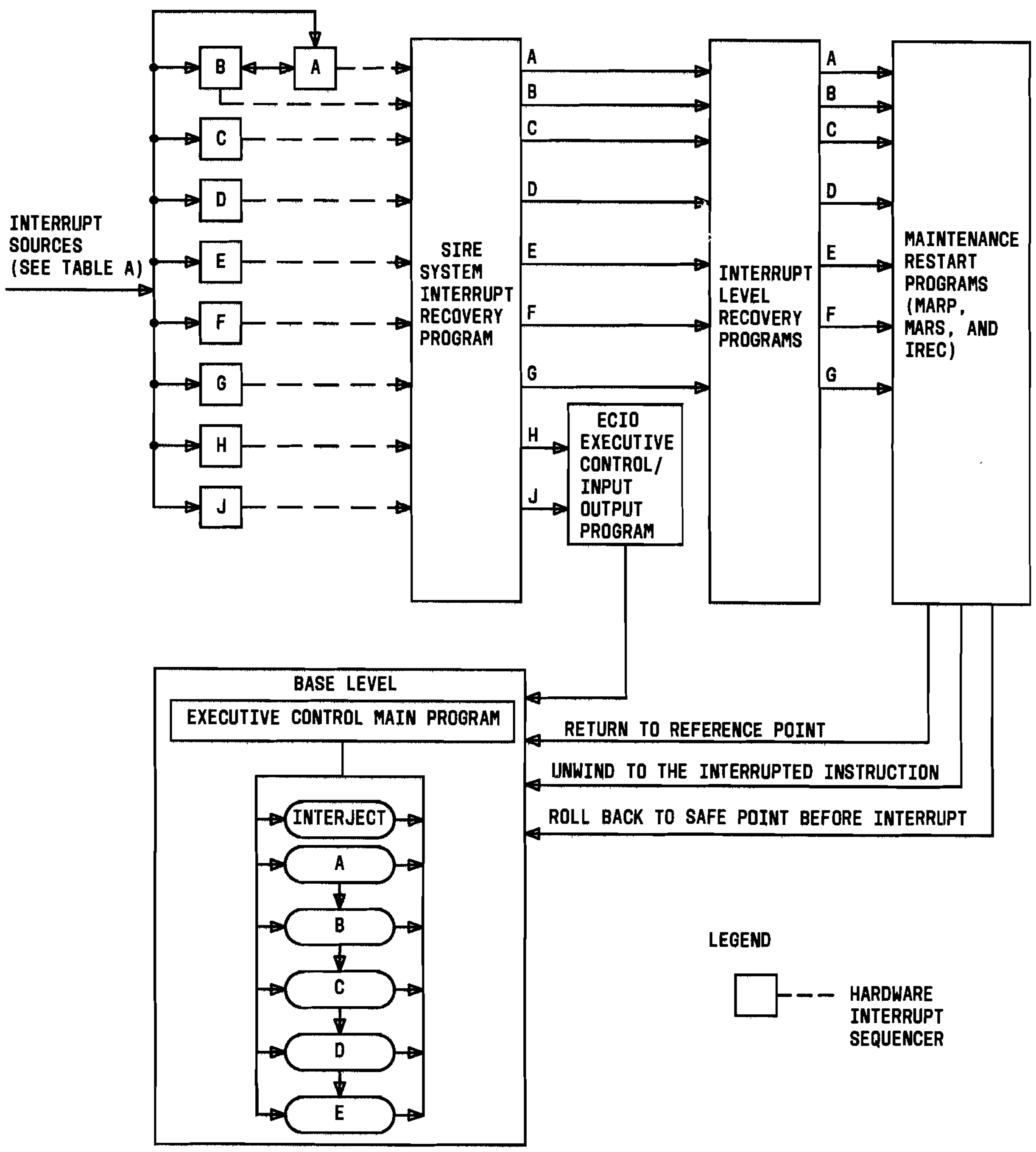

INTERRUPT SOURCES (SEE TABLE A)
A
B
C
D
E
F
G
H
J
SIRE SYSTEM INTERRUPT RECOVERY PROGRAM
ECIO EXECUTIVE CONTROL/ INPUT OUTPUT PROGRAM
INTERRUPT LEVEL RECOVERY PROGRAMS
MAINTENANCE RESTART PROGRAMS (MARP, MARS, AND IREC)
BASE LEVEL
EXECUTIVE CONTROL MAIN PROGRAM
INTERJECT
RETURN TO REFERENCE POINT
UNWIND TO THE INTERRUPTED INSTRUCTION
ROLL BACK TO SAFE POINT BEFORE INTERRUPT
LEGEND
HARDWARE INTERRUPT SEQUENCER

Fig. 52—General Interrupt Structure

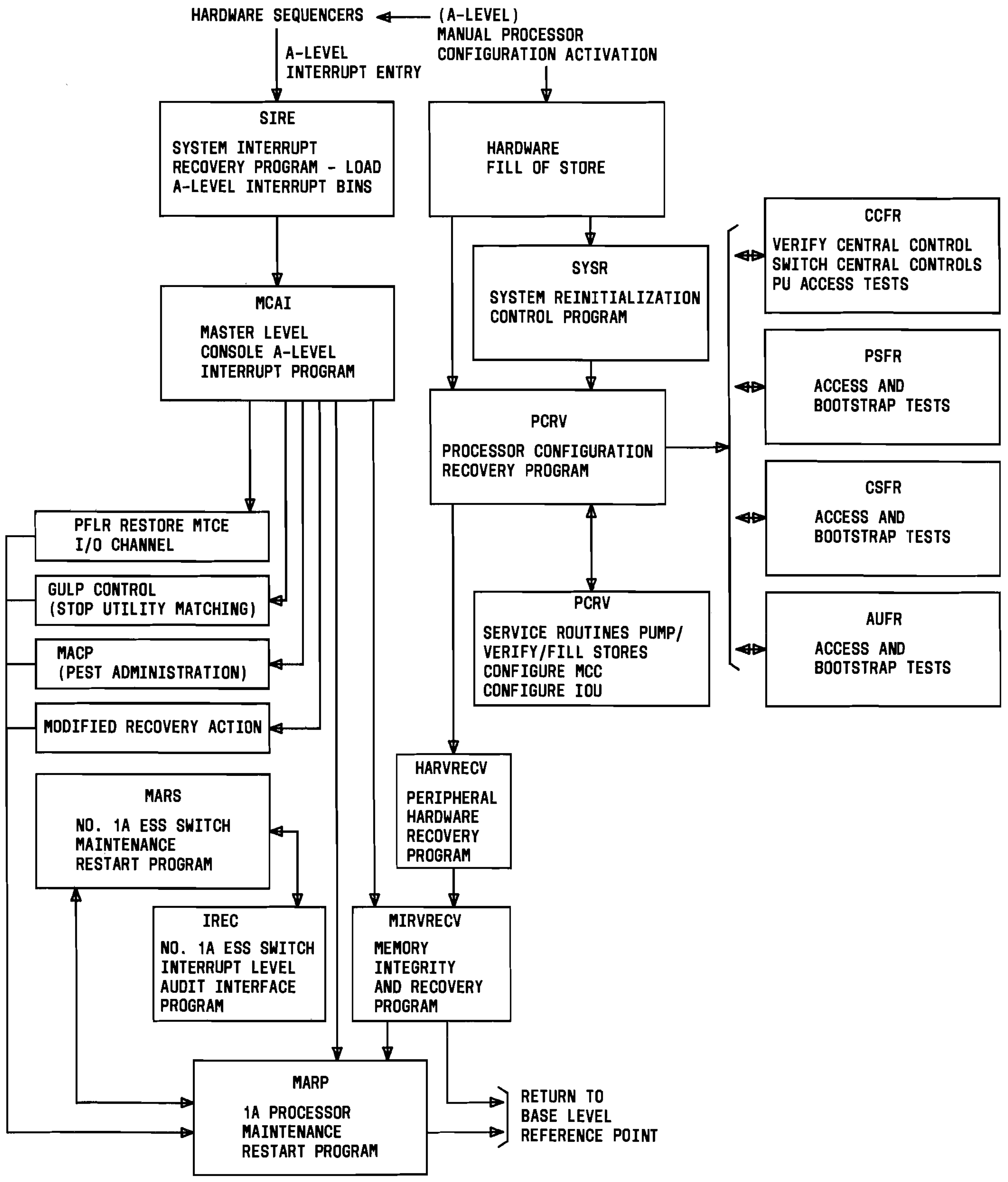

HARDWARE SEQUENCERS
(A-LEVEL) MANUAL PROCESSOR CONFIGURATION ACTIVATION
A-LEVEL INTERRUPT ENTRY
SIRE SYSTEM INTERRUPT RECOVERY PROGRAM - LOAD A-LEVEL INTERRUPT BINS
HARDWARE FILL OF STORE
SYSR SYSTEM REINITIALIZATION CONTROL PROGRAM
MCAI MASTER LEVEL CONSOLE A-LEVEL INTERRUPT PROGRAM
PCRV PROCESSOR CONFIGURATION RECOVERY PROGRAM
CCFR VERIFY CENTRAL CONTROL SWITCH CENTRAL CONTROLS PU ACCESS TESTS
PSFR ACCESS AND BOOTSTRAP TESTS
CSFR ACCESS AND BOOTSTRAP TESTS
AUFR ACCESS AND BOOTSTRAP TESTS
PFLR RESTORE MTCE I/O CHANNEL
GULP CONTROL (STOP UTILITY MATCHING)
MACP (PEST ADMINISTRATION)
MODIFIED RECOVERY ACTION
PCRV SERVICE ROUTINES PUMP/ VERIFY/FILL STORES CONFIGURE MCC CONFIGURE IOU
HARVRECV PERIPHERAL HARDWARE RECOVERY PROGRAM
MARS NO. 1A ESS SWITCH MAINTENANCE RESTART PROGRAM
IREC NO. 1A ESS SWITCH INTERRUPT LEVEL AUDIT INTERFACE PROGRAM
MIRVRECV MEMORY INTEGRITY AND RECOVERY PROGRAM
MARP 1A PROCESSOR MAINTENANCE RESTART PROGRAM
RETURN TO BASE LEVEL REFERENCE POINT

Fig. 53—A-Level Processing—Program Flow and Interfaces

B-Level Interrupt

6.19 The B-level interrupt (Fig. 54) is initiated through automatic processor configuration circuit triggers, program requests to switch active central controls, or GCP failures.

6.20 B-level routines are normally requested automatically by the processor configuration sequencer circuitry; however, they can be entered by manual request at the MCC. The processor configuration circuit establishes a basic processor (central control, basic program store block, and program store bus) and initiates a program sequence which recovers all processor subsystems and may verify and correct transient data.

6.21 This total system recovery is attempted in two levels. The first level is associated with the first 16 states of the processor configuration circuit and attempts to recover call-processing capabilities as quickly as possible. Speed is achieved in this level by attempting to execute the recovery programs as they exist in the program store. That is, file store data backup will not be used by the basic recovery programs. The second level of recovery is associated with the remaining 32 processor configuration circuit states. In this case, level 1 recovery has failed and the primary consideration is forcing recovery and not speed. The probability of recovery is increased in level 2 recovery by using the file store data backup copy for the basic recovery programs.

C-Level Interrupt

6.22 The C-level interrupt (Fig. 55) is initiated by a mismatch of data between the active and the standby central controls when the central controls are operating in step. A C-level interrupt causes the central control hardware sequencer in the active central control to transfer to the C-level entry of SIRE. C-level interrupt routines attempt to verify the integrity of the active central control. If the integrity of the active central control cannot be established, a switch of central controls is performed, and a B-level interrupt is initiated.

D-Level Interrupt

6.23 A D-level interrupt (Fig. 56) may be caused by any of several sources. The sources are as follows:

- Failure to access call store
- Failure to access auxiliary units
- Range errors during call store accesses
- Stack counter underflow/overflow
- Protected area violations
- Program transfers to call store without call store program flip-flop set.

Since these sources involve two subsystems, the call store and the auxiliary unit subsystems, two levels of interrupt source filtering are used.

6.24 If the source filter indicates a call store failure, the call store recovery routines perform initial D-level interrupt source filtering and administer call store fault recovery. If the source filter indicates that the interrupt is due to an auxiliary unit access failure, the control is transferred to the auxiliary unit recovery routines. If the source filter indicates that the interrupt is due to a range error, a stack counter error, a protected area violation, or a program transfer to call store without the flip-flop set, then the interrupt is probably a result of program operational errors.

E-Level Interrupt

6.25 Failure to access a program store will cause an E-level interrupt (Fig. 57). After the interrupt, control is passed to SIRE which stores the appropriate data in the E-level interrupt bins. E-level entry into the fault recovery program is via source filter and task dispenser routine. The routine determines the source of the interrupt and selects the appropriate service and test routines to resolve the problem. When the trouble is located in duplicated program store, the suspect unit is removed from service, and the remaining unit is set to operate as if it is not duplicated. After a configuration of program stores has been selected, an access test is performed on each memory block to verify the integrity of the program store community.

F-Level Interrupt

6.26 The source of F-level interrupts (Fig. 58) is the peripheral unit failures. Peripheral unit failures are caused by faulty peripheral units, faulty peripheral unit buses, or central control matching check failure. When a peripheral unit is addressed,

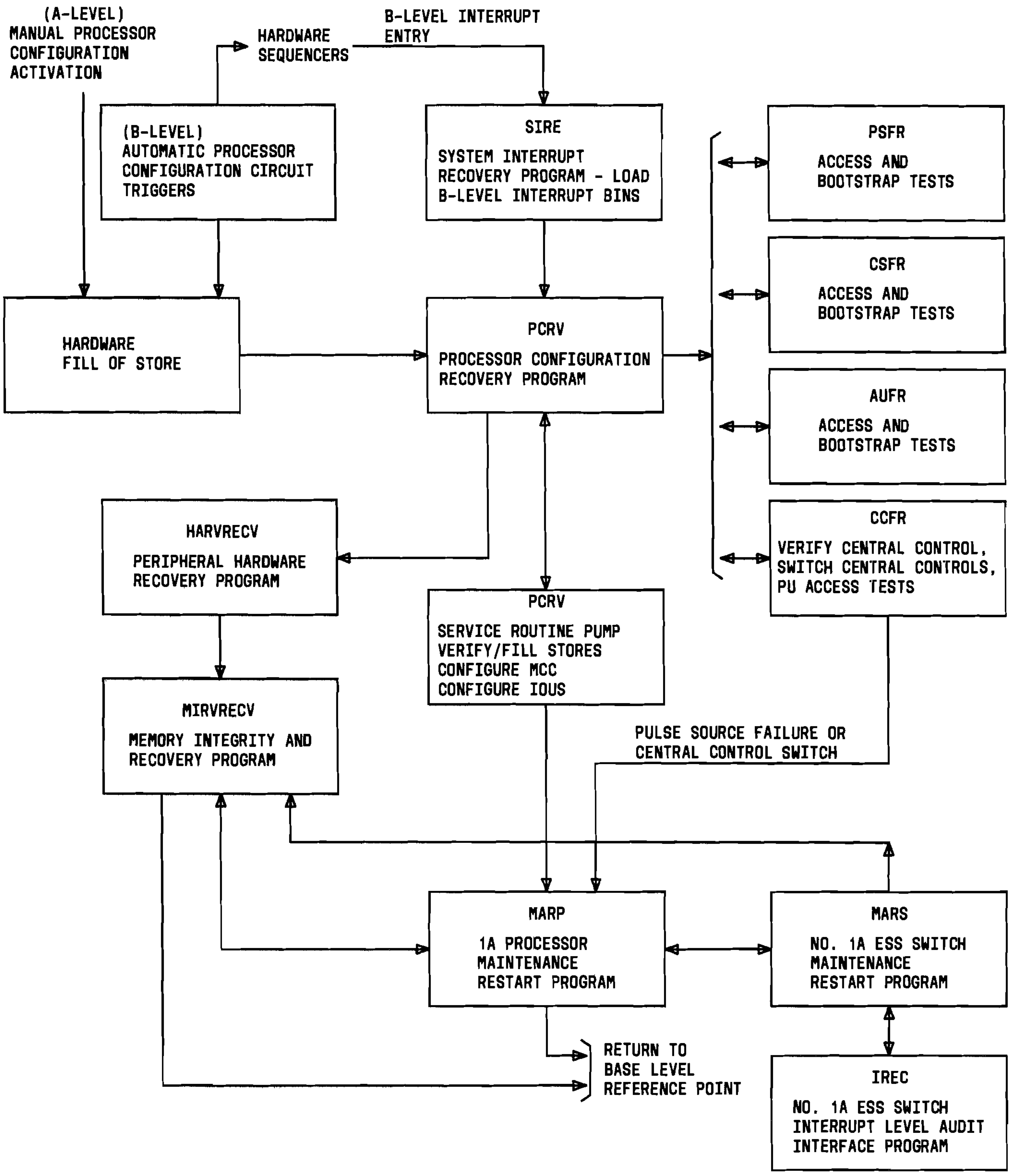

(A-LEVEL)
MANUAL PROCESSOR CONFIGURATION ACTIVATION
HARDWARE SEQUENCERS
B-LEVEL INTERRUPT ENTRY
(B-LEVEL) AUTOMATIC PROCESSOR CONFIGURATION CIRCUIT TRIGGERS
SIRE
SYSTEM INTERRUPT RECOVERY PROGRAM - LOAD B-LEVEL INTERRUPT BINS
PSFR
ACCESS AND BOOTSTRAP TESTS
CSFR
ACCESS AND BOOTSTRAP TESTS
HARDWARE FILL OF STORE
PCRV
PROCESSOR CONFIGURATION RECOVERY PROGRAM
AUFR
ACCESS AND BOOTSTRAP TESTS
HARVRECV
PERIPHERAL HARDWARE RECOVERY PROGRAM
CCFR
VERIFY CENTRAL CONTROL, SWITCH CENTRAL CONTROLS, PU ACCESS TESTS
PCRV
SERVICE ROUTINE PUMP VERIFY/FILL STORES CONFIGURE MCC CONFIGURE IOUS
MIRVRECV
MEMORY INTEGRITY AND RECOVERY PROGRAM
PULSE SOURCE FAILURE OR CENTRAL CONTROL SWITCH
MARP
1A PROCESSOR MAINTENANCE RESTART PROGRAM
MARS
NO. 1A ESS SWITCH MAINTENANCE RESTART PROGRAM
RETURN TO BASE LEVEL REFERENCE POINT
IREC
NO. 1A ESS SWITCH INTERRUPT LEVEL AUDIT INTERFACE PROGRAM

Fig. 54—B-Level Processing—Program Flow and Interfaces

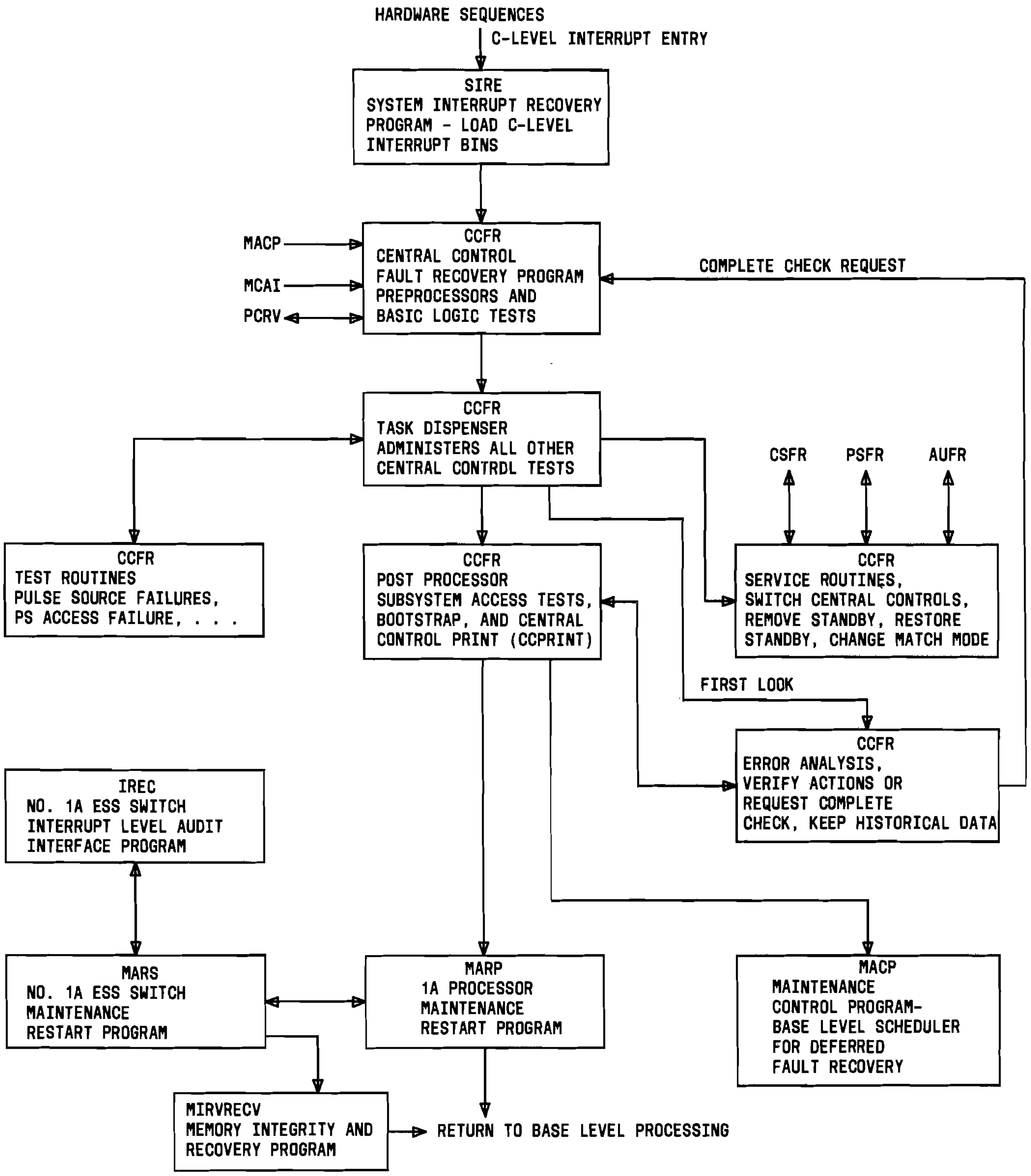

HARDWARE SEQUENCES
C-LEVEL INTERRUPT ENTRY
SIRE
SYSTEM INTERRUPT RECOVERY PROGRAM - LOAD C-LEVEL INTERRUPT BINS
MACP
MCAI
PCRV
CCFR
CENTRAL CONTROL FAULT RECOVERY PROGRAM PREPROCESSORS AND BASIC LOGIC TESTS
COMPLETE CHECK REQUEST
CCFR
TASK DISPENSER ADMINISTERS ALL OTHER CENTRAL CONTRDL TESTS
CSFR
PSFR
AUFR
CCFR
TEST ROUTINES PULSE SOURCE FAILURES, PS ACCESS FAILURE, . . .
CCFR
POST PROCESSOR SUBSYSTEM ACCESS TESTS, BOOTSTRAP, AND CENTRAL CONTROL PRINT (CCPRINT)
CCFR
SERVICE ROUTINES, SWITCH CENTRAL CONTROLS, REMOVE STANDBY, RESTORE STANDBY, CHANGE MATCH MODE
FIRST LOOK
CCFR
ERROR ANALYSIS, VERIFY ACTIONS OR REQUEST COMPLETE CHECK, KEEP HISTORICAL DATA
IREC
NO. 1A ESS SWITCH INTERRUPT LEVEL AUDIT INTERFACE PROGRAM
MARS
NO. 1A ESS SWITCH MAINTENANCE RESTART PROGRAM
MARP
1A PROCESSOR MAINTENANCE RESTART PROGRAM
MACP
MAINTENANCE CONTROL PROGRAM- BASE LEVEL SCHEDULER FOR DEFERRED FAULT RECOVERY
MIRVRECV
MEMORY INTEGRITY AND RECOVERY PROGRAM
RETURN TO BASE LEVEL PROCESSING

Fig. 55—C-Level Processing—Program Flow and Interfaces

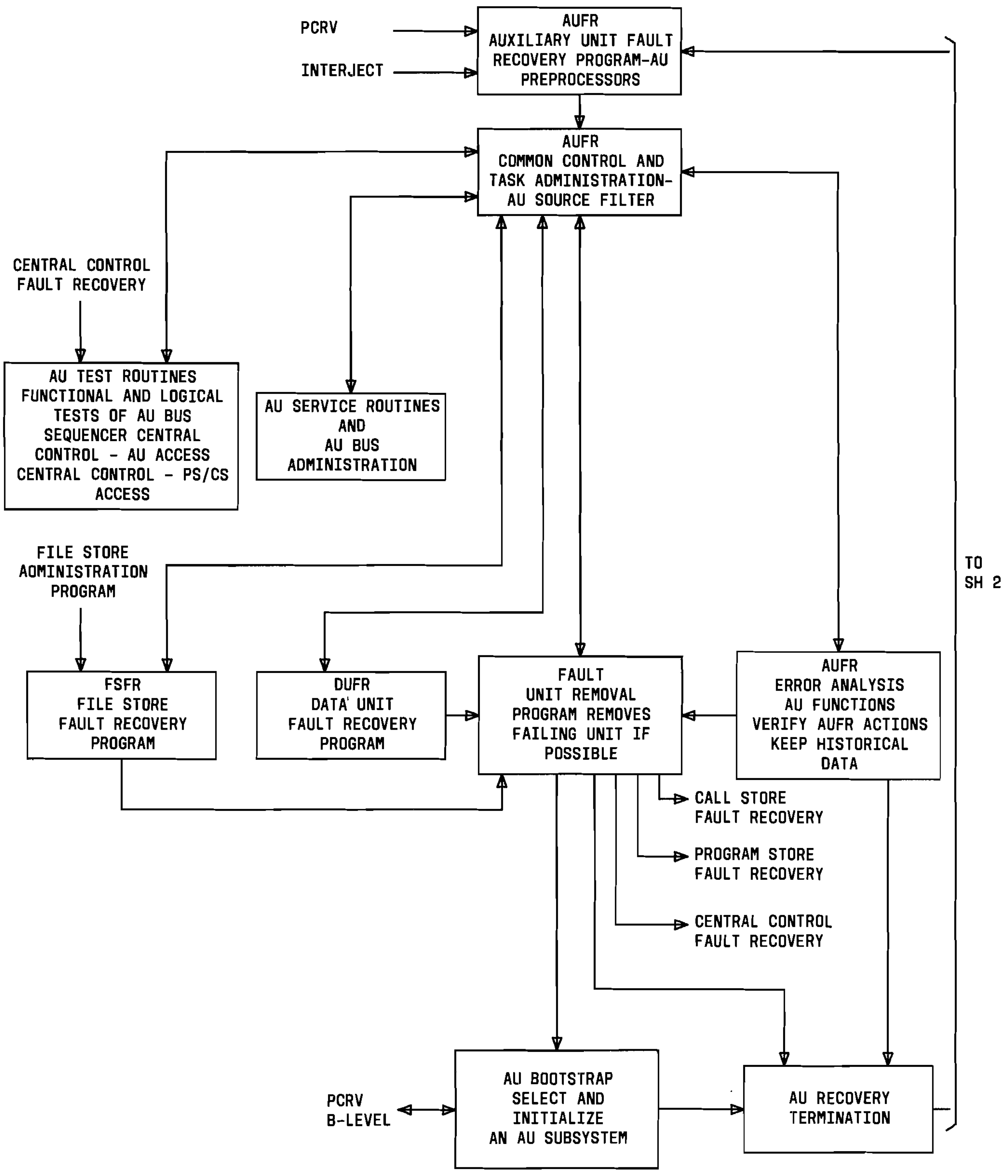

PCRV
INTERJECT
AUFR AUXILIARY UNIT FAULT RECOVERY PROGRAM-AU PREPROCESSORS
AUFR COMMON CONTROL AND TASK ADMINISTRATION-AU SOURCE FILTER
CENTRAL CONTROL FAULT RECOVERY
AU TEST ROUTINES FUNCTIONAL AND LOGICAL TESTS OF AU BUS SEQUENCER CENTRAL CONTROL - AU ACCESS CENTRAL CONTROL - PS/CS ACCESS
AU SERVICE ROUTINES AND AU BUS ADMINISTRATION
FILE STORE ADMINISTRATION PROGRAM
FSFR FILE STORE FAULT RECOVERY PROGRAM
DUFR DATA UNIT FAULT RECOVERY PROGRAM
FAULT UNIT REMOVAL PROGRAM REMOVES FAILING UNIT IF POSSIBLE
AUFR ERROR ANALYSIS AU FUNCTIONS VERIFY AUFR ACTIONS KEEP HISTORICAL DATA
TO SH 2
CALL STORE FAULT RECOVERY
PROGRAM STORE FAULT RECOVERY
CENTRAL CONTROL FAULT RECOVERY
PCRV B-LEVEL
AU BOOTSTRAP SELECT AND INITIALIZE AN AU SUBSYSTEM
AU RECOVERY TERMINATION

Fig. 56—D-Level Processing—Program Flow and Interfaces (Sheet 1 of 2)

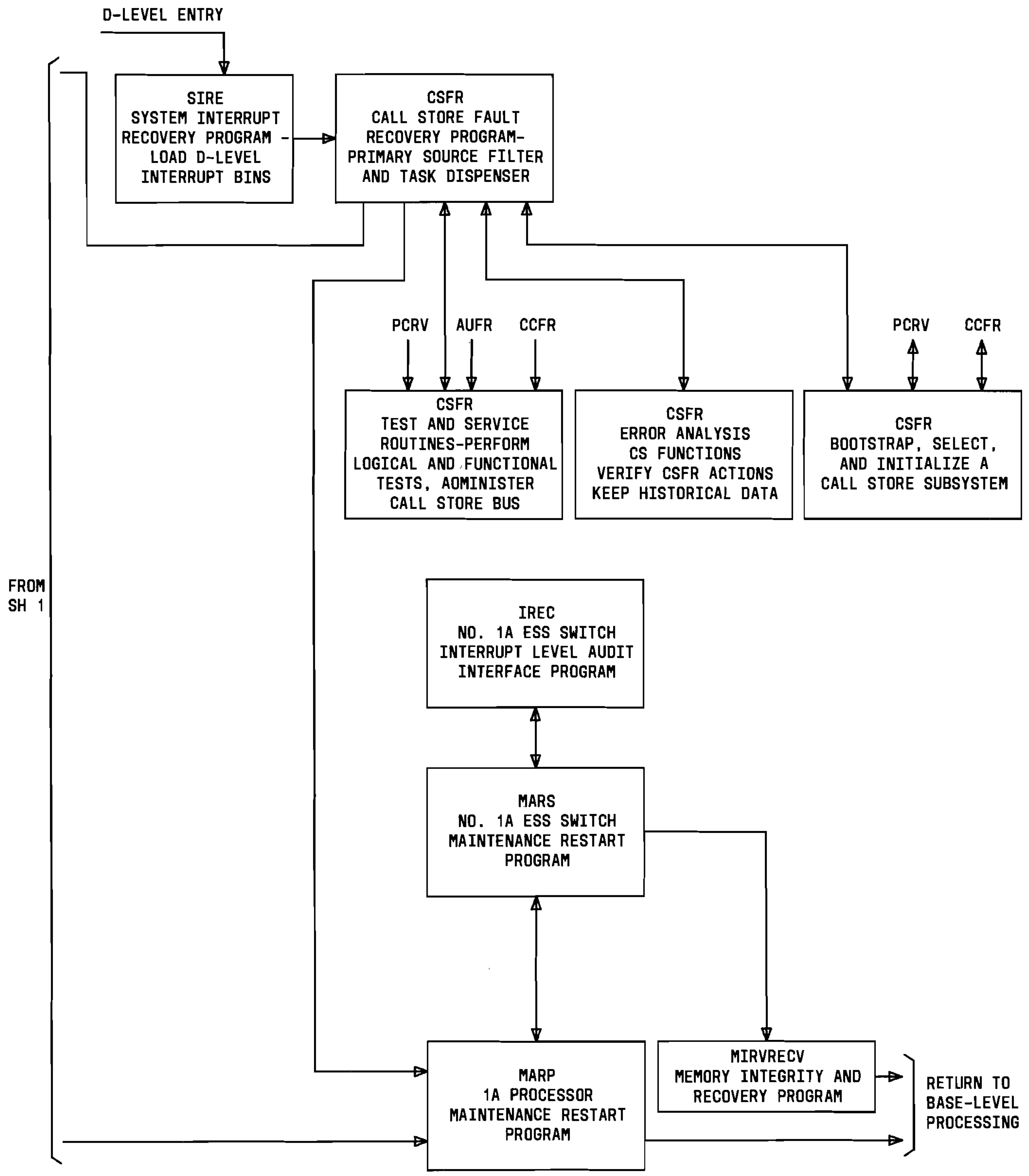

D-LEVEL ENTRY
SIRE SYSTEM INTERRUPT RECOVERY PROGRAM - LOAD D-LEVEL INTERRUPT BINS
CSFR CALL STORE FAULT RECOVERY PROGRAM- PRIMARY SOURCE FILTER AND TASK DISPENSER
PCRV
AUFR
CCFR
CSFR TEST AND SERVICE ROUTINES-PERFORM LOGICAL AND FUNCTIONAL TESTS, ADMINISTER CALL STORE BUS
CSFR ERROR ANALYSIS CS FUNCTIONS VERIFY CSFR ACTIONS KEEP HISTORICAL DATA
PCRV
CCFR
CSFR BOOTSTRAP, SELECT, AND INITIALIZE A CALL STORE SUBSYSTEM
FROM SH 1
IREC NO. 1A ESS SWITCH INTERRUPT LEVEL AUDIT INTERFACE PROGRAM
MARS NO. 1A ESS SWITCH MAINTENANCE RESTART PROGRAM
MARP 1A PROCESSOR MAINTENANCE RESTART PROGRAM
MIRVRECV MEMORY INTEGRITY AND RECOVERY PROGRAM
RETURN TO BASE-LEVEL PROCESSING

Fig. 56—D-Level Processing—Program Flow and Interfaces (Sheet 2 of 2)

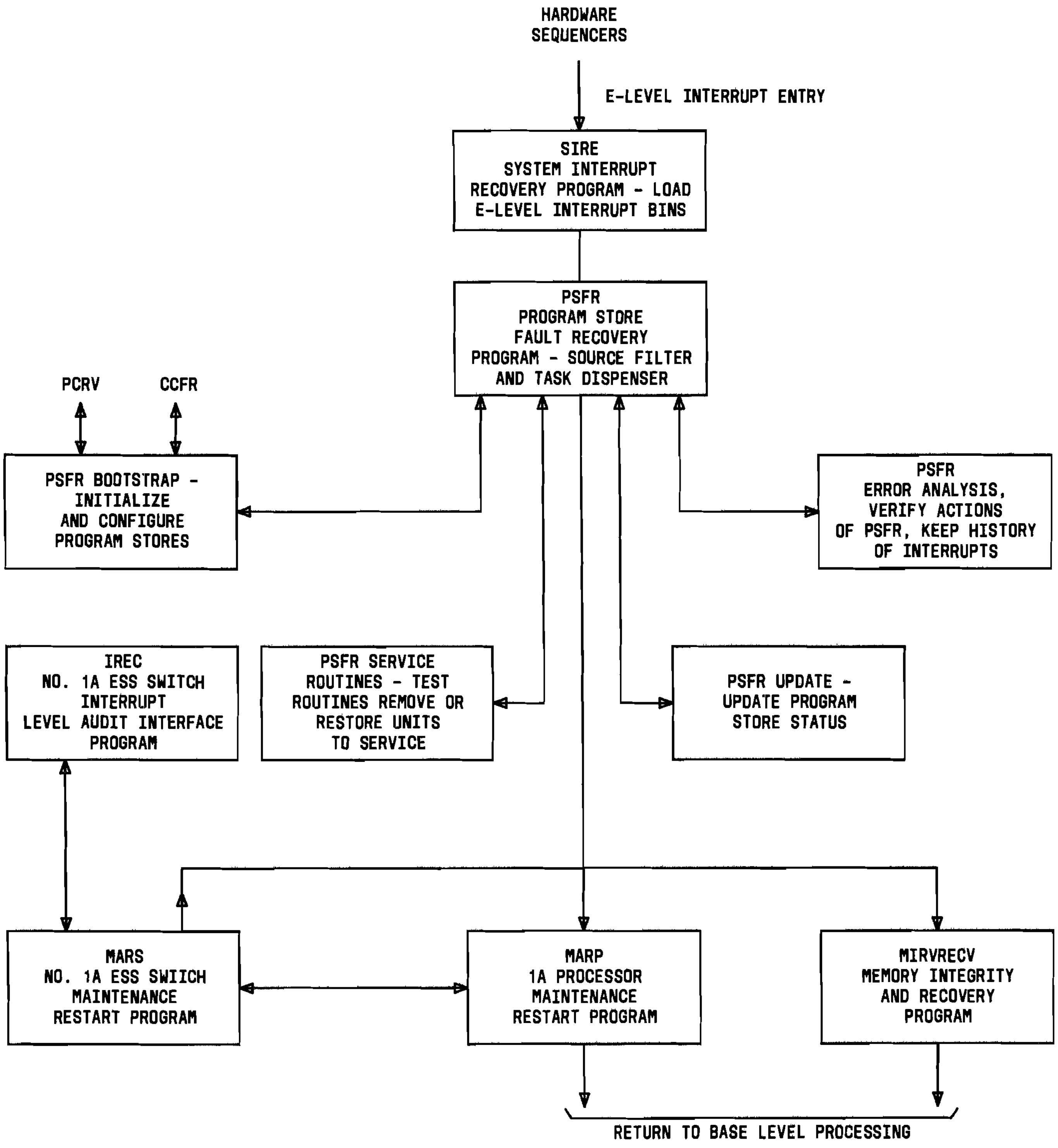

HARDWARE SEQUENCERS
E-LEVEL INTERRUPT ENTRY
SIRE SYSTEM INTERRUPT RECOVERY PROGRAM - LOAD E-LEVEL INTERRUPT BINS
PSFR PROGRAM STORE FAULT RECOVERY PROGRAM - SOURCE FILTER AND TASK DISPENSER
PCRV
CCFR
PSFR BOOTSTRAP - INITIALIZE AND CONFIGURE PROGRAM STORES
PSFR ERROR ANALYSIS, VERIFY ACTIONS OF PSFR, KEEP HISTORY OF INTERRUPTS
IREC NO. 1A ESS SWITCH INTERRUPT LEVEL AUDIT INTERFACE PROGRAM
PSFR SERVICE ROUTINES - TEST ROUTINES REMOVE OR RESTORE UNITS TO SERVICE
PSFR UPDATE - UPDATE PROGRAM STORE STATUS
MARS NO. 1A ESS SWICH MAINTENANCE RESTART PROGRAM
MARP 1A PROCESSOR MAINTENANCE RESTART PROGRAM
MIRVRECV MEMORY INTEGRITY AND RECOVERY PROGRAM
RETURN TO BASE LEVEL PROCESSING

Fig. 57—E-Level Processing—Program Flow and Interfaces

central control examines the response signals to determine if the order instruction has been successfully executed. If these signals (enable verify and all-seems-well signals) are not received, a hardwired sequencer causes an F-level interrupt. Control is passed to SIRE which stores the appropriate data in the F-level interrupt bins. A filtering process routes program control to the appropriate fault recovery program according to the type of F-level interrupt. The F-level recovery routine contains the fault recognition and recovery routines for the interrupt in these cases; it can remove an input/output unit (IOU) or MCC from service and administer the appropriate diagnostic.

G-Level Interrupt

6.27 Interrupt sources for G-level interrupts (Fig. 59) are set by special facilities of utility matching tests and by interval timing functions of the maintenance program. When such sources are set, program control is given to SIRE at the appropriate location to store the necessary information and transfer to interrupt source filter. Control is then transferred to the execution program which performs the utility function, if desired, or to maintenance delay timing routines.

H- and J-Level Interrupts

6.28 The H- and J-levels are used for call processing. See Fig. 60 for program flow. Input and output-related tasks are associated with these levels. A J-level interrupt is normally triggered by a clock interrupt which occurs every 5 milliseconds. Program control is then transferred via hardware to SIRE which stores the appropriate data in the J-level interrupt bins. The J-level tasks are executed and program control is returned to base level programming. If the J-level tasks are not completed before the next 5-millisecond clock interrupt, an H-level interrupt is triggered by the clock interrupt. Program control is then transferred via hardware to SIRE. The SIRE program stores the appropriate data in the H-level interrupt bins and the high priority H-level tasks are executed. After these tasks are executed, the interrupted J-level tasks are executed. Then, program control is returned to the base level program.

Base Level Maintenance

6.29 During base level processing the auxiliary unit and peripheral unit community may request maintenance on base level without disrupting service.

Error Stop

6.30 When the generic program is loaded in main memory, there are some spaces left between programs creating "holes" in main memory, sometimes referred to as "fill areas." Software errors may occur when a transfer of program control is to such a location in main memory, resulting in an insane condition. Provisions are made to correct such a condition by writing instructions in each of these locations to transfer control to the error stop routine in SIRE (see Fig. 61).

Interject

6.31 Interject processing occurs when a unit detects an internal trouble of noninterrupt priority and sets an interject request bit. This bit is checked on regular intervals and when it is set, control is transferred to the proper entry in SIRE. There are four levels of interject processing, each having a fixed address entry to SIRE. Figure 62 is a diagram of interject processing flow and interfaces.

SYSTEM REINITIALIZATION

A. General

6.32 The system reinitialization (SR) programs are provided to recover the 1A ESS switch processing capabilities when file data are mutilated. These programs are also used to load the generic program and office data (ie, translations and parameters) upon initial installation of an office.

6.33 When faults occur in the system and fault recovery programs and software sanity programs are unable to handle them, SR programs are called upon to revive the system. These actions are entirely under manual control from the MCC. An SR action constitutes an entire reload of both main memory data and file data from backup tape-stored data or disk files. The filling of core using the tape-stored data takes several minutes and includes zeroing all transient data and initializing all unit status. Thus an inadvertent or premature initiation of an SR would be detrimental to call processing.

6.34 An SR procedure appears in two parts. First, all main memory and file stores are filled.

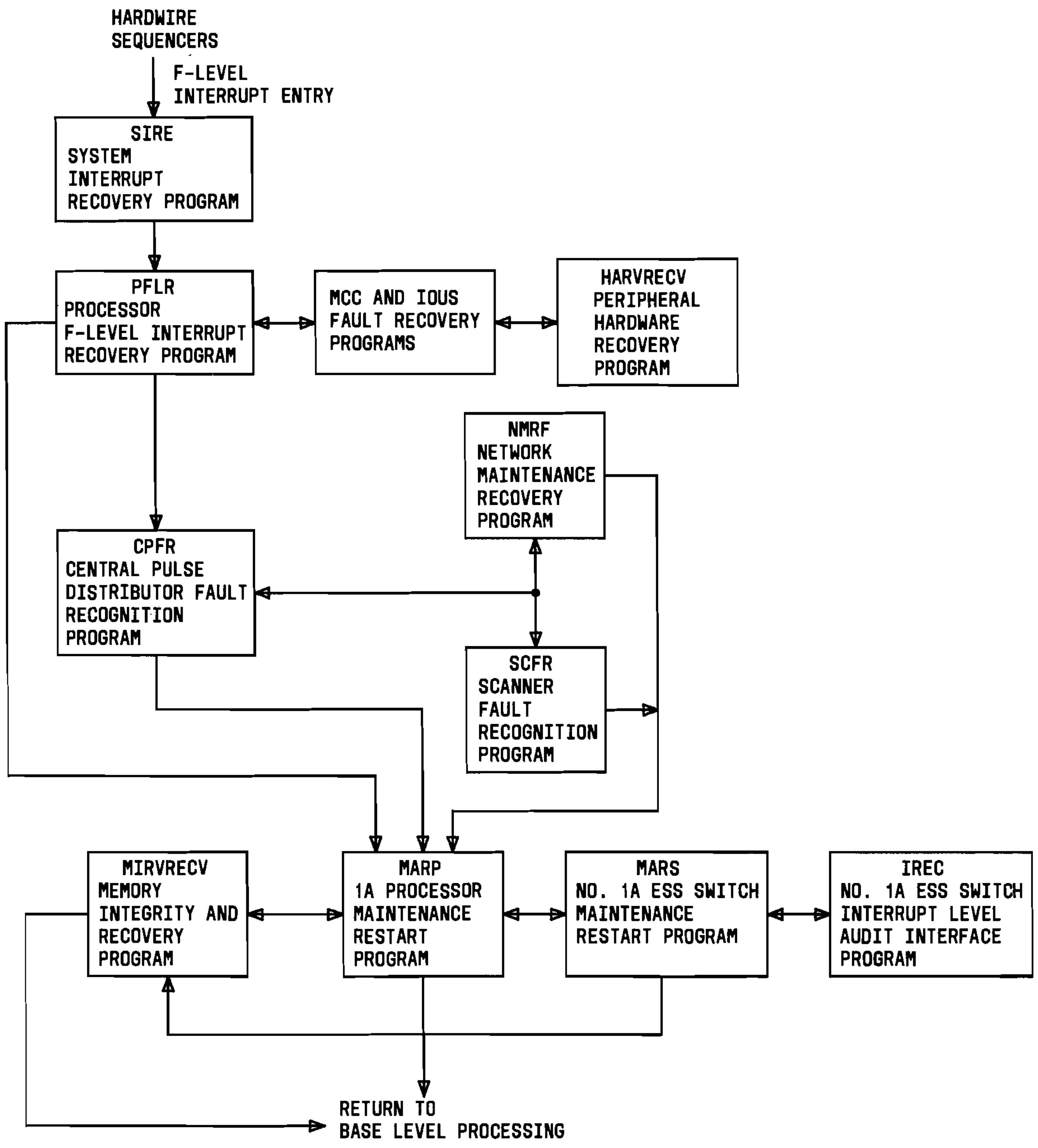

HARDWIRE SEQUENCERS
F-LEVEL INTERRUPT ENTRY
SIRE SYSTEM INTERRUPT RECOVERY PROGRAM
PFLR PROCESSOR F-LEVEL INTERRUPT RECOVERY PROGRAM
MCC AND IOUS FAULT RECOVERY PROGRAMS
HARVRECV PERIPHERAL HARDWARE RECOVERY PROGRAM
NMRF NETWORK MAINTENANCE RECOVERY PROGRAM
CPFR CENTRAL PULSE DISTRIBUTOR FAULT RECOGNITION PROGRAM
SCFR SCANNER FAULT RECOGNITION PROGRAM
MIRVRECV MEMORY INTEGRITY AND RECOVERY PROGRAM
MARP 1A PROCESSOR MAINTENANCE RESTART PROGRAM
MARS NO. 1A ESS SWITCH MAINTENANCE RESTART PROGRAM
IREC NO. 1A ESS SWITCH INTERRUPT LEVEL AUDIT INTERFACE PROGRAM
RETURN TO BASE LEVEL PROCESSING

Fig. 58—F-Level Processing—Program Flow and Interfaces

Then various tests are performed to check the validity of the data just read from tape.

B. Memory and File Store Load

6.35 The SR is initially under control of hardware sequencers since core data is assumed to be multilated. The sequencers are activated by keys at the MCC and the tape unit controller (TUC) and are therefore controlled by the operating personnel. The data unit selector system reinitialization sequencer aborts all jobs in process, sets pest control flip-flops, and transfers the system reinitialization request signal to the TUC that has been placed in the system

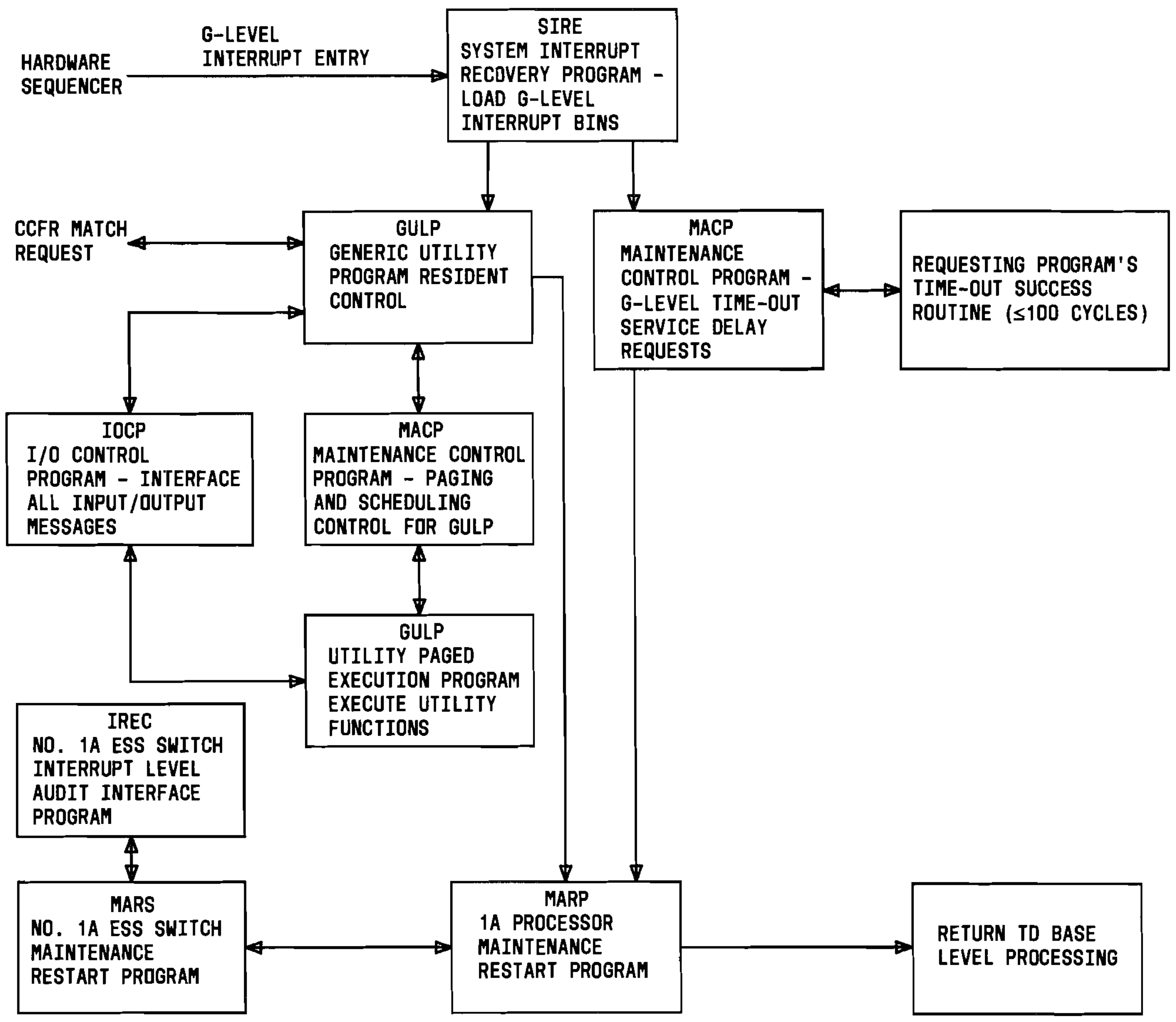

HARDWARE SEQUENCER
G-LEVEL INTERRUPT ENTRY
SIRE SYSTEM INTERRUPT RECOVERY PROGRAM - LOAD G-LEVEL INTERRUPT BINS
CCFR MATCH REQUEST
GULP GENERIC UTILITY PROGRAM RESIDENT CONTROL
MACP MAINTENANCE CONTROL PROGRAM - G-LEVEL TIME-OUT SERVICE DELAY REQUESTS
REQUESTING PROGRAM'S TIME-OUT SUCCESS ROUTINE (≤100 CYCLES)
IOCP I/O CONTROL PROGRAM - INTERFACE ALL INPUT/OUTPUT MESSAGES
MACP MAINTENANCE CONTROL PROGRAM - PAGING AND SCHEDULING CONTROL FOR GULP
GULP UTILITY PAGED EXECUTION PROGRAM EXECUTE UTILITY FUNCTIONS
IREC NO. 1A ESS SWITCH INTERRUPT LEVEL AUDIT INTERFACE PROGRAM
MARS NO. 1A ESS SWITCH MAINTENANCE RESTART PROGRAM
MARP 1A PROCESSOR MAINTENANCE RESTART PROGRAM
RETURN TO BASE LEVEL PROCESSING

Fig. 59—G-Level Processing—Program Flow and Interfaces

reinitialization mode. The system reinitialization sequencer in the TUC controls the transfer of recovery programs from the tape into core and prevents the TUC from aborting the data transfer due to errors. Operating the system reinitialization key at the TUC enables the system reinitialization sequencer in the TUC. Activation of the system reinitialization enable key at the MCC modifies the A-level interrupt sequencer and/or processor configuration sequencer to load program stores 0 from tape instead of file store as in the processor configuration recovery program. The A-level interrupt sequencer:

(a) Establishes the basic processor configuration as selected by operating keys on the MCC. If the processor configuration sequencer is used, the processor configuration state counter is activated.

(b) Sends the pump request to and initializes the data unit selector which activates the TUC system reinitialization sequencer whose system reinitialization key is depressed.

(c) Inhibits all remaining activity on the auxiliary bus.

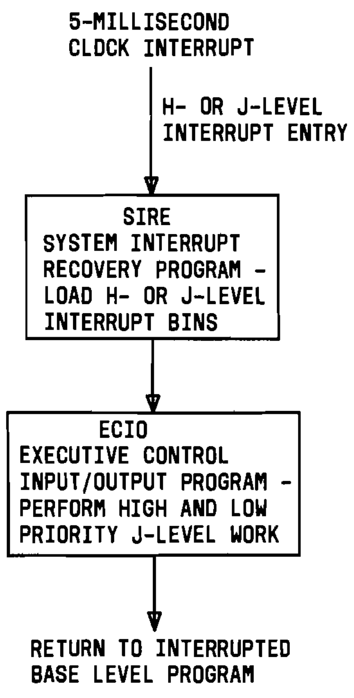

5-MILLISECOND CLOCK INTERRUPT
H- OR J-LEVEL INTERRUPT ENTRY
SIRE SYSTEM INTERRUPT RECOVERY PROGRAM - LOAD H- OR J-LEVEL INTERRUPT BINS
ECIO EXECUTIVE CONTROL INPUT/OUTPUT PROGRAM - PERFORM HIGH AND LOW PRIORITY J-LEVEL WORK
RETURN TO INTERRUPTED BASE LEVEL PROGRAM

Fig. 60—H- and J-Level Processing—Program Flow and Interfaces

(d) Inhibits the central control instruction fetch sequencer from accessing the program store.

6.36 When the A-level sequencer or the processor configuration sequencer attempts to pass control to the correct interrupt program, the central control instruction fetch sequencer is prevented from fetching instructions from core.

6.37 The system reinitialization sequencer in the TUC is activated by the pump request signal from central control. This sequencer:

(a) Clears the TUC (aborting all previous jobs) and rewinds the tape to the load point.

(b) Inhibits the termination of data transfer due to errors.

(c) Sets the character count, which indicates the end of data transfer. The load address is set to that of the A-level interrupt program.

(d) Begins the data transfer from tape to core.

6.38 A pump complete signal is sent from the TUC through the data unit selector to the central

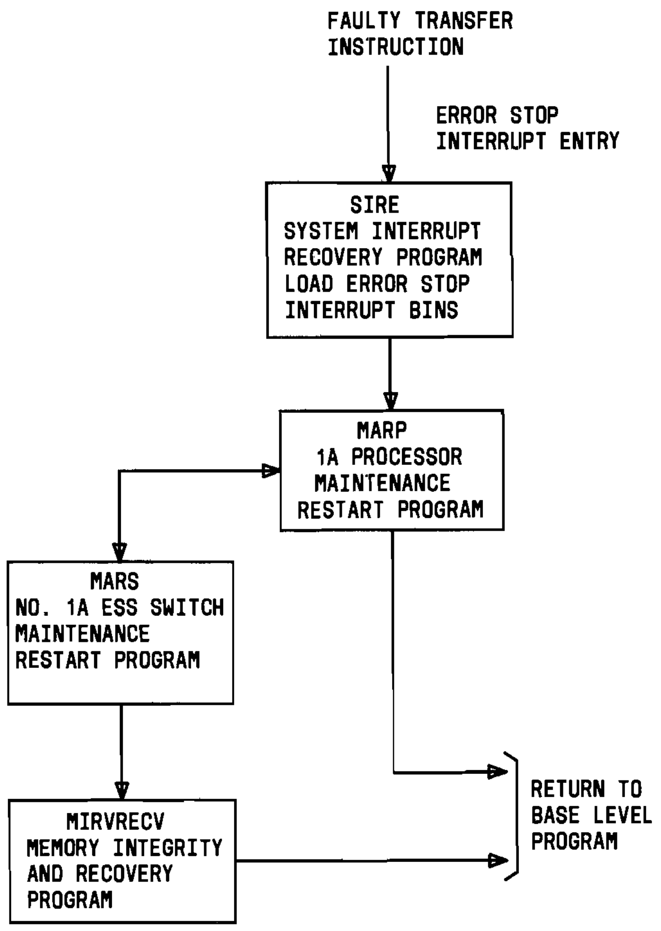

FAULTY TRANSFER INSTRUCTION
ERROR STOP INTERRUPT ENTRY
SIRE SYSTEM INTERRUPT RECOVERY PROGRAM LOAD ERROR STOP INTERRUPT BINS
MARP 1A PROCESSOR MAINTENANCE RESTART PROGRAM
MARS NO. 1A ESS SWITCH MAINTENANCE RESTART PROGRAM
MIRVRECV MEMORY INTEGRITY AND RECOVERY PROGRAM
RETURN TO BASE LEVEL PROGRAM

Fig. 61—Error Stop Processing—Program Flow and Interfaces

control to release inhibit on the instruction fetch sequencer. Program execution than begins. Figure 63 shows a functional flowchart of SR.

6.39 When errors are encountered during the transfer, a signal is sent to the MCC and the SR ERRORS lamp is lighted. Upon completion of the data transfer, as indicated by the character count in the TUC, a pump completed signal is sent to the central control. This signal resets the inhibits that block the instruction fetch sequencer in the central control.

C. Data Verification

6.40 Verification of a successful load is accomplished through a hash sum calculation. This calculation is performed on some data as it is loaded into core buffers. All other data is hash summed

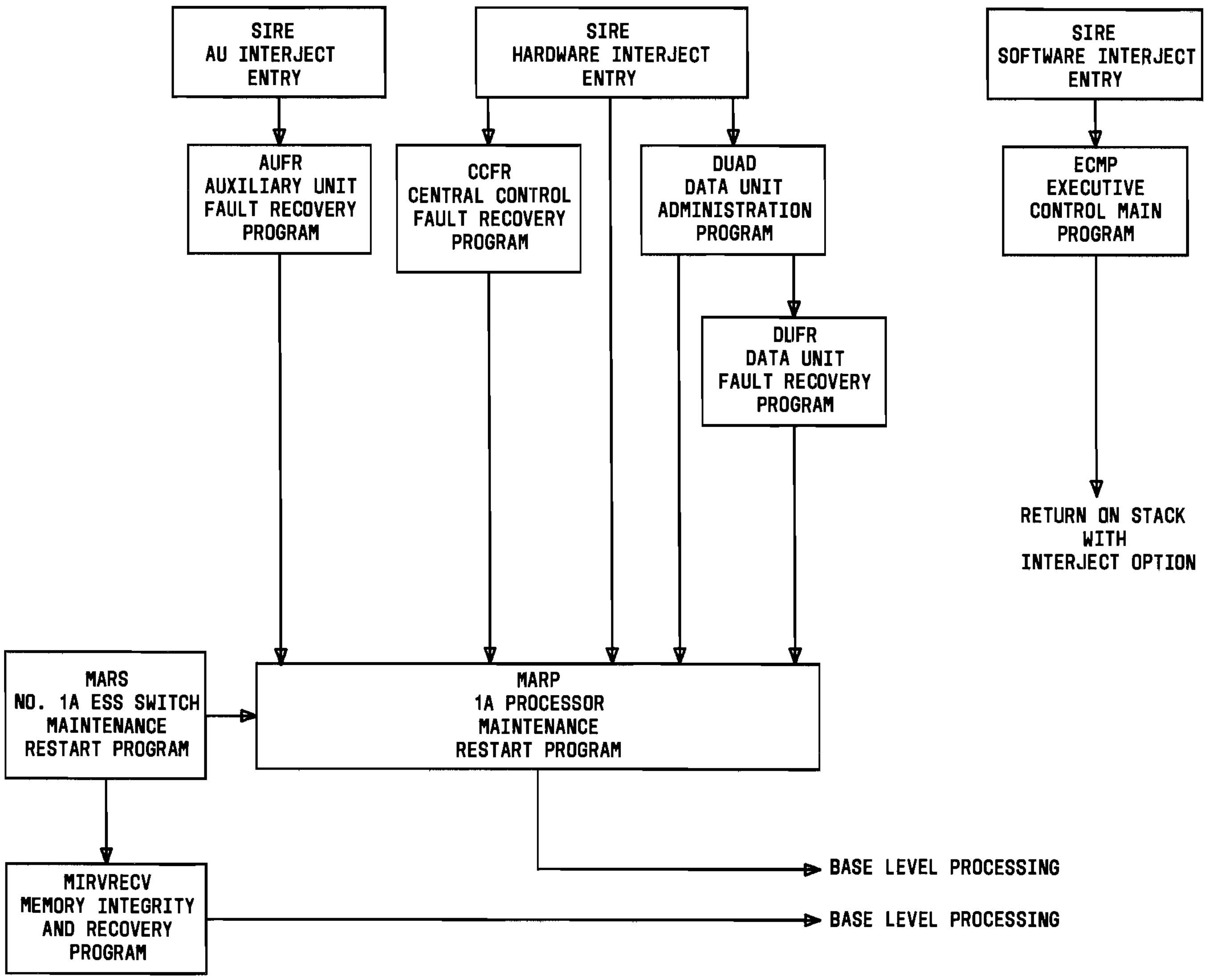

SIRE AU INTERJECT ENTRY
SIRE HARDWARE INTERJECT ENTRY
SIRE SOFTWARE INTERJECT ENTRY
AUFR AUXILIARY UNIT FAULT RECOVERY PROGRAM
CCFR CENTRAL CONTROL FAULT RECOVERY PROGRAM
DUAD DATA UNIT ADMINISTRATION PROGRAM
ECMP EXECUTIVE CONTROL MAIN PROGRAM
DUFR DATA UNIT FAULT RECOVERY PROGRAM
RETURN ON STACK WITH INTERJECT OPTION
MARS NO. 1A ESS SWITCH MAINTENANCE RESTART PROGRAM
MARP 1A PROCESSOR MAINTENANCE RESTART PROGRAM
MIRVRECV MEMORY INTEGRITY AND RECOVERY PROGRAM
BASE LEVEL PROCESSING
BASE LEVEL PROCESSING

Fig. 62—Interject Processing—Program Flow and Interfaces

while it is in the core buffers. For data which is normally resident in core, the hash sum verifies both the data and the low order bits of the core address. The file store address is verified in part by file administration programs when range-checking the file store mapping table. For data which resides in file stores only, the hash sum verifies both the data and the low order bits of the file store addresses.

6.41 Due to the wide range amounts of data between applications and offices within one application, an overall time limit is not satisfactory for the detection of an insane sequence. Instead, the core-to-file store mapping table and disk layout table images are used to verify that the data is arriving in an expected sequence. The program sanity timer is used to verify correct progression of the configuration programs and to time the read of individual records. The worst case limit in terms of program sanity timer resets is placed upon the reading of individual records. The timer is periodically reset during the read sequence. If the limit upon the number of resets is reached, the timer is allowed to time out and generate a B-level interrupt. (See paragraph 6.19.)

D. Central Control Basic Sanity Tests

6.42 The central control basic sanity tests are performed after the program store data has been

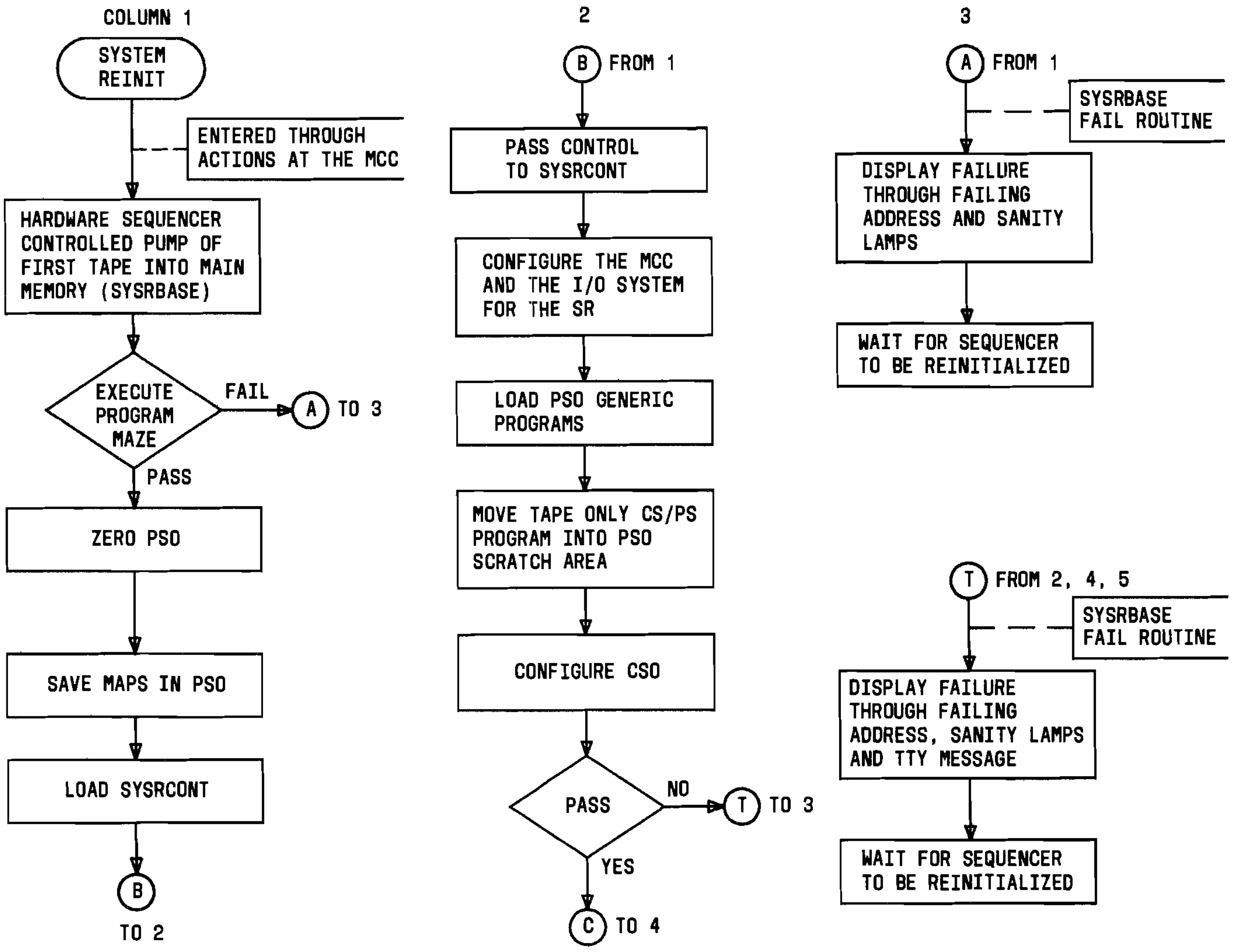

COLUMN 1
SYSTEM REINIT
ENTERED THROUGH ACTIONS AT THE MCC
HARDWARE SEQUENCER CONTROLLED PUMP OF FIRST TAPE INTO MAIN MEMORY (SYSRBASE)
EXECUTE PROGRAM MAZE
FAIL
A TO 3
PASS
ZERO PSO
SAVE MAPS IN PSO
LOAD SYSRCONT
B TO 2
2
B FROM 1
PASS CONTROL TO SYSRCONT
CONFIGURE THE MCC AND THE I/O SYSTEM FOR THE SR
LOAD PSO GENERIC PROGRAMS
MOVE TAPE ONLY CS/PS PROGRAM INTO PSO SCRATCH AREA
CONFIGURE CSO
PASS
NO
T TO 3
YES
C TO 4
3
A FROM 1
SYSRBASE FAIL ROUTINE
DISPLAY FAILURE THROUGH FAILING ADDRESS AND SANITY LAMPS
WAIT FOR SEQUENCER TO BE REINITIALIZED
T FROM 2, 4, 5
SYSRBASE FAIL ROUTINE
DISPLAY FAILURE THROUGH FAILING ADDRESS, SANITY LAMPS AND TTY MESSAGE
WAIT FOR SEQUENCER TO BE REINITIALIZED

Fig. 63—System Reinitialization—Flowchart (Sheet 1 of 7)

hashed to verify the contents as good data. The processor configuration sanity timer is used to time the basic sanity task. Auxiliary units are stopped during the central control basic sanity test. Some auxiliary unit buffer registers are changed during the basic sanity test and these will be initialized after the sanity tests have been completed. The central control basic sanity tests may be bypassed by a manual request.

E. Recovery From Catastrophic Faults

6.43 The processor configuration sequencer and the system reinitialization facilities are used along with special generic fault recovery routines to control the recovery from most catastrophic hardware faults. Catastrophic hardware faults are faults that prevent the fault recovery programs from configuring the processor using the backup units. An example of such a situation is a fault occurring in both the unit and its backup. The recovery process depends on the ability of one central control to load program store 0 from units on the auxiliary bus. When faults occur that completely inhibit the central control from its loading capability, recovery is achieved through an office installation test using the portable recovery test set (PORTs).

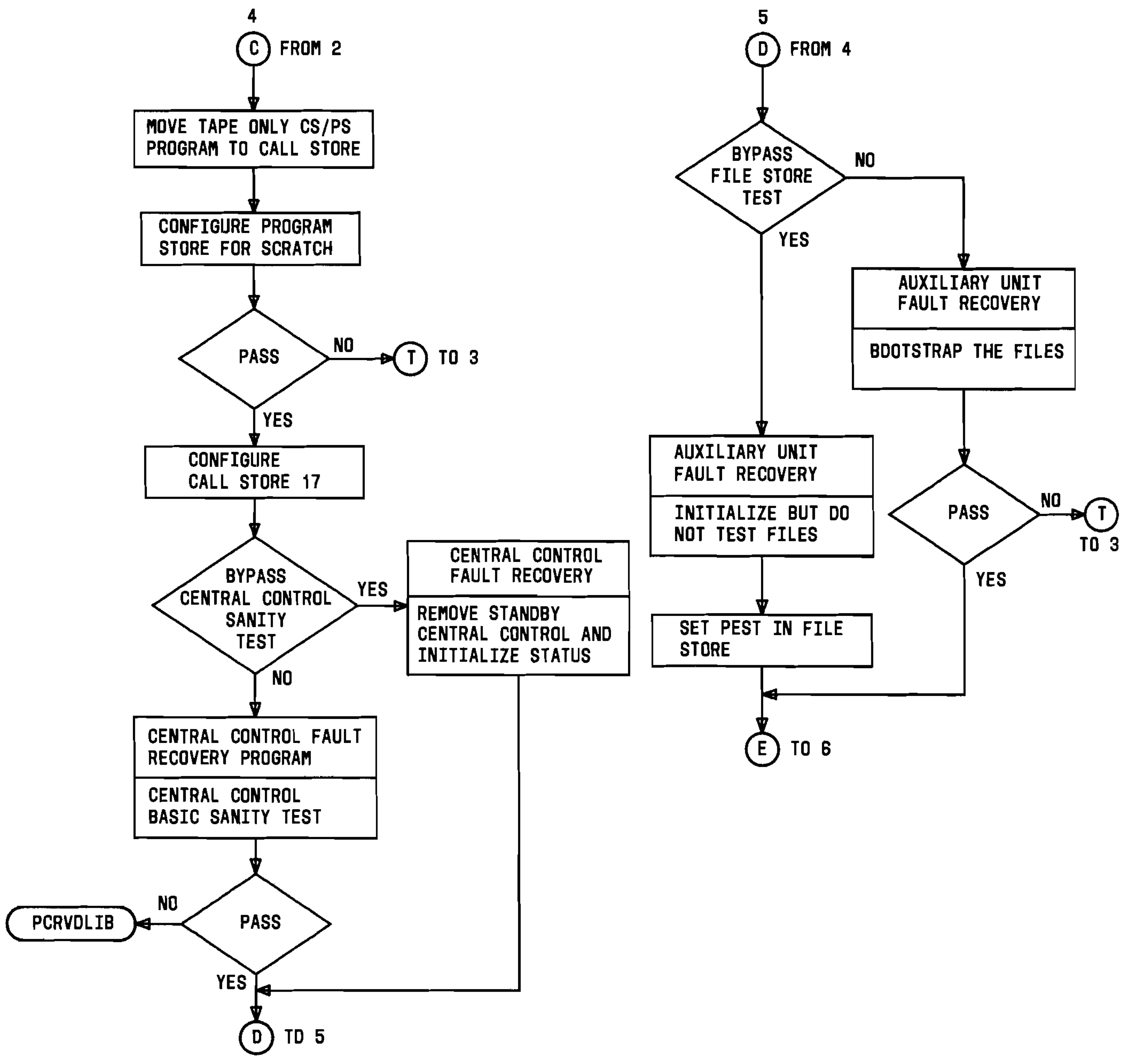

4
C FROM 2
MOVE TAPE ONLY CS/PS PROGRAM TO CALL STORE
CONFIGURE PROGRAM STORE FOR SCRATCH
PASS
NO
T TO 3
YES
CONFIGURE CALL STORE 17
BYPASS CENTRAL CONTROL SANITY TEST
YES
CENTRAL CONTROL FAULT RECOVERY
REMOVE STANDBY CENTRAL CONTROL AND INITIALIZE STATUS
NO
CENTRAL CONTROL FAULT RECOVERY PROGRAM
CENTRAL CONTROL BASIC SANITY TEST
PASS
NO
PCRVDLIB
YES
D TO 5
5
D FROM 4
BYPASS FILE STORE TEST
NO
AUXILIARY UNIT FAULT RECOVERY
BOOTSTRAP THE FILES
YES
AUXILIARY UNIT FAULT RECOVERY
INITIALIZE BUT DO NOT TEST FILES
PASS
NO
T TO 3
YES
SET PEST IN FILE STORE
E TO 6

Fig. 63—System Reinitialization—Flowchart (Sheet 2 of 7)

F. Process in the Presence of Failures

6.44 The process in the presence of failures procedure is designed for recovery from faults in both the unit and its backup. When the central control is at fault, the central control basic sanity test is bypassed. In this mode the central control may be able to configure a system with operational (but not maintenance) capabilities. The program updates the central control status but performs no sanity tests.

6.45 The maintenance tests on call stores and program stores are bypassed to increase the chances of configuring a working subsystem. The fault recovery programs for call store and program store are called to configure these stores without

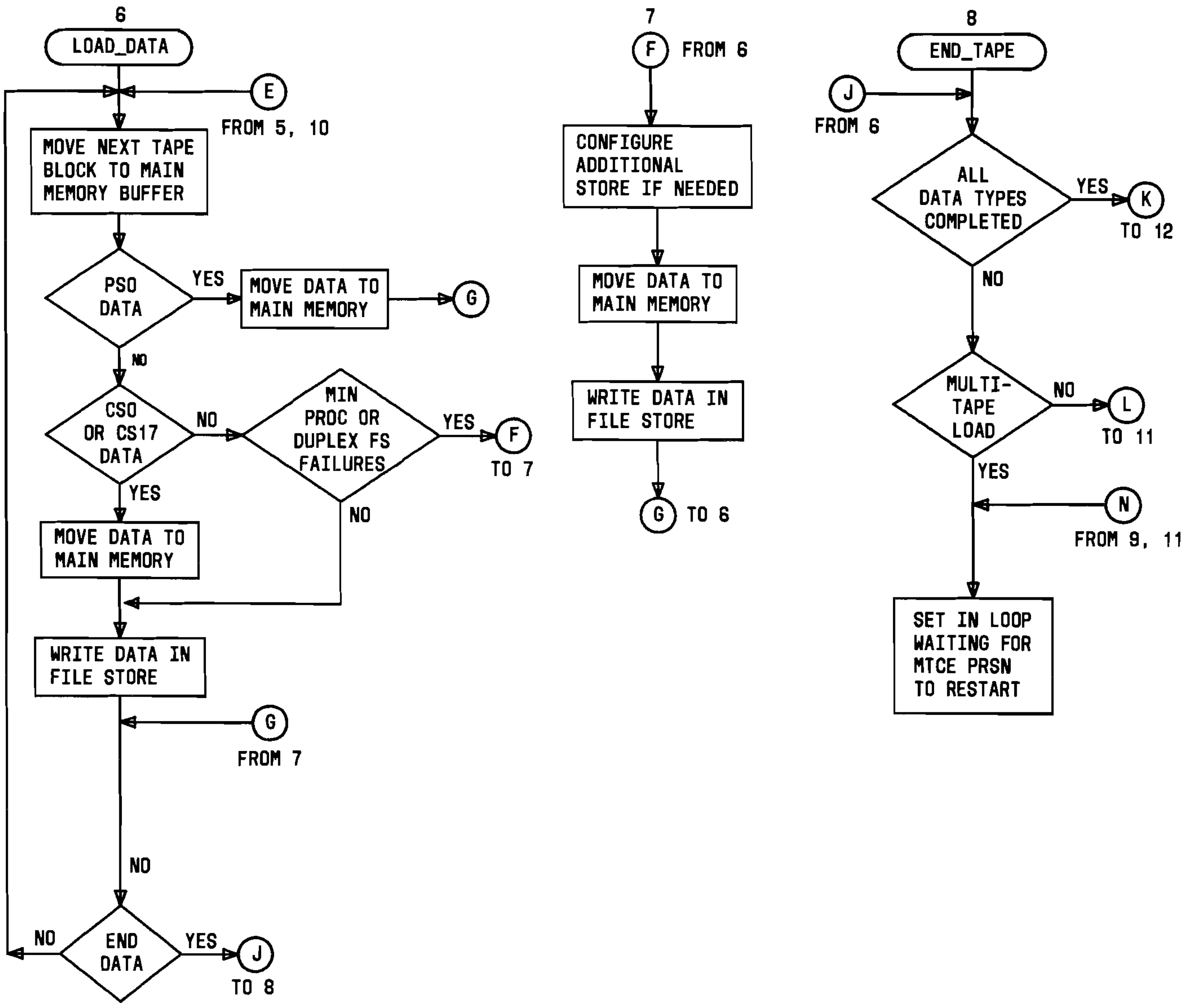


Fig. 63—System Reinitialization—Flowchart (Sheet 3 of 7)

maintenance capabilities. The operational tests are performed on the stores, and a system configured using this procedure is able to process calls and to diagnose units. However, one is cautioned that configuring a call store or program store which fails the maintenance test could result in that store interfering with the diagnosis of other units in the same community.

6.46 The Ignore Data Transfer Errors mode is used with the system reinitalization sequencers. This mode is also selected by the maintenance personnel and then remains in effect throughout the recovery. This mode is used when auxiliary unit faults exist that do not affect the validity of the data transferred. When this mode is selected, the recovery program initializes the auxiliary units but does not perform any tests. The error-checking circuits in the auxiliary units are also pested. In order to verify that the file stores thus configured are acceptable, the system reinitialization reads generic and translation data written on the file, and compares it to the original data from the tape.

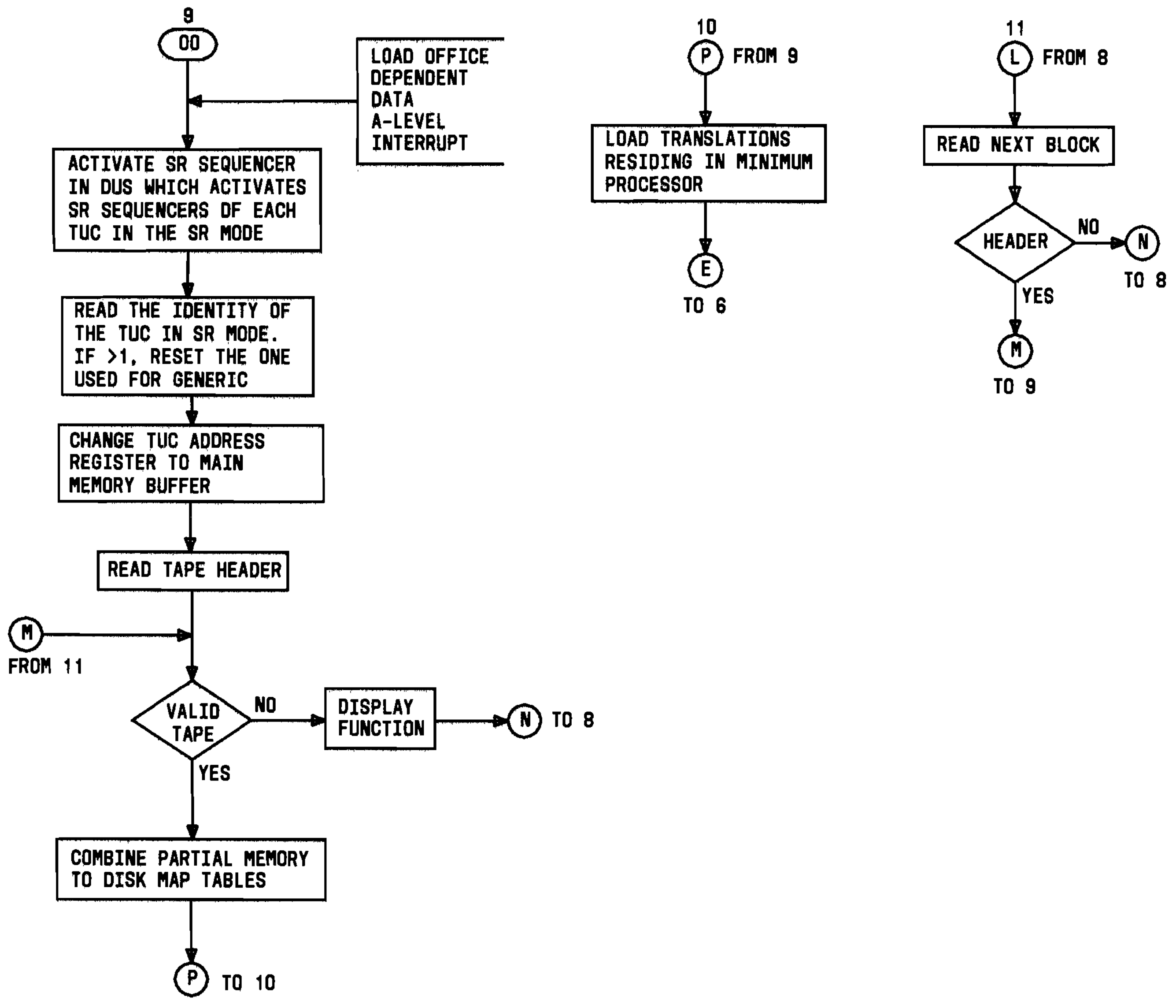

9
00
LOAD OFFICE DEPENDENT DATA A-LEVEL INTERRUPT
ACTIVATE SR SEQUENCER IN DUS WHICH ACTIVATES SR SEQUENCERS OF EACH TUC IN THE SR MODE
READ THE IDENTITY OF THE TUC IN SR MODE. IF >1, RESET THE ONE USED FOR GENERIC
CHANGE TUC ADDRESS REGISTER TO MAIN MEMORY BUFFER
READ TAPE HEADER
M FROM 11
VALID TAPE
NO
DISPLAY FUNCTION
N TO 8
YES
COMBINE PARTIAL MEMORY TO DISK MAP TABLES
P TO 10
10
P FROM 9
LOAD TRANSLATIONS RESIDING IN MINIMUM PROCESSOR
E TO 6
11
L FROM 8
READ NEXT BLOCK
HEADER
NO
N TO 8
YES
M TO 9

Fig. 63—System Reinitialization—Flowchart (Sheet 4 of 7)

6.47 The duplex file store or APS failure mode is used when a parity file store, API, and 3B is suspected before performing system reinitialization. When attempting system reinitialization, key 23 is pressed and a message appears with a request to power down all non-operational stores. The system attempts to load system generic information into file store or APS, call store and program store. If file store or APS is faulty, then the system places file store or APS off-line and proceeds with loading generic into call store and program store.

HARDWARE RECOVERY

A. General

6.48 When the No. 1A ESS switch develops a problem which results in a high-level maintenance interrupt (ie, A-level or B-level interrupt), the hardware recovery programs are responsible for configuring a working system of hardware. These programs attempt a systematic recovery of the call processing capabilities by configuring the various parts of the No. 1A ESS switch (eg, central control, peripheral buses, program store buses, etc) to find a valid configuration of the system. When it is necessary, these programs include the filling of main memory stores using the backup data maintained in the file store.

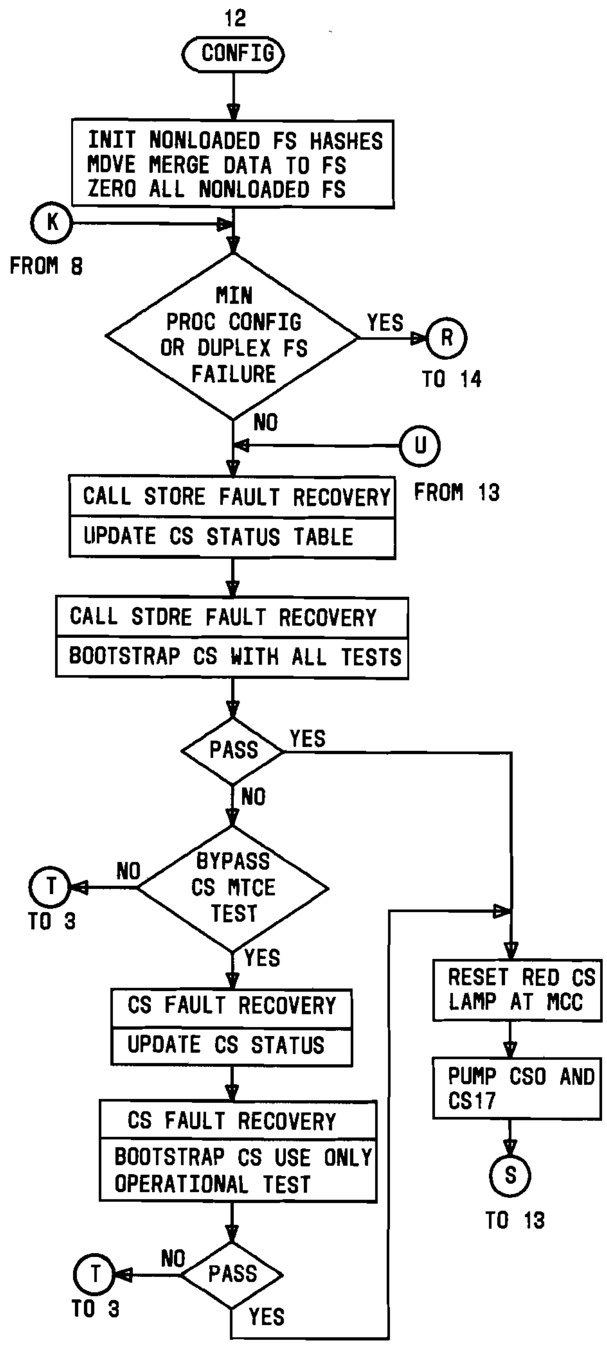


Fig. 63—System Reinitialization—Flowchart (Sheet 5 of 7)

6.49 Most of the activities involved with hardware recovery operate on the 1A processor, not the No. 1A ESS switch in general. The processor configuration programs are designed to recover the system processing ability given faults which the fault recovery or software sanity programs are unable to process. The processor configuration programs are normally requested automatically by the processor configuration sequencer. However, they can be entered via a manual request from the MCC or they can be requested by other fault recovery programs and phase programs.

6.50 Processor configuration programs recover from the active processor faults which prevent successful execution of fault recovery programs or from software faults which inhibit the running of the monitor programs. The processor configuration recovery includes filling of main memory stores using the file store as backup. This fill is requested when tests show that the main memory data is mutilated or when repeated failures in the configuration routines indicate mutilated generic or office data, or when the sequencer reaches a pump state.

6.51 The processor configuration circuit is housed in the 1A processor central control. It provides a means to recover the system from the following two classes of faults:

(a) Hardware faults, the first class, prevent maintenance interrupt levels C through F from recovering specific processor subsystems. The loss of central control operational clock and other faults that cause both central controls to be active or standby are examples of the first class of faults.

(b) Software faults, the second class, prevent the scheduling of the software monitor programs. The monitoring function is inhibited by such faults, thereby blocking programmed recovery. The mutilation of the software monitor program in the program store is an example of such a fault.

6.52 The processor configuration circuit establishes a basic processor configuration consisting of a central control, basic program store block, and program store bus. As the processor configuration advances to more drastic recovery procedures, the processor configuration circuit also selects a file

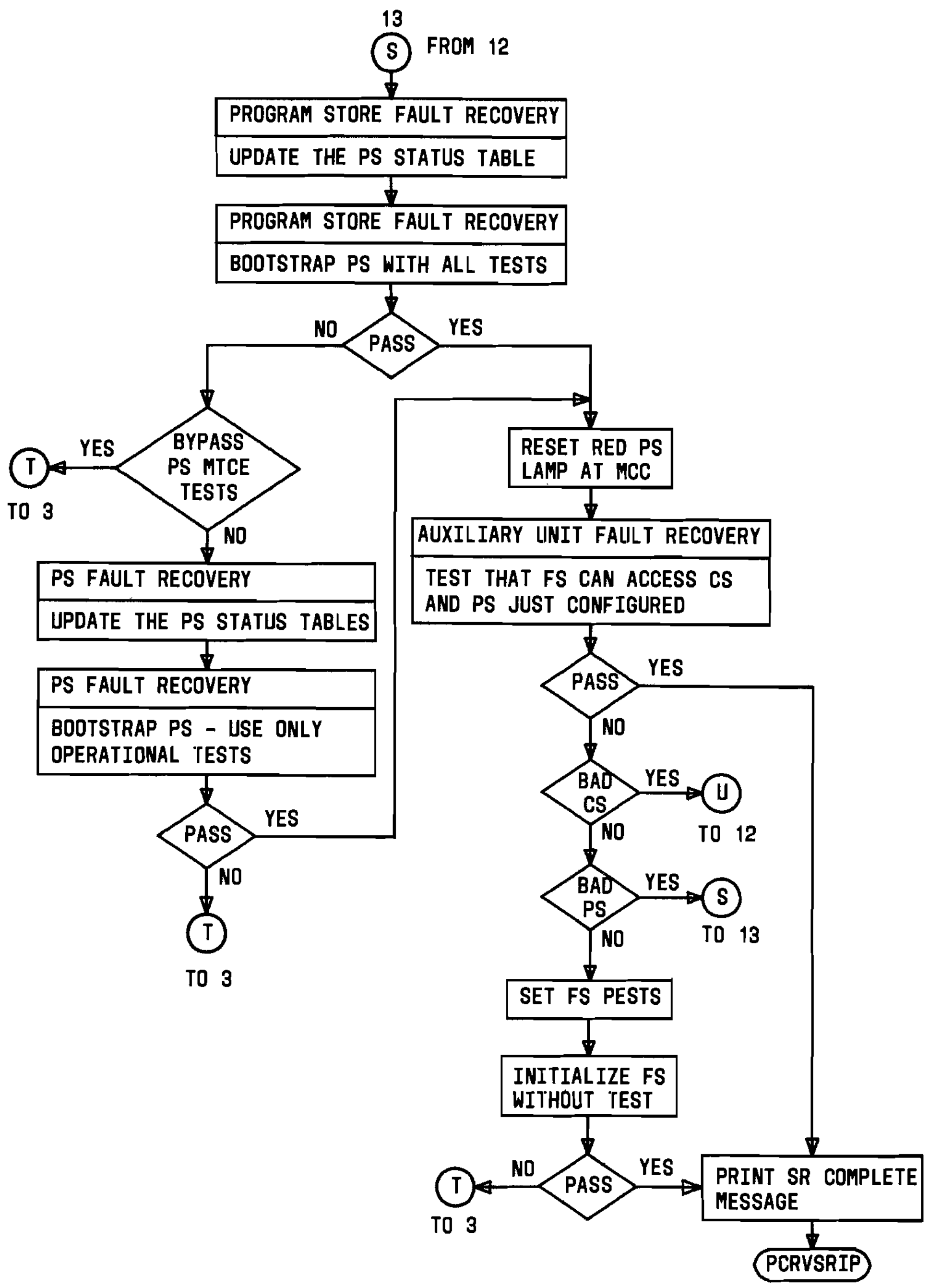

13
S
FROM 12
PROGRAM STORE FAULT RECOVERY
UPDATE THE PS STATUS TABLE
PROGRAM STORE FAULT RECOVERY
BOOTSTRAP PS WITH ALL TESTS
NO
PASS
YES
YES
BYPASS PS MTCE TESTS
T
TO 3
NO
PS FAULT RECOVERY
UPDATE THE PS STATUS TABLES
PS FAULT RECOVERY
BOOTSTRAP PS - USE ONLY OPERATIONAL TESTS
PASS
YES
NO
T
TO 3
RESET RED PS LAMP AT MCC
AUXILIARY UNIT FAULT RECOVERY
TEST THAT FS CAN ACCESS CS AND PS JUST CONFIGURED
PASS
YES
NO
BAD CS
YES
U
TO 12
NO
BAD PS
YES
S
TO 13
NO
SET FS PESTS
INITIALIZE FS WITHOUT TEST
T
TO 3
NO
PASS
YES
PRINT SR COMPLETE MESSAGE
PCRVSRIP

Fig. 63—System Reinitialization—Flowchart (Sheet 6 of 7)

store and file store bus with which to replace main memory data. It also forces an automatic pump of a special disk only program.

6.53 The particular configuration that is attempted upon each activation of the processor configuration circuit is a function of the processor configuration state counter. This counter is incremented during each activation of the processor configuration circuit, thus cycling the processor configuration circuit through all possible configuration states. The counter is reset via program access after a working processor has been found. The central controls are switched in several of the processor configuration states during the recovery process.

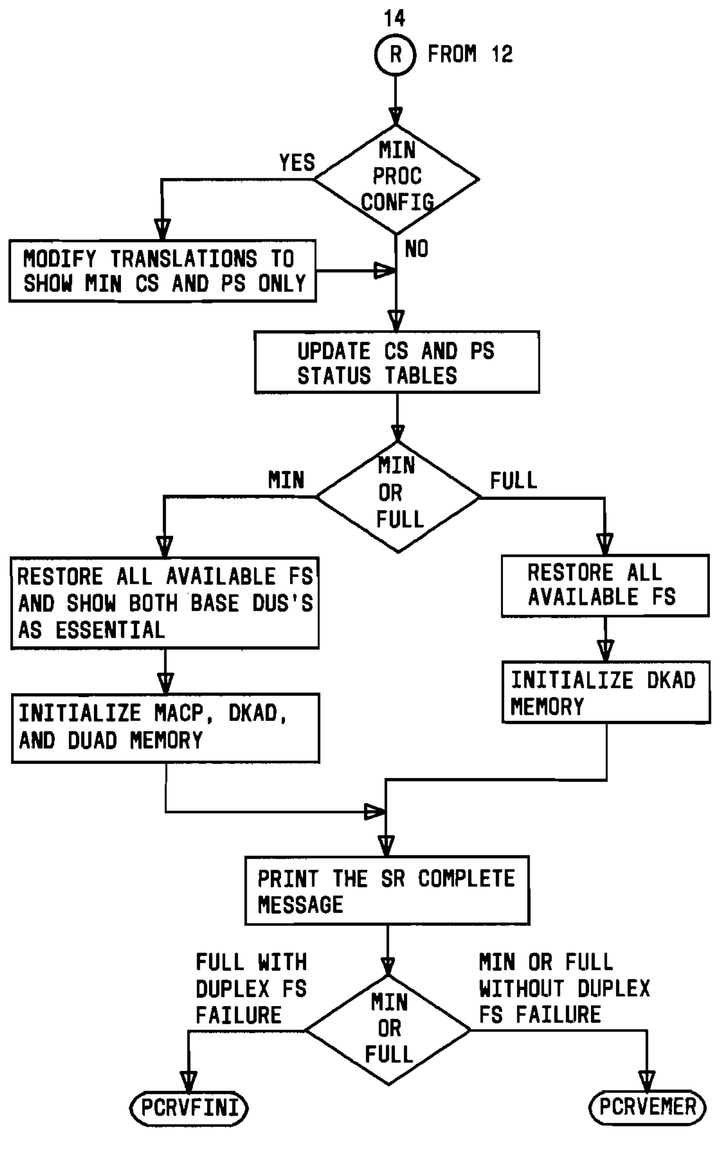

14
R
FROM 12
MIN PROC CONFIG
YES
NO
MODIFY TRANSLATIONS TO SHOW MIN CS AND PS ONLY
UPDATE CS AND PS STATUS TABLES
MIN OR FULL
MIN
FULL
RESTORE ALL AVAILABLE FS AND SHOW BOTH BASE DUS'S AS ESSENTIAL
RESTORE ALL AVAILABLE FS
INITIALIZE MACP, DKAD, AND DUAD MEMORY
INITIALIZE DKAD MEMORY
PRINT THE SR COMPLETE MESSAGE
MIN OR FULL
FULL WITH DUPLEX FS FAILURE
MIN OR FULL WITHOUT DUPLEX FS FAILURE
PCRVFINI
PCRVEMER

Fig. 63—System Reinitialization—Flowchart (Sheet 7 of 7)

6.54 The processor configuration state counter provides 48 unique states. Each state combined with the sequence of analog clock pulses used to gate processor configuration functions defines a particular processor configuration activity.

6.55 A program sequence is initiated to recover all processor subsystems and to verify and correct transient data. This system recovery is attempted in two levels:

(a) The first level utilizes the first 16 states of the processor configuration circuit. The speed of recovery is the prime objective in the first level and is achieved by using the existing program store data. The second level of recovery uses the remaining 32 processor configuration states.

(b) The second level is concerned with forcing recovery and not speed. The probability of recovery is increased by using the file store data for the basic recovery programs. Failure to recover the system in level 1 results in an automatic escalation to level 2. System reinitialization and software initialization programs become involved when the processor configuration state counter reaches the second level or pump state.

B. Processor Configuration Recovery (Level 1)

6.56 Processor configuration program entries are the result of A-level or B-level interrupts. The first level of processor configuration recovery is concerned with the speed of recovery. The main memory and file stores are assumed to be good.

6.57 The processor configuration circuit establishes a basic processor for the recovery procedure consisting of a central control, a program store block, and a program store bus. These basic units are forced on-line and the processor sequencer begins a systematic effort to confirm the viability of each basic unit. Level 1 is comprised of 16 states and if the processor configuration is not recovered, escalation to level 2 is automatic. Figure 64 shows the general flow of the level 1 recovery sequence.

C. Processor Configuration Recovery (Level 2)

6.58 The second level of processor configuration recovery is reached when the first level fails to recover the systems or by a manual pump request (A-level interrupt) at the MCC. The level 2 recovery assumes that all software is mutilated and operates only on data that has been pumped from file store or data that has been hash summed. The hardware sequencer forces on-line a central control, a program store, a program store bus, and a file store with which to operate. All other program stores have been forced into the maintenance mode and all other auxiliary units are denied access to an auxiliary unit bus. See Fig. 65 for recovery sequence.

D. Hardware Initialization

6.59 After successful completion of the processor configuration recovery action, selected hardware must be initialized. Initializing, in this respect, includes a limited amount of testing of the selected hardware and updating the software image of the hardware. The hardware that is initialized includes:

- Coded enable peripheral unit bus
- Input/output unit selectors (IOUSs) and input/output unit controllers (IOUCs)
- Central pulse distributor enable address bus (CPDB)
- Central pulse distributors (CPDs)
- Peripheral unit address bus (PUAB).

See Fig. 66 for a flowchart covering hardware initialization.

6.60 Hardware initialization is always followed by validity procedures to ensure good software. Although main memory and file data may have been renewed, there is a chance that some faults may still exist. Since all base level processing has stopped during these high level interrupts (A or B-level), it is important that processing resume with a minimal chance of further interrupts. Following this philosophy, the software is continually verified following any events within the processor community. These events include both levels of processor configuration and system reinitialization.

SOFTWARE INITIALIZATION

A. General

6.61 There are two types of software data: nontransient and transit. Nontransient data is stored in main memory and backed up in file store. It includes the generic program, parameters, and translations. Obviously, this data must be good or the system will not operate properly. Transient data is just as important to the system sanity. Software initialization programs are provided to check the validity of transient data.

6.62 Transient data consists of variable call processing data, ie, real-time activity data. Since this data is constantly changing, it is only backed up by a duplicate copy available in call store. Transient data is also routinely checked via audits to ensure the reliability of the data. These audits cannot use hash sum calculations due to the nature of the data. Therefore, the audits detect errors by redundancy checks and pattern matching. Serious hardware or software faults can cause mutilation and if recovery time is lengthy, invalid transient data may result. If the transient data is not quickly regenerated, the system will become very unstable. When this occurs, it is necessary to regenerate large portions of transient data. This regeneration is called software initialization and is performed in progressive stages called phases. During execution of some phases, nontransient data is checked via hash sum calculations and corrected, if necessary. However, phase ac-

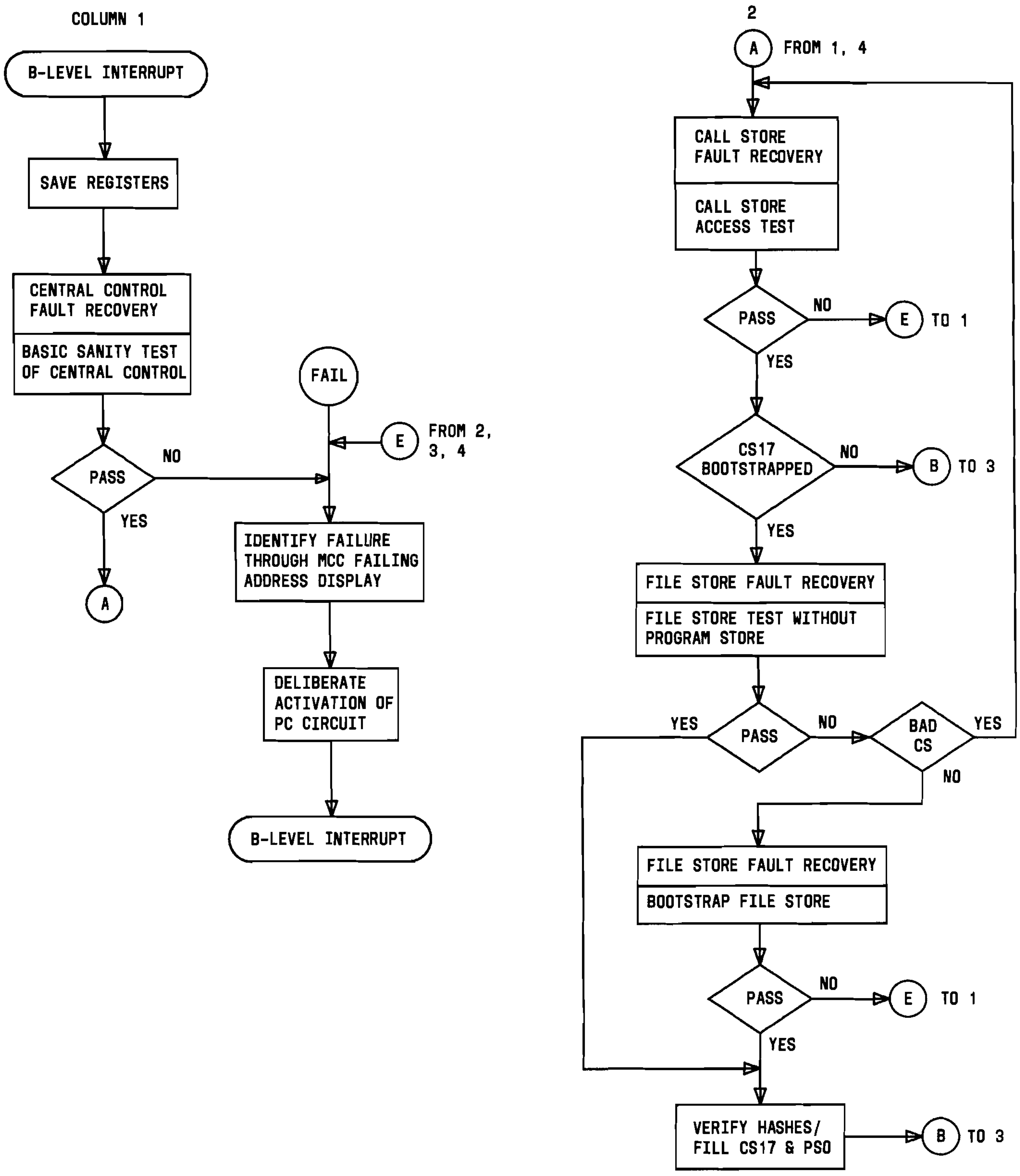

COLUMN 1
B-LEVEL INTERRUPT
SAVE REGISTERS
CENTRAL CONTROL FAULT RECOVERY
BASIC SANITY TEST OF CENTRAL CONTROL
PASS
NO
YES
A
FAIL
E
FROM 2, 3, 4
IDENTIFY FAILURE THROUGH MCC FAILING ADDRESS DISPLAY
DELIBERATE ACTIVATION OF PC CIRCUIT
B-LEVEL INTERRUPT
2
A
FROM 1, 4
CALL STORE FAULT RECOVERY
CALL STORE ACCESS TEST
PASS
NO
E
TO 1
YES
CS17 BOOTSTRAPPED
NO
B
TO 3
YES
FILE STORE FAULT RECOVERY
FILE STORE TEST WITHOUT PROGRAM STORE
YES
PASS
NO
BAD CS
YES
NO
FILE STORE FAULT RECOVERY
BOOTSTRAP FILE STORE
PASS
NO
E
TO 1
YES
VERIFY HASHES/ FILL CS17 & PS0
B
TO 3

Fig. 64—Processor Configuration Recovery Level 1—Flowchart (Sheet 1 of 3)

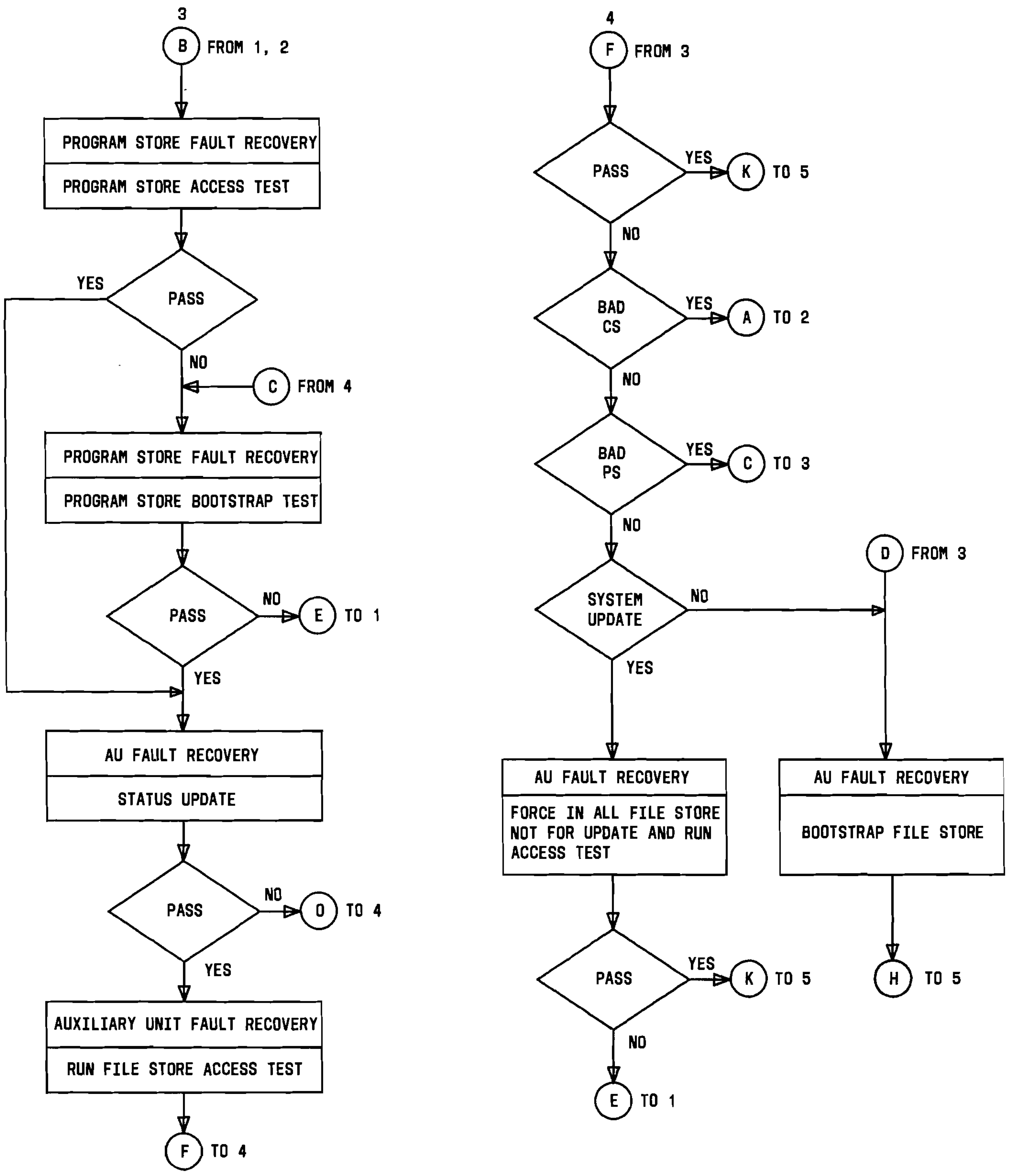


Fig. 64—Processor Configuration Recovery Level 1—Flowchart (Sheet 2 of 3)

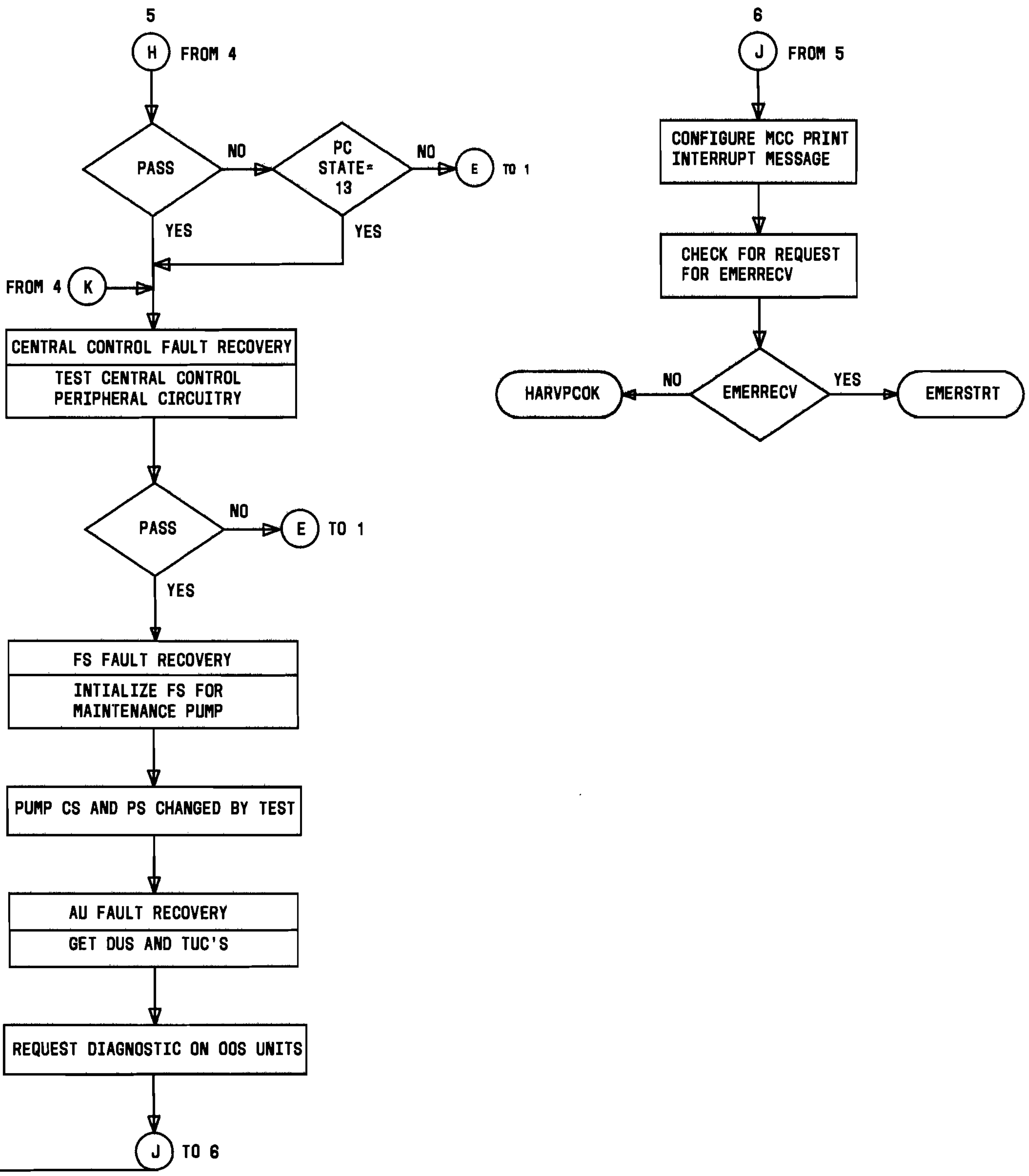

5
H
FROM 4
PASS
NO
PC STATE= 13
NO
E
TO 1
YES
YES
FROM 4
K
CENTRAL CONTROL FAULT RECOVERY
TEST CENTRAL CONTROL PERIPHERAL CIRCUITRY
PASS
NO
E
TO 1
YES
FS FAULT RECOVERY
INTIALIZE FS FOR MAINTENANCE PUMP
PUMP CS AND PS CHANGED BY TEST
AU FAULT RECOVERY
GET DUS AND TUC'S
REQUEST DIAGNOSTIC ON OOS UNITS
J
TO 6
6
J
FROM 5
CONFIGURE MCC PRINT INTERRUPT MESSAGE
CHECK FOR REQUEST FOR EMERRECV
HARVPCOK
NO
EMERRECV
YES
EMERSTRT

Fig. 64—Processor Configuration Recovery Level 1—Flowchart (Sheet 3 of 3)

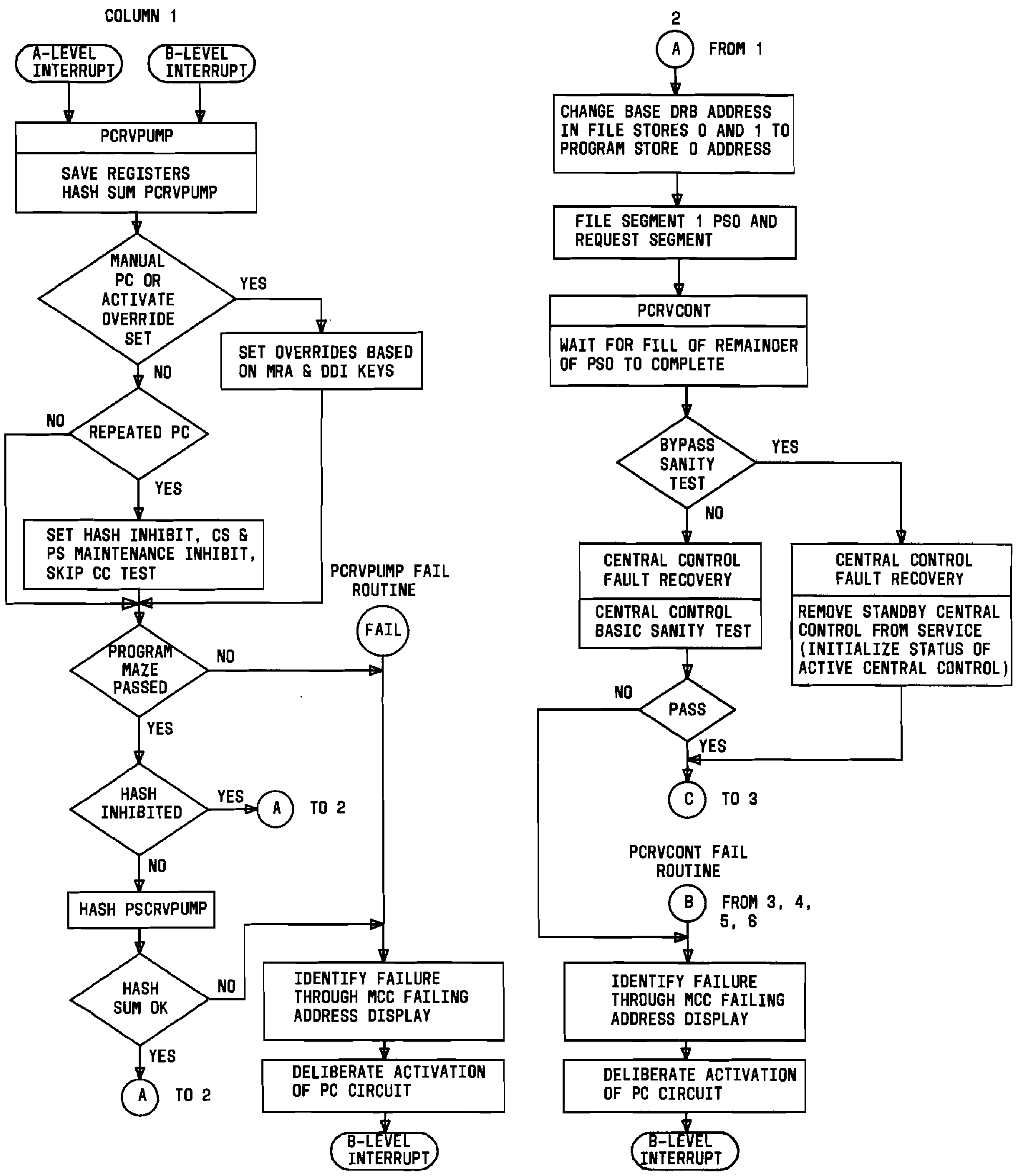

COLUMN 1
A-LEVEL INTERRUPT
B-LEVEL INTERRUPT
PCRVPUMP
SAVE REGISTERS
HASH SUM PCRVPUMP
MANUAL PC OR ACTIVATE OVERRIDE SET
YES
SET OVERRIDES BASED ON MRA & DDI KEYS
NO
NO
REPEATED PC
YES
SET HASH INHIBIT, CS & PS MAINTENANCE INHIBIT, SKIP CC TEST
PCRVPUMP FAIL ROUTINE
FAIL
PROGRAM MAZE PASSED
NO
YES
HASH INHIBITED
YES
A
TO 2
NO
HASH PSCRVPUMP
HASH SUM OK
NO
YES
A
TO 2
IDENTIFY FAILURE THROUGH MCC FAILING ADDRESS DISPLAY
DELIBERATE ACTIVATION OF PC CIRCUIT
B-LEVEL INTERRUPT
2
A
FROM 1
CHANGE BASE DRB ADDRESS IN FILE STORES 0 AND 1 TO PROGRAM STORE 0 ADDRESS
FILE SEGMENT 1 PSO AND REQUEST SEGMENT
PCRVCONT
WAIT FOR FILL OF REMAINDER OF PSO TO COMPLETE
BYPASS SANITY TEST
YES
NO
CENTRAL CONTROL FAULT RECOVERY
CENTRAL CONTROL BASIC SANITY TEST
CENTRAL CONTROL FAULT RECOVERY
REMOVE STANDBY CENTRAL CONTROL FROM SERVICE (INITIALIZE STATUS OF ACTIVE CENTRAL CONTROL)
NO
PASS
YES
C
TO 3
PCRVCONT FAIL ROUTINE
B
FROM 3, 4, 5, 6
IDENTIFY FAILURE THROUGH MCC FAILING ADDRESS DISPLAY
DELIBERATE ACTIVATION OF PC CIRCUIT
B-LEVEL INTERRUPT

Fig. 65—Processor Configuration Recovery Level 2—Flowchart (Sheet 1 of 4)

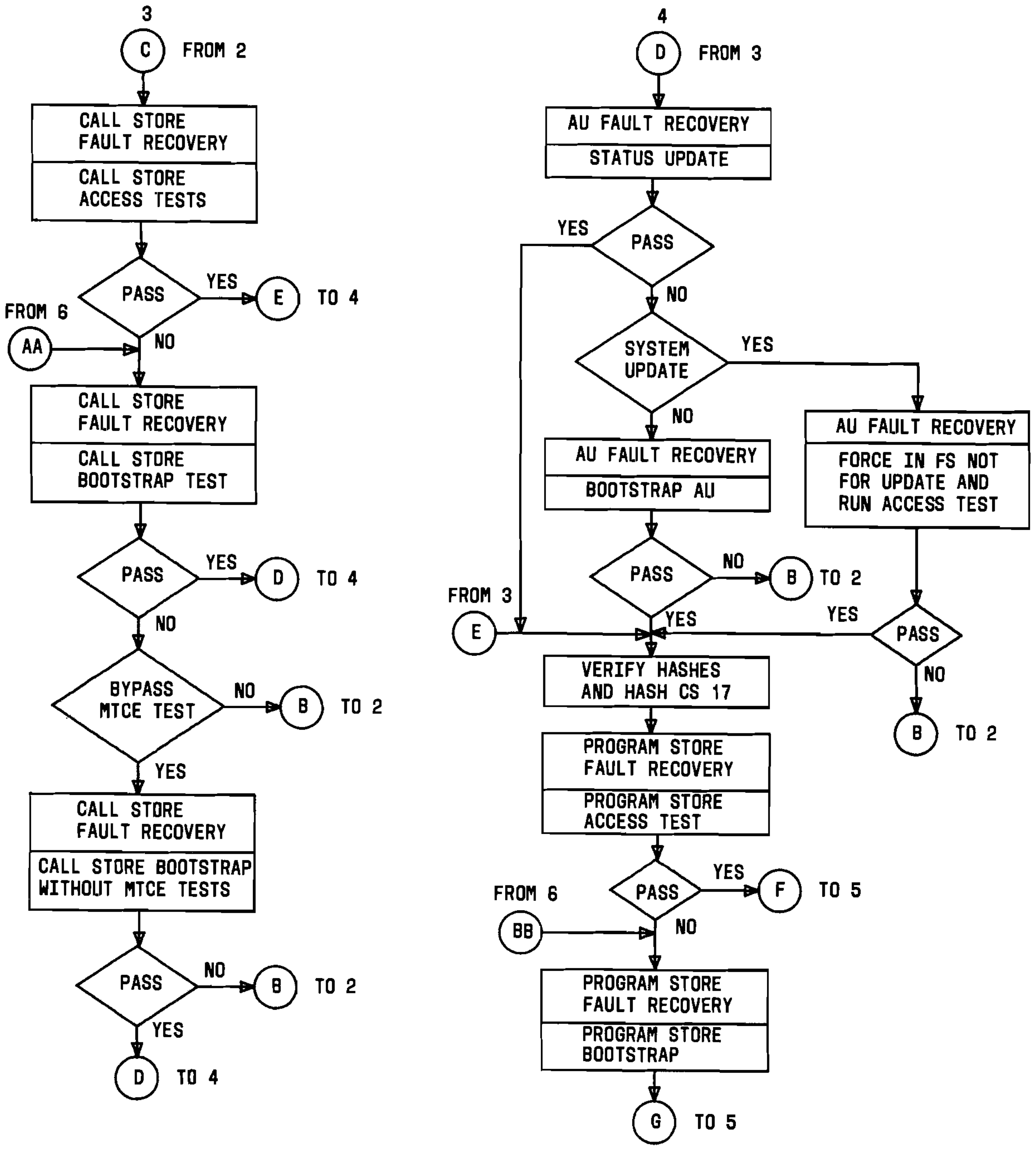


Fig. 65—Processor Configuration Recovery Level 2—Flowchart (Sheet 2 of 4)

tivity is primarily directed at regenerating transient data.

6.63 In the No. 1A switch, there are four phases of initialization. The phases progress numerically from phase 1 to phase 6, excluding phases 2 and 3. Phases 1, 4, 5, and 6 consist of audits that are stitched together; ie, the priority of the audits is raised, and they are executed consecutively. Each phase is more comprehensive than the previous one and has a more drastic effect on the system. For example, phase 1 initializes a relatively small portion of transient call store data. During a phase 6 all calls in progress are knocked down. Once a phase is trig-

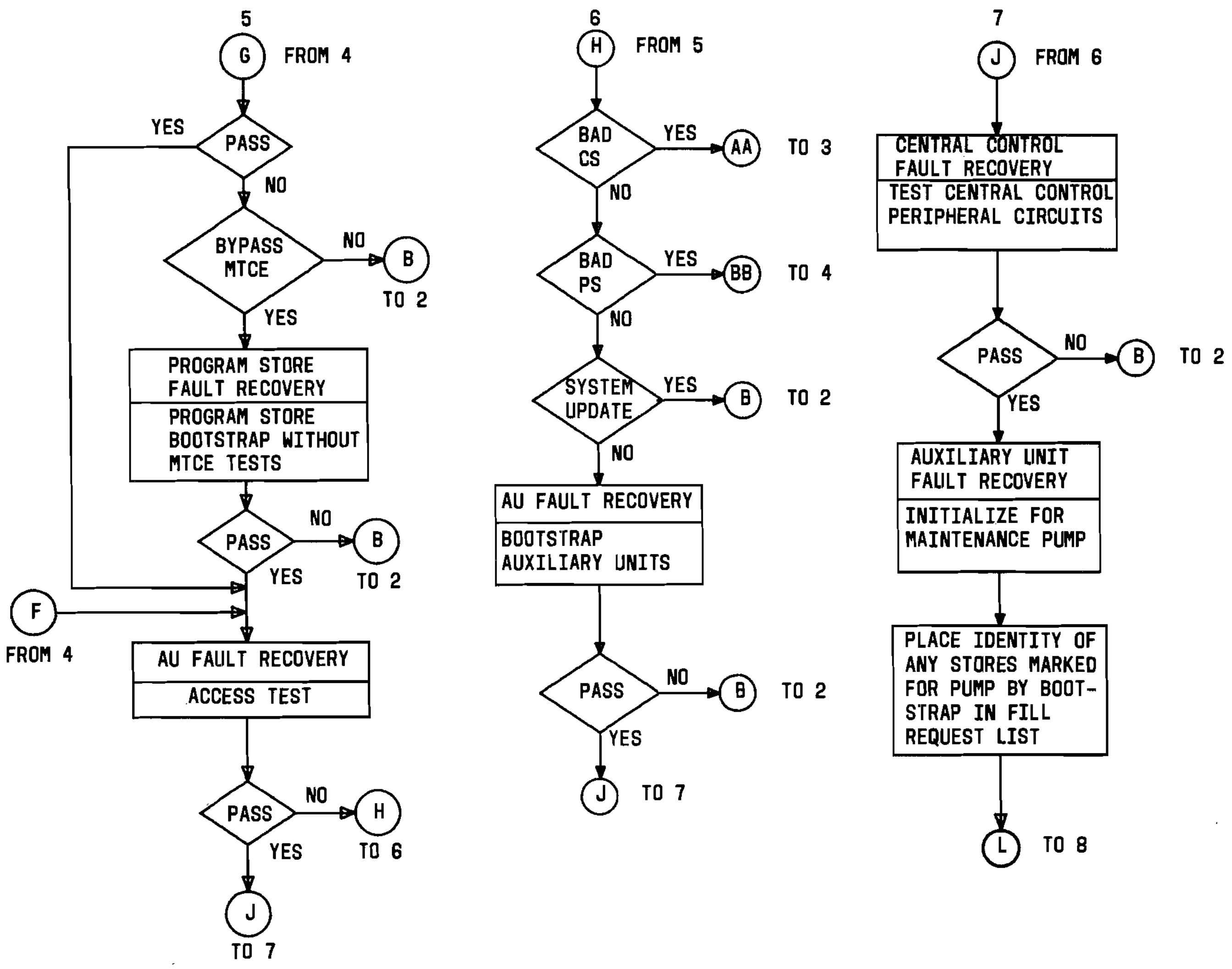

5
G
FROM 4
YES
PASS
NO
BYPASS MTCE
NO
B
TO 2
YES
PROGRAM STORE FAULT RECOVERY
PROGRAM STORE BOOTSTRAP WITHOUT MTCE TESTS
PASS
NO
B
YES
TO 2
F
FROM 4
AU FAULT RECOVERY
ACCESS TEST
PASS
NO
H
YES
TO 6
J
TO 7
6
H
FROM 5
BAD CS
YES
AA
TO 3
NO
BAD PS
YES
BB
TO 4
NO
SYSTEM UPDATE
YES
B
TO 2
NO
AU FAULT RECOVERY
BOOTSTRAP AUXILIARY UNITS
PASS
NO
B
TO 2
YES
J
TO 7
7
J
FROM 6
CENTRAL CONTROL FAULT RECOVERY
TEST CENTRAL CONTROL PERIPHERAL CIRCUITS
PASS
NO
B
TO 2
YES
AUXILIARY UNIT FAULT RECOVERY
INITIALIZE FOR MAINTENANCE PUMP
PLACE IDENTITY OF ANY STORES MARKED FOR PUMP BY BOOTSTRAP IN FILL REQUEST LIST
L
TO 8

Fig. 65—Processor Configuration Recovery Level 2—Flowchart (Sheet 3 of 4)

gered, it must run to completion. Phases are automatically or manually initiated in response to sanity affecting faults. A phase 1 can be triggered manually or automatically and, if the system is unable to resume call processing, the system automatically advances to a phase 4. If the system encounters difficulties while processing a phase 4, the system automatically advances to a phase 5. If a phase 5 fails to recover the system, it will loop in a phase 5 because the system cannot automatically advance to a phase 6. A phase 6 can only be initiated manually.

B. Phase Triggers

6.64 There are ten sources that trigger phases. There are three phase triggers generated internally by the phase control program. These triggers deal with base level malfunction, unanswered interject, and audit time-outs during a phase. The other seven phase triggers are generated externally to the phase control program. All ten phase triggers can be classified into four categories. Each one has several individual trigger code numbers which identify the source of phase activity. The four categories are:

(a) ***Manual Request:*** These triggers are related to manual actions by the maintenance personnel (eg, a manual phase 6 requested at the MCC).

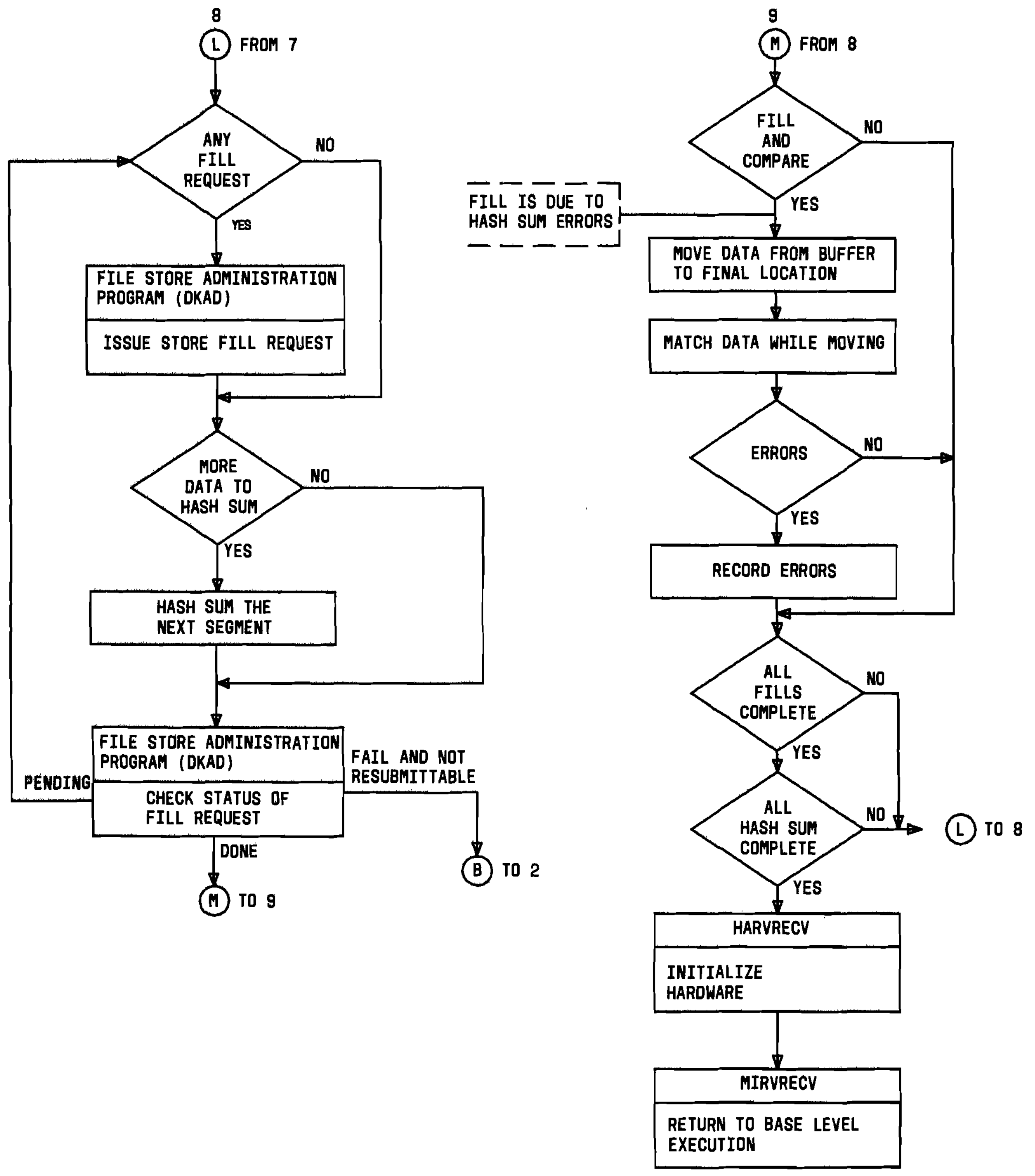


Fig. 65—Processor Configuration Recovery Level 2—Flowchart (Sheet 4 of 4)

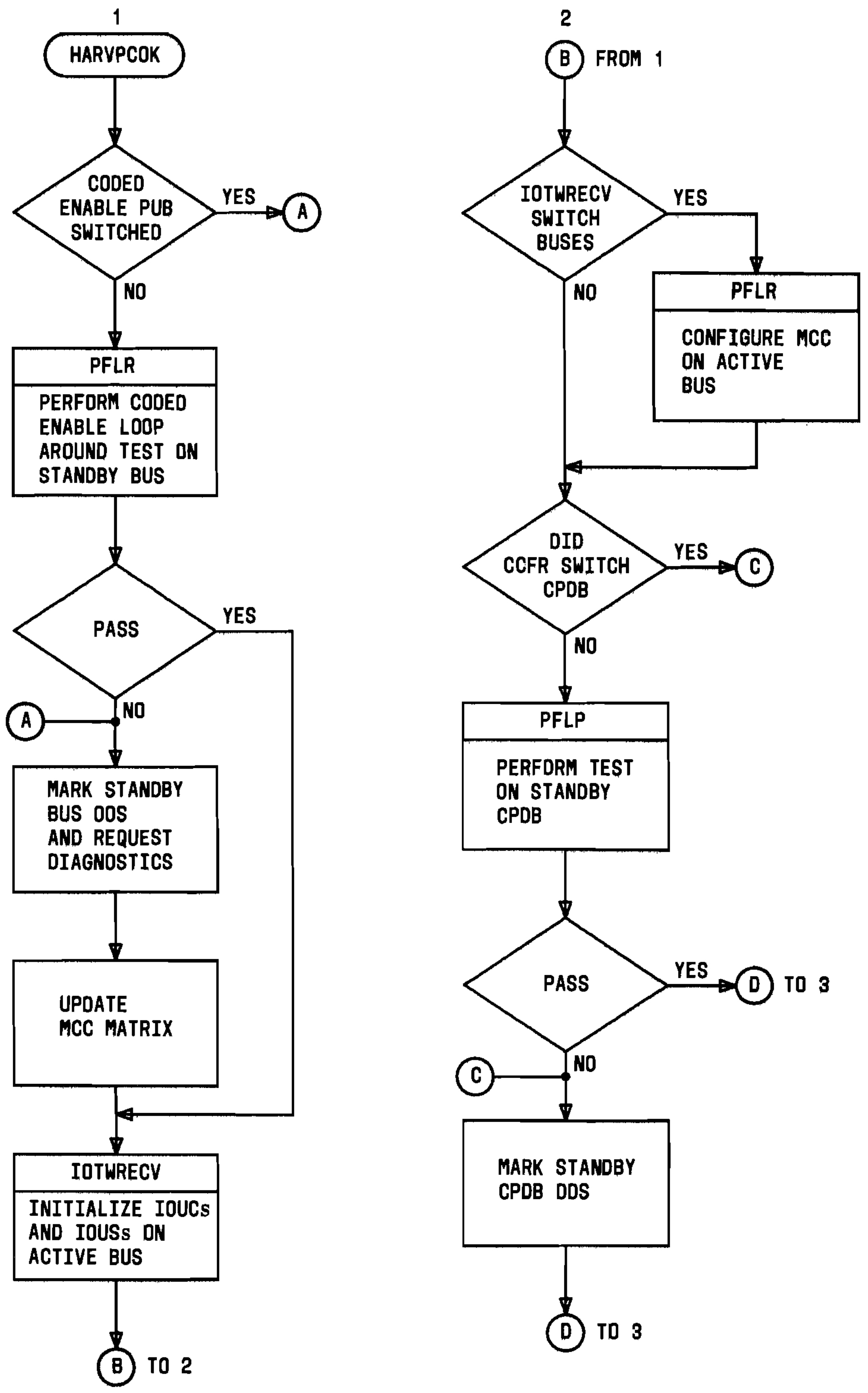

1
HARVPCOK
CODED ENABLE PUB SWITCHED
YES
A
NO
PFLR
PERFORM CODED ENABLE LOOP AROUND TEST ON STANDBY BUS
PASS
YES
A
NO
MARK STANDBY BUS OOS AND REQUEST DIAGNOSTICS
UPDATE MCC MATRIX
IOTWRECV
INITIALIZE IOUCs AND IOUSs ON ACTIVE BUS
B TO 2
2
B FROM 1
IOTWRECV SWITCH BUSES
YES
NO
PFLR
CONFIGURE MCC ON ACTIVE BUS
DID CCFR SWITCH CPDB
YES
C
NO
PFLP
PERFORM TEST ON STANDBY CPDB
PASS
YES
D TO 3
C
NO
MARK STANDBY CPDB OOS
D TO 3

Fig. 66—Hardware Initialization—Flowchart (Sheet 1 of 4)

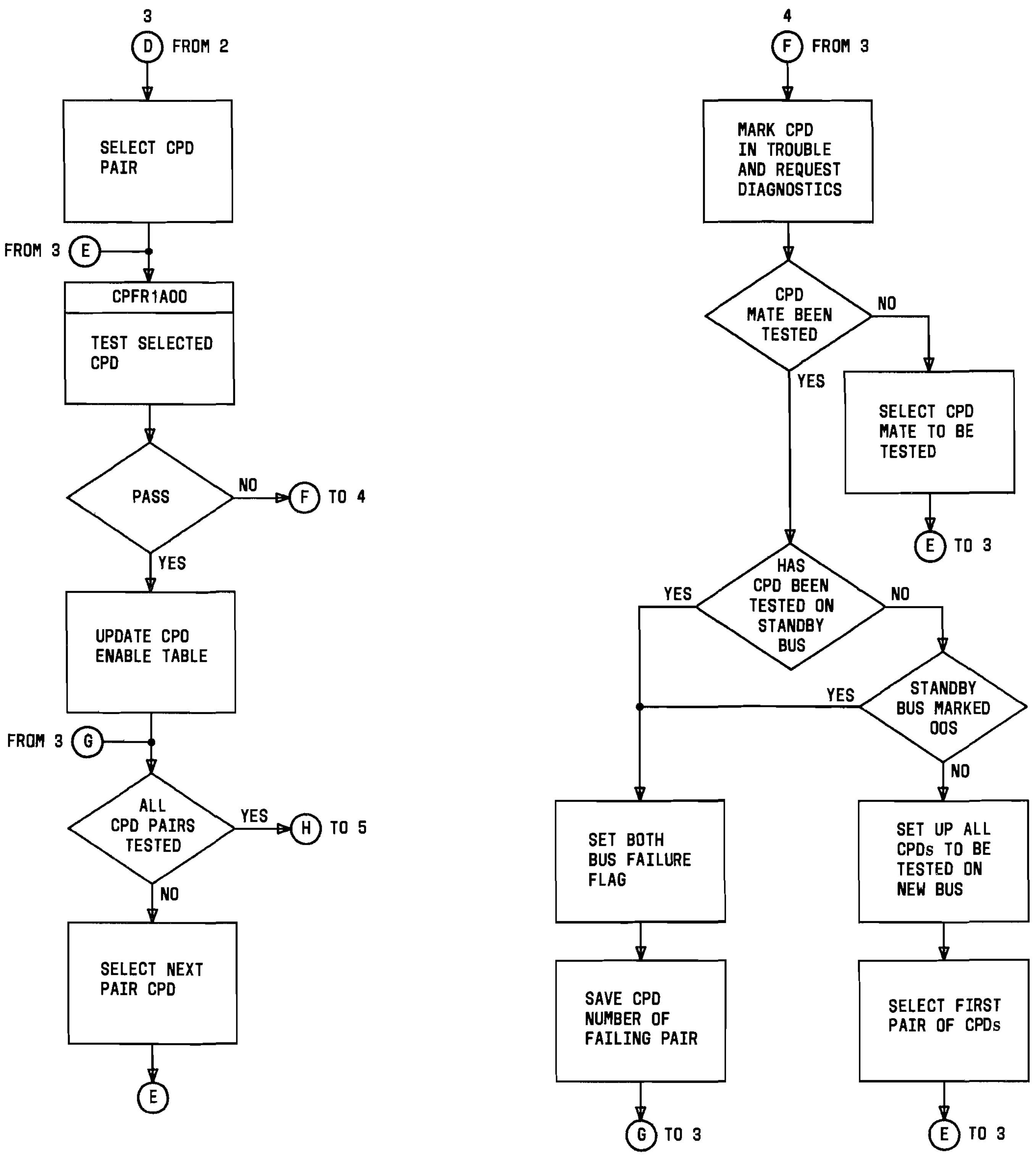


Fig. 66—Hardware Initialization—Flowchart (Sheet 2 of 4)

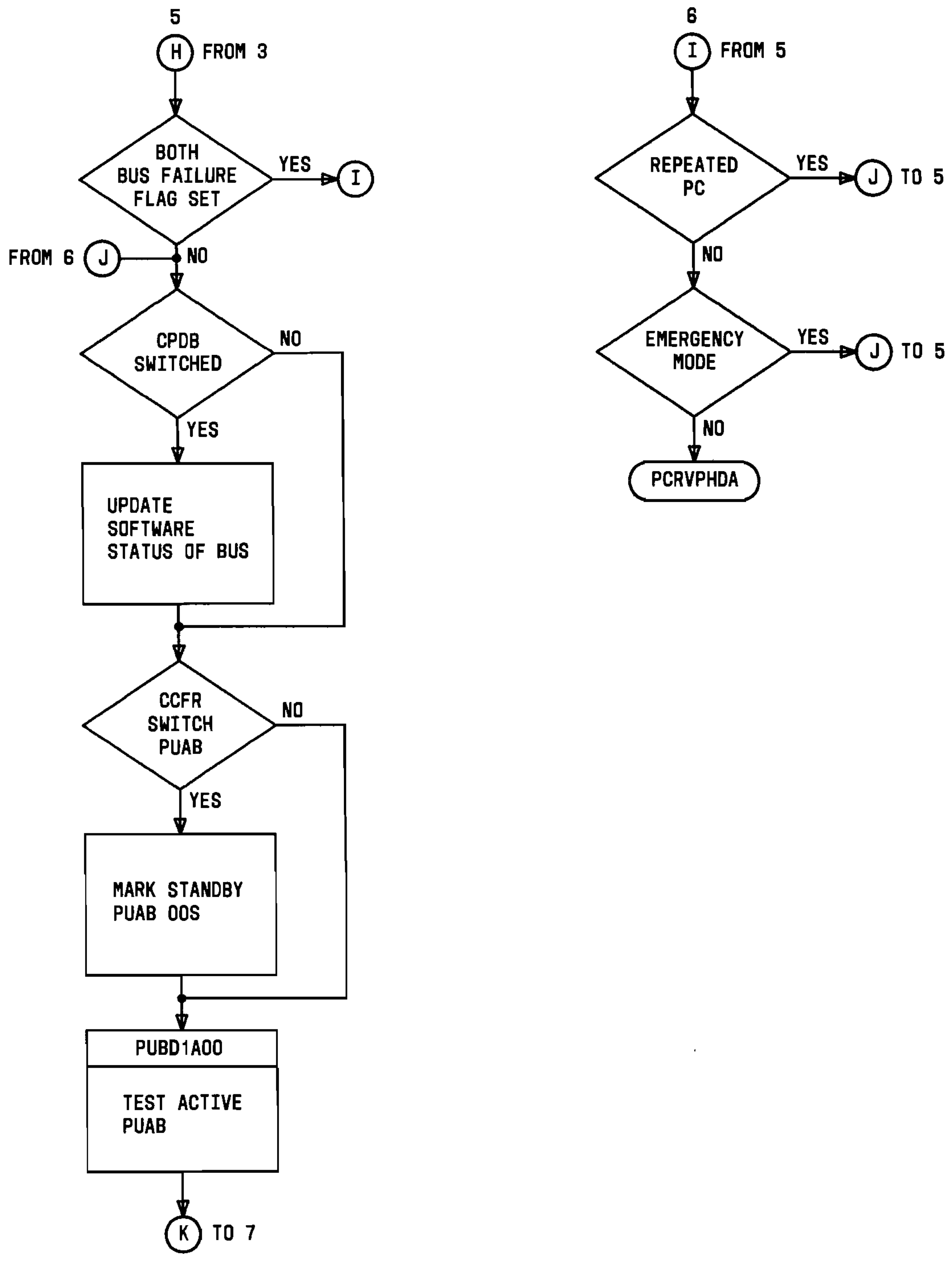


Fig. 66—Hardware Initialization—Flowchart (Sheet 3 of 4)

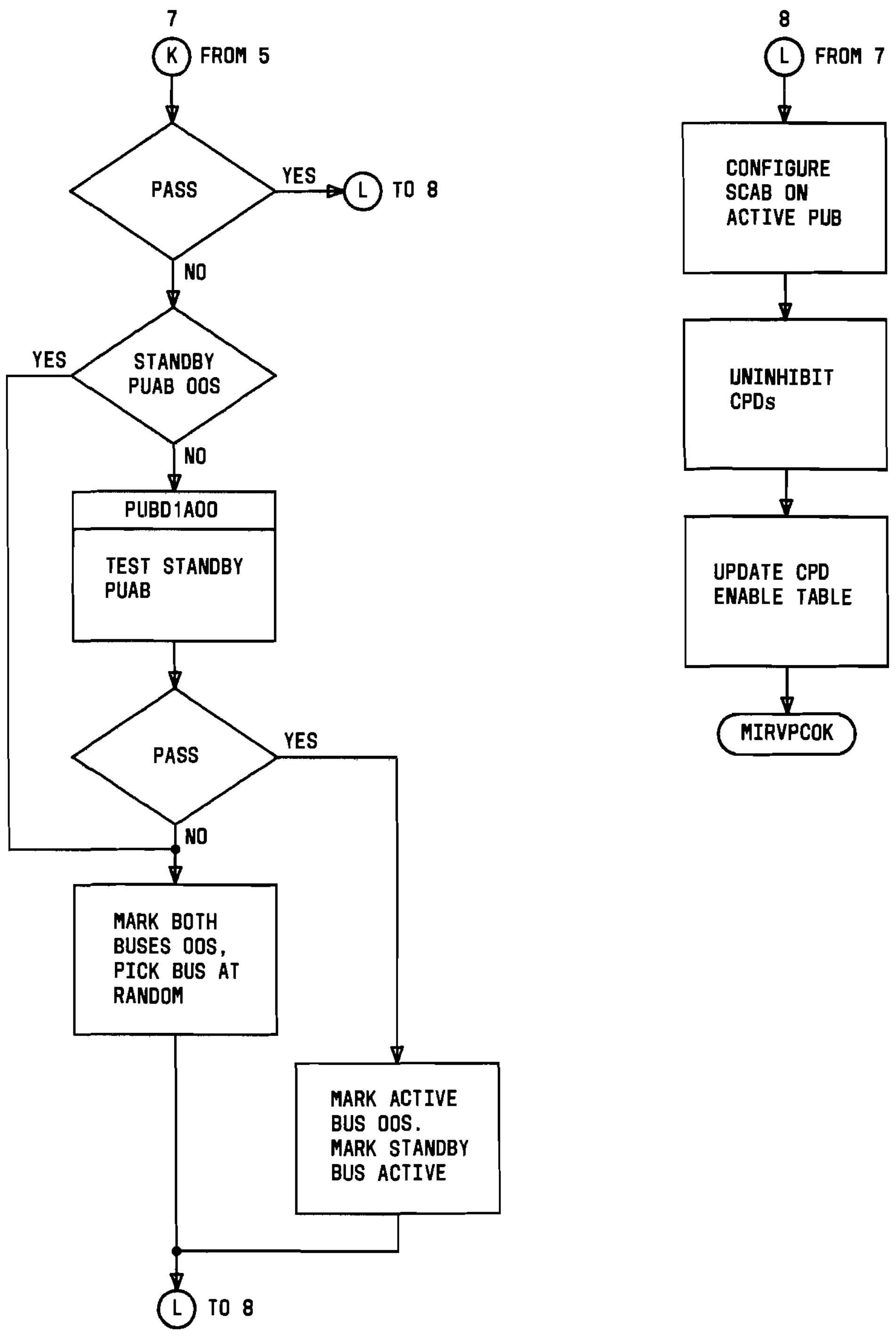


Fig. 66—Hardware Initialization—Flowchart (Sheet 4 of 4)

(b) ***Program Checks:*** These triggers are related to program check failures which cause a phase (eg, unanswered interject).

(c) ***Phase Problems:*** These triggers are related to problems encountered during a phase execution (eg, a phase exceeds a time limit).

(d) ***Automatic Interrupt Triggers:*** These triggers are related to hardware faults that cause a phase (eg, duplicated call store failure).

C. Phase Activity

Phase 1

6.65 Phase 1 is the least comprehensive of the phases and has the shortest duration. Phase 1 is completed in 2 seconds. Phase 1 has virtually no effect on calls in progress in either a stable state (ie, an established talking path) or an unstable state (eg, digit reception, ringing, etc). Any RCs in progress are removed when the phase in triggered. All nontransient data that is duplicated in the file store is hashed and corrected, if necessary. Overwrites in progress and active utility executes are removed and the appropriate interrupts are inhibited. The audits run in phase 1 include audits 3 through 11, 18, 32, 40, and 71. These audits (eg, network management audit, ring tip scan audits, etc) check and, if necessary, generate constant values within the transient data. If phase 1 is triggered by a data validation failure, additional audits are run in conjunction with audits normally requested. The additional audits are audits 24, 34, 36, 42, 43, 44, and 45. See Table E for an overview of phase activity.

Phase 4

6.66 Phase 4 is the lowest level phase that zeros a specified area of call store. This area is defined by office parameters. Calls in the stable state are maintained; however, calls in the unstable state are knocked down. Nontransient data that is duplicated in file store is hashed and corrected, if necessary. RCs in transition to call store or RCs already processed are removed. Overwrites in progress and active utility executes are removed and the appropriate interrupts are inhibited. The audits run in phase 4 include audits 0, 4-10, 16, 18, 19, 22, 32, 37-43, 46, 48, 50, 52, 58-63, and 71. Phase 4 completes in 25 seconds (see Table E).

Phase 5

6.67 Phase 5 consists of the same audits as phase 4, but it is more comprehensive. Phase 5 is the first possible pump phase; ie, processor configuration state counter must be greater than or equal to 16. This implies that the system has been hashed and if necessary pumped; ie, nontransient data in main memory has been replaced with data from file store or it has been verified via hash sums. Once all transient memory corrections have been attempted (via phases 1 and 4) and a phase still triggers, the problem is either nontransient data or hardware. Main memory and disk overwrites in the copy state are removed. Calls in the stable state are maintained while calls in the unstable state are knocked down. RCs are handled the same as in phase 4. An office parameter defined area of call store is zeroed. The utility executes are removed and the appropriate interrupts are inhibited. When phase 5 is repeated, additional interrupts are inhibited (see Table D). Phase 5 completes in approximately 25 seconds.

Phase 6

6.68 Phase 6 which can only be initiated manually is the most comprehensive and drastic phase. All calls, whether stable or unstable, are knocked down. The entire network and software subsystem is placed in the idle mode. Phase 6 like phase 5 is a pump phase. The processor configuration state counter must be greater than or equal to 16. Main memory and disk overwrites in the copy state are removed. Phase 6 also zeros an office parameter defined area of call store and removes any utility executes in progress. The H-, G-, F-, K-, and D-level sources are inhibited. When phase 6 fails, it steps down to run a phase 5 and it is treated as a repeated phase 5 and additional interrupts are inhibited (see Table E). Phase 6 completes in approximately 32 seconds.

EMERGENCY MODE CONTROL SOFTWARE

6.69 Failure of the processor configuration recovery sequence to establish a viable processor configuration necessitates manual recovery procedures. These are invoked through controls at the MCC. The first manual recovery step taken consists of establishing a basic configuration using the override control keys and requesting the second level of processor configuration recovery. The override control keys have the advantage over the basic configuration sequence of being able to force a basic

TABLE E

1A PHASE ACTIVITY

ACTIVITY \ PHASE		1	4	5	6
OVERVIEW		Constant's Audits	Zero CS Data	First Pump Phase	Manual Only
DURATION		1-2 sec	25 sec	25 sec	32 sec
EFFECT ON NONTRANSIENT DATA		Hash/Correct	Hash/Correct	Pump PC (state ≥ 16)	Pump PC (state ≥ 16)
EFFECT ON CALL IN PROGRESS	STABLE	None	None	None	Knocked Down
	UNSTABLE	None	Knocked Down	Knocked Down	Knocked Down
RCs IN PROGRESS		Canceled	Canceled	Canceled	Canceled
CS ZEROED		None	Same Compool and Parameter Defined Data	Same Compool and Parameter Defined Data	Same Compool and Parameter Defined Data
UTILITY EXECUTES AND LIBRARY ACTIVITY		Removed	Removed	Removed	Removed
OVERWRITES IN PROGRESS		Main Memory in Copy State Removed	Main Memory in Copy State Removed	Main Memory and Disk in Copy State Removed	Main Memory and Disk in Copy State Removed
INTERRUPTS INHIBITED		H, G, F, K, and D (auxiliary unit read/write failure only)	H, G, F, K, and D (auxiliary unit read/write failure only)	First Phase 5 — Same as phase 4 Repeated phase 5 — H, G, F, E, K, D (all sources) and B (GBP sources only)	As requested
AUDITS RUN		3, 4, 5, 6, 7, 8 9, 10, 11, 18, 32, 40, 71	0, 4, 5, 6, 8, 9 10, 16, 17, 18 19, 22, 32, 37, 38, 39, 40, 41, 42, 43, 46, 48, 50, 52, 58, 59, 60, 61, 62, 63, 71	Same as phase 4	0, 4, 5, 6, 7, 8, 9, 10, 12, 14, 16, 18, 19, 20, 22, 24, 26, 32, 34, 36, 37, 38, 39, 40, 41, 42, 43, 46, 48, 50, 58, 60, 61, 62, 63, 64, 71

configuration which fault recovery programs cannot change.

6.70 If the system still fails, a final set of manual recovery procedures is initiated. It involves forcing the system into an emergency mode of operation in which only manually initiated tasks are executed. All other tasks including call processing are discarded. In the event of excessive call store, program store, or file store failures, this emergency mode can be entered with a minimal processor configuration (EMERMIN) that consists of a central control and only sufficient memory to execute maintenance tasks. The emergency mode may also be entered with a complete memory (EMERFULL) in the event of peripheral faults or program problems that cause a loss of system sanity (see Fig. 67).

6.71 Failure to recover system sanity through the override controls may be due to mutilated nontransient data in both the main memory stores and file data. Therefore, the next step in manual recovery is to reload this data from tape. This is called system reinitialization. The program initiates the load of data from tape and directs programs loaded with this data to configure a complete processor.

PROCESSOR RECOVERY SOFTWARE

A. General

6.72 The processor fault recovery programs (Fig. 68) are normally entered as a result of a maintenance interrupt. However, they may also be entered on interject or via the input/output handler program, via routine exercise programs, via the maintenance control program for deferred (base level) fault recovery testing, or via manual requests from the TTY. But, the primary purpose of these programs is to restore the system to call processing in the face of system errors or faults.

6.73 The fault recovery programs are designed to isolate faulty units or subsystems rather than to identify replaceable components. These programs recognize and isolate most call-affecting faults during a single interrupt interval.

6.74 The basic techniques of fault recovery strategy are centered around rapid resolution of problems and quick return to normal system operation. The fault recovery programs report error data to the error analysis programs. Error analysis maintains a history of interrupts and associated data.

6.75 After the fault recovery program has selected a working configuration of hardware, the program must perform several "housekeeping" tasks. The program must set appropriate flags that will cause base level after the system has returned to call processing. Also, the program must record the actions it has taken in the appropriate error analysis data history.

6.76 Finally, the fault recovery program initiates output messages to convey its actions to maintenance personnel. If several interrupts have failed to resolve a persistent problem, the output messages may be utilized to supplement the automatic error analysis. The maintenance personnel may analyze the output messages and select a working configuration of hardware manually.

6.77 The 1A Processor community has six components, each of which could default at anÿ time for various reasons. Fault recovery programs are written for each component and these programs interface with many other programs in an effort to recover the system sanity. Central control, call store, program store, file store, auxiliary unit buses, and data units are all subject to faults and their recovery is done on an individual basis.

B. Central Control Fault Recovery

6.78 Duplicated central controls are the primary functional elements of the processor community. The central controls interface with all internal and external signal and control buses and provide the processing capability for the system. For reliability purposes, the two central controls are connected in parallel. Either one can control system operation. The normal system configuration provides for the two central controls to operate in step, each performing matching checks on the other.

6.79 One central control functions as the active unit and the other functions as the standby. During this normal mode of operation, both central controls are matched to ensure that they execute the same instructions, receive the same data, and make the same conditional decisions. In the event that the active central control malfunctions, the standby central control is designated active and assumes control of processor functions (the switch of active and standby central controls may be accomplished automatically under program control, by the processor configuration hardware, or by manual activation from the control and display panel).

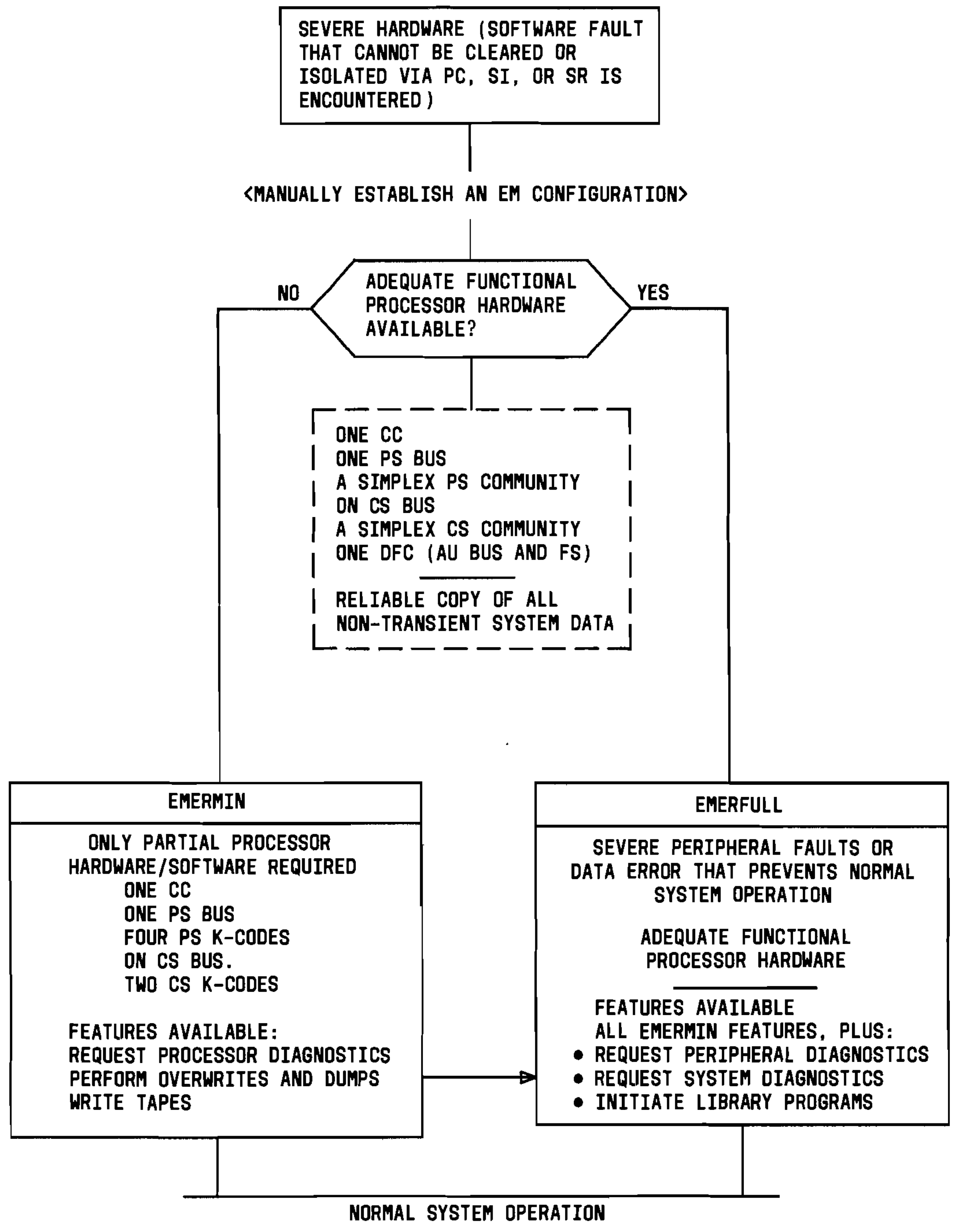

SEVERE HARDWARE (SOFTWARE FAULT THAT CANNOT BE CLEARED OR ISOLATED VIA PC, SI, OR SR IS ENCOUNTERED)
<MANUALLY ESTABLISH AN EM CONFIGURATION>
NO
ADEQUATE FUNCTIONAL PROCESSOR HARDWARE AVAILABLE?
YES
ONE CC
ONE PS BUS
A SIMPLEX PS COMMUNITY ON CS BUS
A SIMPLEX CS COMMUNITY
ONE DFC (AU BUS AND FS)
RELIABLE COPY OF ALL NON-TRANSIENT SYSTEM DATA
EMERMIN
ONLY PARTIAL PROCESSOR HARDWARE/SOFTWARE REQUIRED
ONE CC
ONE PS BUS
FOUR PS K-CODES
ON CS BUS.
TWO CS K-CODES
FEATURES AVAILABLE:
REQUEST PROCESSOR DIAGNOSTICS
PERFORM OVERWRITES AND DUMPS
WRITE TAPES
EMERFULL
SEVERE PERIPHERAL FAULTS OR DATA ERROR THAT PREVENTS NORMAL SYSTEM OPERATION
ADEQUATE FUNCTIONAL PROCESSOR HARDWARE
FEATURES AVAILABLE
ALL EMERMIN FEATURES, PLUS:
• REQUEST PERIPHERAL DIAGNOSTICS
• REQUEST SYSTEM DIAGNOSTICS
• INITIATE LIBRARY PROGRAMS
NORMAL SYSTEM OPERATION

Fig. 67—Emergency Mode Configuration

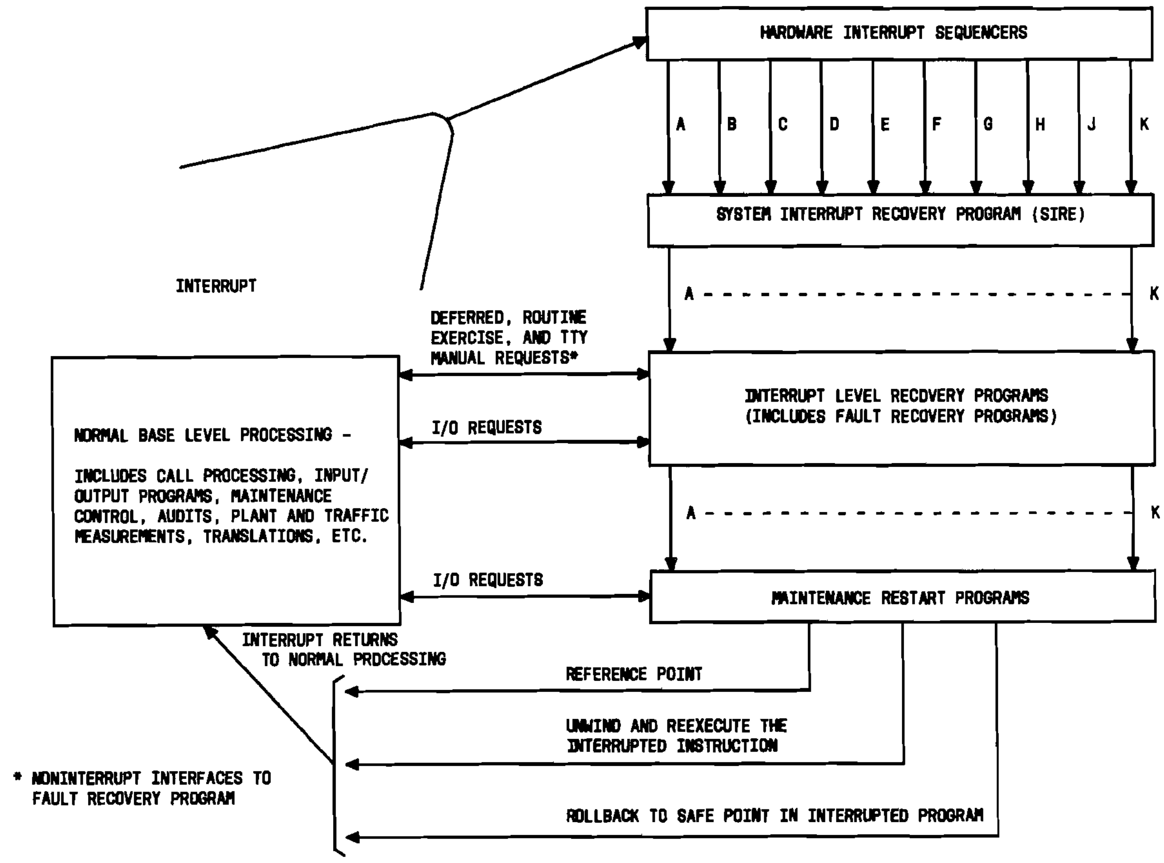

HARDWARE INTERRUPT SEQUENCERS
A
B
C
D
E
F
G
H
J
K
SYSTEM INTERRUPT RECOVERY PROGRAM (SIRE)
INTERRUPT
DEFERRED, ROUTINE EXERCISE, AND TTY MANUAL REQUESTS*
I/O REQUESTS
INTERRUPT LEVEL RECOVERY PROGRAMS (INCLUDES FAULT RECOVERY PROGRAMS)
NORMAL BASE LEVEL PROCESSING -
INCLUDES CALL PROCESSING, INPUT/OUTPUT PROGRAMS, MAINTENANCE CONTROL, AUDITS, PLANT AND TRAFFIC MEASUREMENTS, TRANSLATIONS, ETC.
I/O REQUESTS
MAINTENANCE RESTART PROGRAMS
INTERRUPT RETURNS TO NORMAL PROCESSING
REFERENCE POINT
UNWIND AND REEXECUTE THE INTERRUPTED INSTRUCTION
* NONINTERRUPT INTERFACES TO FAULT RECOVERY PROGRAM
ROLLBACK TO SAFE POINT IN INTERRUPTED PROGRAM

Fig. 68—General Program Flow for Fault Recovery

6.80 System troubles are normally detected by trouble-detection circuits and the call processing interject. As stated earlier, its primary purpose is to verify the integrity of the active central control; however, the tests that are used to perform the verification vary and depend upon the conditions under which the program is entered.

6.81 Recovery is organized around a common control program (Fig. 69) which calls one or more independent test routines. Control is based on a dynamic control word. The recovery programs initialize the control word to specify which test routines to run on each entry. The control word is normally set to all zeros which is recognized as an invalid entry condition. The control word consists of three fields of bits. Within the first field, each bit has a one to one relationship to a test routine which tests a portion of the central control. The second field consists of a single bit and signifies a special test procedure used only on B-level interrupts caused by a pulse source failure. The remaining bits make up the third field and indicate the origin and the termination of the request.

6.82 The program has several different entry points, each with its own unique requirements. A preprocessor program is provided for each input to prepare the necessary information and perform the required initialization before entering the common control program. At the conclusion of the common control program, a termination program is provided for each entry. There is also a special termination program in case the control programs find the control word to contain all zeros. These terminating programs perform access tests on other subsystems, update status words, request subsystem normalization, update error counters, and in general perform cleanup and housekeeping tasks before returning the system to call processing.

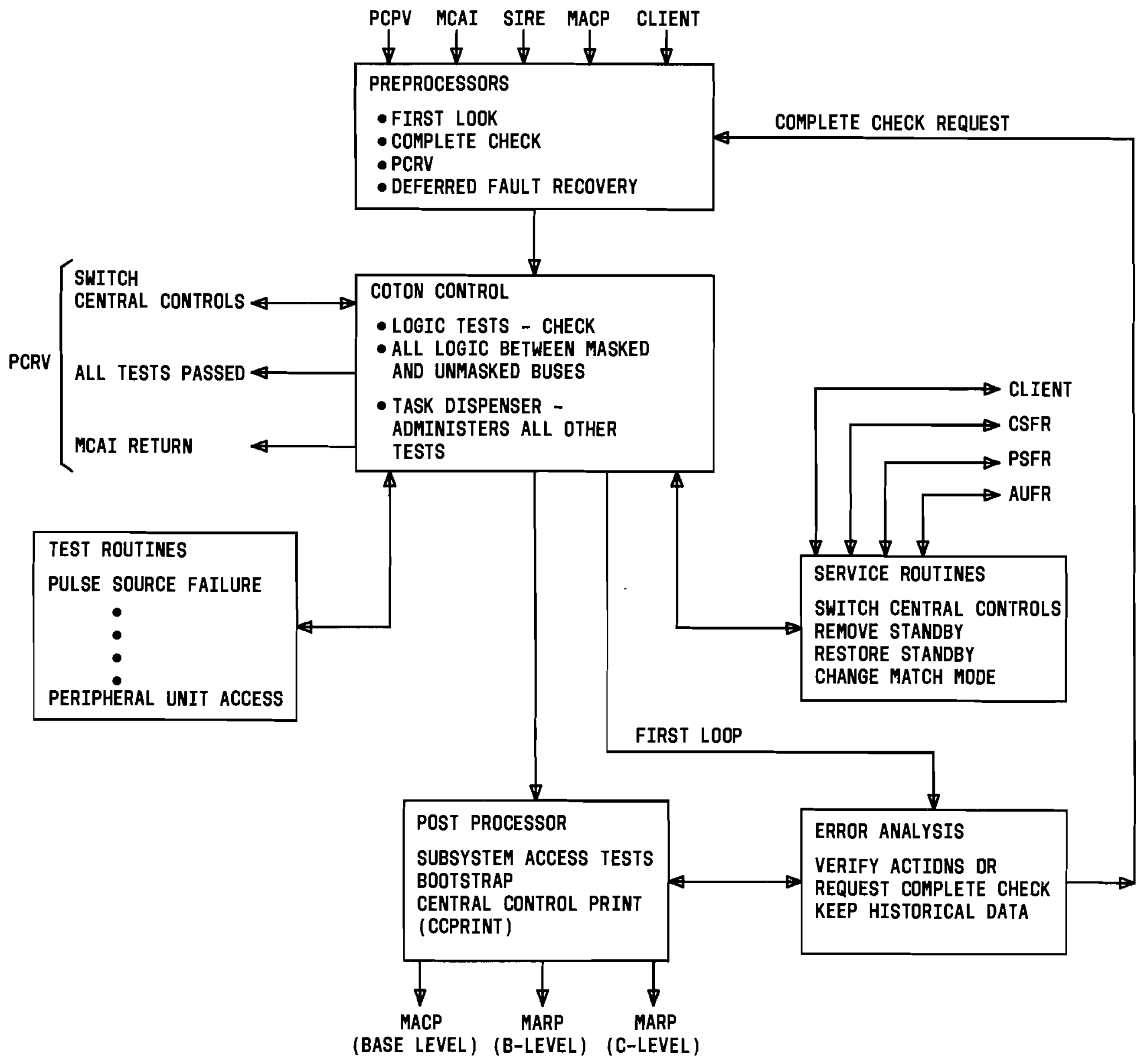


Fig. 69—Central Control Fault Recovery—Flowchart

C. Call Store Fault Recovery

6.83 The primary purpose of call store recovery is to return the system to normal call processing (Fig. 70) as quickly as possible after a fault or error has been detected in the call store community. Therefore, whenever possible, the program will remove the faulty store from service on a "first-look" basis. The first-look approach utilizes error indicators in the central control, the bus configuration, and the store status to identify the faulty store.

6.84 The first-look approach replaces the faulty unit with a duplicate if one is available. If a duplicate is not available, the recovery selects a store (from the duplicated call stores) and initiates a copy of the suspected unit into the selected unit. If tests are unable to detect trouble within the faulty store, it is restored to service before the update completes. When the update is completed, the updated store is placed into service and the suspect store is removed and diagnosed. If the faulty unit fails again before

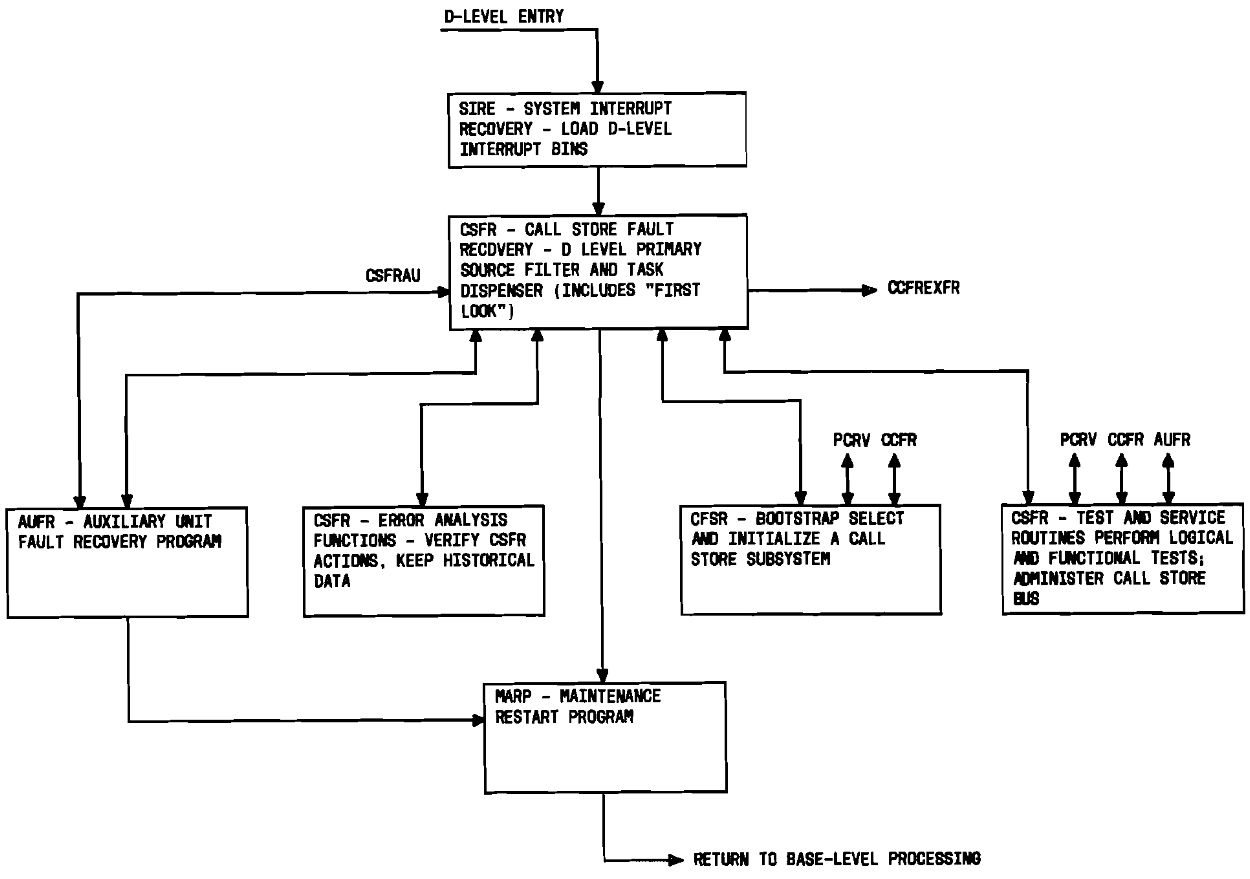

D-LEVEL ENTRY
SIRE - SYSTEM INTERRUPT RECOVERY - LOAD D-LEVEL INTERRUPT BINS
CSFR - CALL STORE FAULT RECOVERY - D LEVEL PRIMARY SOURCE FILTER AND TASK DISPENSER (INCLUDES "FIRST LOOK")
CSFRAU
CCFREXFR
PCRV CCFR
PCRV CCFR AUFR
AUFR - AUXILIARY UNIT FAULT RECOVERY PROGRAM
CSFR - ERROR ANALYSIS FUNCTIONS - VERIFY CSFR ACTIONS, KEEP HISTORICAL DATA
CFSR - BOOTSTRAP SELECT AND INITIALIZE A CALL STORE SUBSYSTEM
CSFR - TEST AND SERVICE ROUTINES PERFORM LOGICAL AND FUNCTIONAL TESTS; ADMINISTER CALL STORE BUS
MARP - MAINTENANCE RESTART PROGRAM
RETURN TO BASE-LEVEL PROCESSING

Fig. 70—Call Store Fault Recovery—Program Flow and Interfaces—Simplified

the update is completed, the faulty unit is removed from service and the system must wait for the successful completion of the substitute store's update from a backup copy stored on file store.

6.85 However, when the failing store is not duplicated and a store is not available for selection as a substitute, the first-look approach is not used. Call store fault recovery uses the error history of the failing store to determine the action to be attempted. If the store's error history is acceptable, call store fault recovery corrects the failing word, performs a complete access test of the store, and restores it to service.

6.86 An unacceptable error history or a failure of the complete access test causes call store fault recovery to perform a bootstrap of the call store community. The bootstrap routine attempts to assemble a complete copy of the call store information by using only stores that pass the bootstrap tests. If necessary, call stores will be used regardless of their status prior to the failure. Should the bootstrap fail to establish a valid copy of call store, a program transfer is made to the processor configuration recovery program to switch central controls (B-level interrupt).

6.87 The call store service routines and bootstrap routines may also be entered from central control fault recovery, auxiliary unit fault recovery, and processor configuration programs. These programs may use the service routines to verify access to call stores. In addition, central control fault recovery may enter the call store bootstrap routine if one or more call store memory blocks are not error-free or are not provided in the configuration that has been established. Therefore, call store bootstrap is entered to recover a valid call store configuration.

D. Program Store Fault Recovery

6.88 The 1A Processor program store community is made up of a number of individual program stores. The number of program stores varies according to the type and size of the switching office installation. System software (including program store fault recovery) can accommodate 33.

6.89 The central controls access the program stores via duplicated program store buses that interconnect every program store frame with both central controls. (The buses may have as many as two branches.)

6.90 Each program store word location is identified by a unique address. This address consists of a K-code and a data location address. The K-code portion of the address identifies the specific program store to be addressed. The data location identifies the specific location to be accessed within the program store.

6.91 Two program stores are normally designated as spares (called rovers) and can be assigned to replace any program store which malfunctions. During normal operation, the spare program stores contain duplicate copies of information stored in program store.

6.92 The primary purpose of program store fault recovery is to return the system to normal call processing (Fig. 71) as quickly as possible after a fault or error condition has been detected in the program store community. Therefore, whenever possible, the fault recovery will remove the faulty store on a first-look basis. The first-look approach works in the same manner as call store fault recovery first look. The first-look approach utilizes error indicators in the central control, the bus configuration, and the store status to identify the faulty store.

6.93 When the trouble is located in duplicated program store, the suspect unit is removed from service, and the remaining is set to operate as if it were not duplicated. An unduplicated block of memory requires further analysis. Program store fault recovery attempts to find a rover store that can be loaded with a copy of the suspect memory block. If it is able to select a rover store and initiate the copy of the suspect store, it then checks the error history (kept by program store fault recovery) of the suspect store. If the history is acceptable, the store is left in service until the rover update is completed.

6.94 If the error history is unacceptable, the suspected unit is removed from service and the system must wait until the rover is filled from file store. Also, the program checks to see if a previous error has resulted in a rover store being prepared as a duplicate for the suspected block of memory. If a rover has been updated, the rover is placed in service and the suspect unit is removed. If a rover is in the process of being updated, the system waits for the update to be completed and replaces the suspect unit with the rover store.

6.95 After a configuration of program stores has been selected, an access test is performed on each memory block to verify the integrity of the program store community. Failure of the access test after the suspect store has been removed from service causes the program to transfer to the program store bootstrap routine.

6.96 Bootstrap attempts to assemble a complete copy of program store using only stores which pass the bootstrap qualifying tests. The bootstrap is considered successful if a full copy of program store (with or without the use of rover stores) has been assembled. Failure of bootstrap results in a transfer to the processor configuration recovery program to switch central controls (B-level interrupt). After a successful fault recovery, control is returned to normal processing.

E. Auxiliary Unit Fault Recovery

6.97 The 1A Processor system has a bus system which enables autonomous processing units to access the call store and program store bus system of the central control. The autonomous processing units are referred to as auxiliary units and there may be as many as 16 auxiliary units on the auxiliary unit bus. The auxiliary unit bus is linked to the call store and program store buses by special hardware in the central control which is called the auxiliary unit bus sequence (AUBSQ). The AUBSQ resolves bus occupancy conflicts among auxiliary units on the auxiliary unit bus and resolves bus, store, or AU occupancy conflicts between the central control or any auxiliary unit on either the auxiliary unit, call store, or program store bus. This document will refer to the auxiliary units and the AUBSQ as the auxiliary unit bus system.

6.98 The auxiliary unit bus system will have at least two and a maximum of four file store controllers in the file store environment. A file store controller may control from one to four disk files on

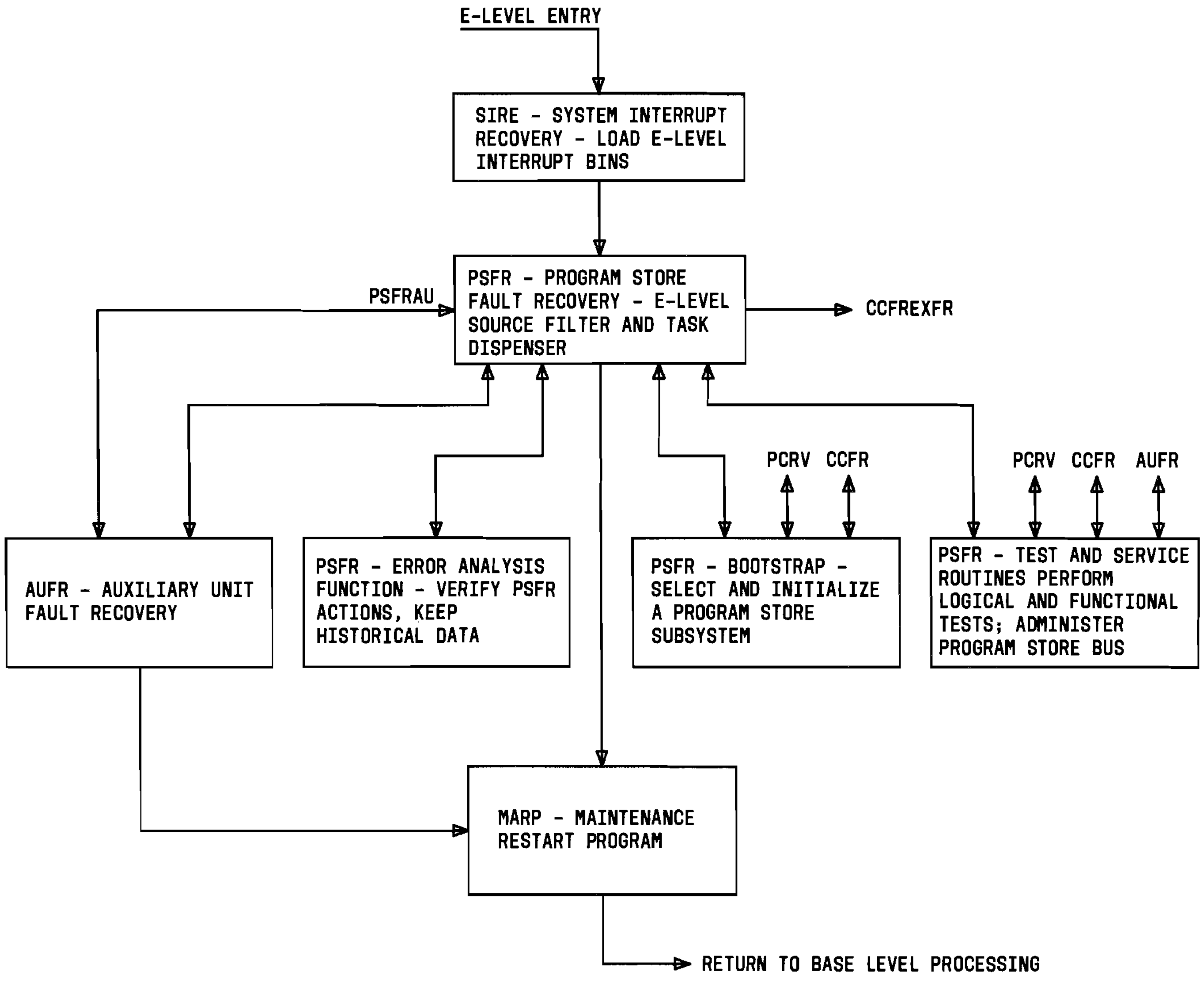

E-LEVEL ENTRY
SIRE - SYSTEM INTERRUPT RECOVERY - LOAD E-LEVEL INTERRUPT BINS
PSFRAU
PSFR - PROGRAM STORE FAULT RECOVERY - E-LEVEL SOURCE FILTER AND TASK DISPENSER
CCFREXFR
PCRV CCFR
PCRV CCFR AUFR
AUFR - AUXILIARY UNIT FAULT RECOVERY
PSFR - ERROR ANALYSIS FUNCTION - VERIFY PSFR ACTIONS, KEEP HISTORICAL DATA
PSFR - BOOTSTRAP - SELECT AND INITIALIZE A PROGRAM STORE SUBSYSTEM
PSFR - TEST AND SERVICE ROUTINES PERFORM LOGICAL AND FUNCTIONAL TESTS; ADMINISTER PROGRAM STORE BUS
MARP - MAINTENANCE RESTART PROGRAM
RETURN TO BASE LEVEL PROCESSING

Fig. 71—Program Store Fault Recovery—Program Flow and Interfaces—Simplified

which a large amount of data can be stored. A file store controller is a wired logic processor which will process central control request(s) to transfer a data block between the relatively slow serial access memory of a disk file and the fast random access memory of a call store or program store. For reliability, the memory content of one set of disk files associated with a file store controller will be duplicated on an identical set of disk files associated with another file store controller. Collectively, all file store controllers with associated disk files are referred to as the file store system. The file store system serves as a primary data backup and bulk data storage facility for program, translation, and other information for the 1A Processor system.

6.99 ➧The auxiliary unit bus system, in the APS environment, will have at least 2 and a maximum of 16 APIs. The APS is a high capacity disk system for the 1A Processor. The API allows the sending and receiving of messages and blocks of data between the 1A Processor and up to eight 3B Processors. ➧

6.100 The auxiliary unit bus system will also have at least two and a maximum of four data unit selectors. A pair of data unit selectors may control from two to 16 data units. A data unit selector is similar to a file store controller except that instead of disk files, the data unit selector is designed to handle slower data devices such as tape units. The data unit system serves as a backup to the file store system for system reinitialization and as a primary facility for program updating, automatic message accounting data recording, and other functions.

6.101 The auxiliary unit fault recovery program is designed to function under a number of diverse conditions: interject, D-level interrupt, base level maintenance and other processor interrupt levels. Basically, the program employs the first-look approach to fault recovery. The first-look approach involves the retrying of the failing operations utilizing simple and fast-testing techniques. If this approach fails to identify the source of the trouble, fault recovery will then resort to more detailed testing to isolate the problem.

6.102 The auxiliary unit fault recovery program will be entered basically under three conditions. The first condition involves maintenance action for the auxiliary unit bus system. This maintenance action will ordinarily be initiated through the interject request mechanism instead of the normal maintenance interrupt control hardware sequencer. This method is used because auxiliary unit processing is independent of central control processing and can be momentarily deferred without degrading system performance.

6.103 In the second condition, the fault recovery will also be entered on D level from the call store fault recovery program when the central control encounters an auxiliary unit read/write failure. An auxiliary unit read/write failure may occur when the central control addresses an auxiliary unit and an accessing error is detected by either the central control or auxiliary unit. The central control may also address an auxiliary unit which is in a troubled state and has requested maintenance action through the interject mechanism. An auxiliary unit which makes an interject request will not respond to central control addressing until it has been restored to service by the specific type of auxiliary unit fault recovery program. The program processes D-level entries basically as it processes interject entries.

6.104 Finally, the program may also be called by TTY request or by the processor configuration or another processor fault recovery program to test or reconstruct the auxiliary unit bus system interface with the central control system.

F. File Store Fault Recovery

6.105 The 1A Processor utilizes a file store or APS to provide backup storage for program and translation data. The file store or APS is also used to store programs and data that are infrequently used and consequently are not normally kept in the program or the call store. The disk memory used by the file store possesses serial rather than random access characteristics. Because of its serial character, the time required to retrieve or store data from the file store is variable (a function of the position of the disk when the request is made). Because time required to retrieve or store data is on the order of milliseconds, it is not practical for the central control to directly access disk memory. Instead, a file store controller is provided to perform this function. The file store controller is a special purpose wired logic processor which buffers requests from the central control to read from or to write into disk memory and transfers information from disk to main memory or from main memory to disk.

6.106 Each file store contains one to four disk files. File stores are arranged in pairs, and each pair is referred to as a community. The 1A Processor software is designed to accommodate a maximum of two communities. File store 0 (on bus 0) and file store 1 (on bus 1) make up one community; file store 2 (on bus 0) and file store 3 (on bus 1) make up another community.

6.107 When the fault recovery program for file store is entered, it determines the type of error, increments the counter, and checks to see if the counter limit has been reached. If the counter had not reached its limit, the error is recorded, and a return to the calling program is executed. If the counter has reached its limit, the file controller or disk file is removed from service.

6.108 Whenever a file store controller or disk file is removed from service, a diagnosis is requested. Because the removal of a file store controller from service could mean that as many as four disk files would be inaccessible, every effort will be made to leave in service as many disk files as possible. Therefore, when the source of the trouble may be either the file store controller or disk file, only the disk file will be removed from service. If it is determined that the disk file only contains an error-prone record, the record will be rewritten and verified instead of immediately requesting a diagnosis.

6.109 Because the read or write of a disk file record is a relatively time-consuming process, all file store fault recovery maintenance actions that require a disk file access operation are deferred. All disk file access operations are processed as a standard job request through the normal file store administration program routines. Furthermore, those disk file related error sources which are expected to have a relatively high rate of occurrence are processed through the status failure report mechanism. Consequently, they will not require the more time-consuming and service-affecting actions of normal maintenance procedures.

G. ♦Attached Processor System Single Strategy Fault Recovery (SSFR)

6.110 Fault recovery of 1A Processor subsystems has traditionally been handled by a single program for each subsystem. However, with the addition of the APS, fault recovery is handled by two programs with a common recovery control. Fault recovery in the APS is divided into two major categories. The first category is fault recovery on interrupt or interject level and is handled by either the auxiliary unit fault recovery program (AUFR) or the attached processor fault recovery program (APFR). The second is fault recovery on base level that is handled by the APFR. The size and complexity of a fault recovery package for each major category resulted in a single recovery package serving both functions. The single recovery package is called the single strategy fault recovery (SSFR). It provides common recovery control for both interrupt or interject level faults and base level maintenance faults.

6.111 The SSFR does fault recovery tasks for faults occurring in either the active or the standby APIs. These faults or failures may be initiated by either the 1A or the 3B processor. The SSFR is divided into four major areas:

- Interrupt and interject control
- Base level maintenance control
- Common recovery control
- Timing administration.

The interrupt and interject control takes place in the AUFR, and the base level maintenance control is handled within the APFR.

APS Organization

6.112 The APS replaces the disk file system on either the No. 1A or No. 4 ESS switch. The APS consists of one to eight 3B processors connected to the 1A Processor through an API system. The APS provides the 1A Processor disk access to a high-capacity 3B disk system. The API allows the sending and receiving of messages and blocks of data between the 1A and 3B Processor Systems. The API supports the APCL protocol between the 1A and 3B processors. The APCL protocol has both efficient block transfer and message-handling capabilities. The APCL protocol also includes a high-priority maintenance message communication capability that is supported by the API. These messages are communicated in a closely coupled, synchronous, high-priority way by using the 3B input/output interrupt and the 1A auxiliary unit bus maintenance interject mechanisms.

6.113 The APS includes attached processor message handlers on both the 1A and 3B sides of the API (Fig. 72). Also included are the file manager interface, the file manager, the disk driver, and the disk file controller, all on the 3B side.

Interrupt and Interject Control

6.114 The AUFR program is the interrupt and interject control program for the SSFR. All auxiliary unit interjects and D-levels are first handled by AUFR; consequently, AUFR processes the interject or interrupt and tries to isolate the problem. The problem may be in the central control, the main memory, the auxiliary unit bus or in an individual auxiliary unit. In the APS version of the 1A Processor, the auxiliary units are the API and the data unit selector.

6.115 If the AUFR determines the fault was caused by an auxiliary unit, AUFR communicates with the fault recovery programs for the faulty auxiliary unit. The AUFR program communicates with the unique fault recovery programs through a transfer vector table. For interrupt and interject processing, AUFR requests the unique fault recovery programs to do these tasks:

- Load unique bins
- Process the unique trouble
- Report data
- Update plant measurements.

If the API was the faulty auxiliary unit, APFR is the unique fault recovery program and is called to do the above tasks.

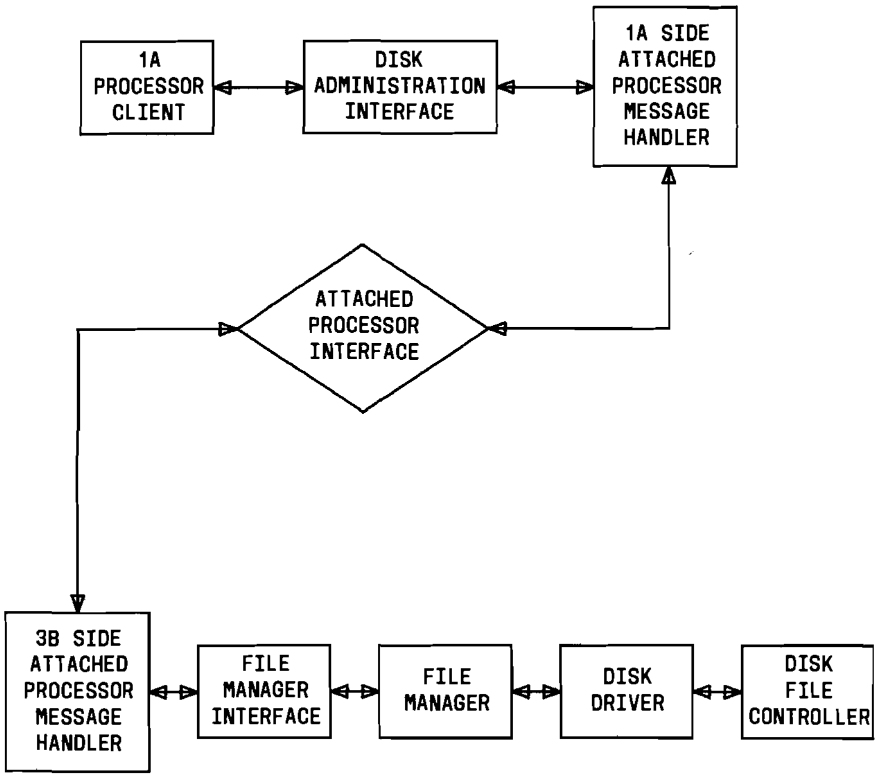

1A PROCESSOR CLIENT
DISK ADMINISTRATION INTERFACE
1A SIDE ATTACHED PROCESSOR MESSAGE HANDLER
ATTACHED PROCESSOR INTERFACE
3B SIDE ATTACHED PROCESSOR MESSAGE HANDLER
FILE MANAGER INTERFACE
FILE MANAGER
DISK DRIVER
DISK FILE CONTROLLER

Fig. 72—Attached Processor Interface Layout

Base Level Maintenance Control

6.116 The APFR program is the base level maintenance control program for the SSFR. During normal operations of the APS, there is continual checking for errors in these major areas:

- Link integrity monitor
- APS message handler
- System audit of disk.

6.117 The link integrity monitor does a check every second on the active and standby links to the 3B. Base level maintenance is called when there is no access to the 3B or the API through either the active or standby link.

6.118 The APS message handler calls a service routine within APFR to get the K-code of the active API. If an active API cannot be found, a failure is returned to the message handler. The message handler will then call base level maintenance control.

6.119 There is an audit done by the System Audit for File Store Administration Program (SADK) that monitors disk activity to ensure that jobs are being completed. If jobs are being accepted but not completed within a certain time, base level maintenance is called.

6.120 The base level maintenance control routine, APFRBLM, does three primary functions:

(a) Gets a base level maintenance report printed on the TTY

(b) Stops all auxiliary units

(c) Sets all interrupt inhibits.

Common Recovery Control

6.121 The common recovery control routine, APFRTBL, may be called from either Interrupt and Interject Control or Base Level Maintenance Control. The purpose of this routine is to determine the error and to recover from the error condition. The error and recovery information is formatted into proper form for printing on the TTY. The recovery interaction between APFRTBL and AUFR is important since the AUFR routine, known as double trouble (AUFRDTBL), is the backup recovery used by APFR.

6.122 Common recovery control functions are:

- Error detection
- Error analysis
- Error recovery
- Error termination.

6.123 The error information word is built during error detection. This word is used to record as much information about the fault condition as possible. This information is used by the recovery module to determine which course of action to take for recovery.

6.124 One of the basic strategies of error recovery is to initialize the buffers only when absolutely necessary. If recovery can be made without initializing the buffers, jobs to and from the 3B will not be affected; but, if the buffers require initializing, all the jobs in the buffers are lost.

6.125 Another strategy of error recovery is to have a configuration of the API regardless of the state of the finite state machine (routine APFRTBL). The current active API may be reconfigured or its mate can be configured. The API that is configured will be the active API upon return to the system. There is one exception. When a fault occurs in the standby API and it is removed from service, there will not be any configuration performed on the active API. If an API is configured, the following actions take place:

(1) The peripheral interface controller (PIC) is reset.

(2) The API is informed about the location of the common buffer resources.

(3) The appropriate state for the API and update status is determined.

(4) The 3B is informed of the configuration of this API.

(5) The MCC lamps and power switch lamps are updated.

6.126 Following the recovery actions in any state, the active link to the 3B is tested to ensure communication between the 3B and 1A exists before the fault recovery ends.

6.127 Another strategy of error recovery is for all recovery routines to have a pass-fail indication. Error recovery does not assume that recovery actions are done successfully. The pass-fail indications from recovery modules allow for intelligent decisions to be made within error recovery.

6.128 The final strategy of error recovery is the recording of all recovery actions. These recovery actions are recorded in the recovery information word in memory. The recovery information word is included in the printout and is used for determining the exact recovery actions taken. If any recovery module fails, the reason for failure is saved in memory and is also included in the printout.

6.129 The error termination function used by all three states to end processing is the same for all states. It does three functions:

(1) Saves all information gathered during processing

(2) Formats information for printing

(3) Updates lamps.

Timing Administration

6.130 The common recovery control routine, APFRTBL, has three states: 0, 1, and 2. The processing of a fault may begin in any state and will end in the same state. To return the routine to state 0, a sequence timer allows the state counter to be reset to 0 after a certain time has elapsed. ◆

H. Data Unit Fault Recovery

6.131 The data unit fault recovery program performs the fault recovery tasks for the ADS consisting of data units connected to the auxiliary unit bus system. The data unit fault recovery approach consists of finding a set of data units which is capable of carrying out the normal tasks associated with data stored on tape. This determination is made by a detailed set of tests which are run on interject or interrupt level priority or on demand via a TTY message. In order to perform its functions, the fault recovery program interfaces with a number of other programs.

6.132 The fault recovery program is designed to operate under D-level interrupts, maintenance interject, base level maintenance, and on demand via TTY request. The data unit fault recovery performs the following functions:

- Removals and restorals of data units
- Tests of bus circuitry, internal data unit selector registers, and data unit selector flip-flops
- Performs configuration of the data unit community
- Administers diagnostic requests
- Administers TTY message input and output requests
- Provides a common interface for other system programs.

6.133 In some cases the fault recovery program utilizes the first-look approach to fault recovery which consists of retrying the failing operation with simple and fast-testing techniques. If this approach fails to identify the source of trouble, the program resorts to more detailed testing to isolate the problem.

6.134 The data unit fault recovery program attempts to keep the tape units normalized as much as possible. This means configuring an equal number of tape unit controllers to each data unit selector. System conditions may require that the data units be configured differently to maintain a viable auxiliary data system. The normalization scheme is exercised once a day at some nonbusy hour to equally divide the TUCs between the data unit buses. At this time, all tape unit controllers are switched from the data unit selector they have been configured with to the other data unit selector. This is done to ensure that all communication paths are exercised. The tape unit controllers and data unit selectors are also exercised (diagnosed) once a day to ensure that they are capable of being used as the need arises. This exercise is also performed at a nonbusy hour.

ERROR ANALYSIS SOFTWARE

A. General

6.135 The 1A Processor error analysis program is an on-line information storage and retrieval (data administration) program which collects system data used to aid in the resolution of difficult hardware and software problems. The data base maintained by this program is also useful in making relatively detailed assessments of system performance from a maintenance viewpoint.

6.136 A major objective of error analysis is to provide a data base comprising a relatively long-term history of system maintenance actions. Full access to this data base is provided in order to enhance the capabilities of the craftsperson for maintaining the system. The data base is particularly useful in resolving faults of a transient, intermittent, or marginal nature which are not inherently reproducible. It is also useful in the investigation of subsystem interface faults whose symptoms may at first be misleading.

6.137 The 1A Processor error analysis program does not perform automatic analyses of the data it collects. However, it has extensive search/retrieval capabilities which maintenance personnel can use for manual analysis of the data.

6.138 The 1A Processor error analysis program does not have an active role in the processor recovery scheme. Its only function while on maintenance interrupt level is the collection of pertinent failure data. Furthermore, there is no automatic control over the configuration of the processor or over the decisions of other maintenance programs.

6.139 The 1A Processor error analysis program has two basic functions:

- Data collection
- Data retrieval.

6.140 In performing the data collection function, various types of data are collected. Data which is routinely collected includes

(a) Maintenance interrupt data on A, B, C, D, E, and F levels. K-level interrupt data is collected from the application program if it is available.

(b) Error stop data.

(c) Maintenance interject data.

(d) Base level maintenance data.

(e) Diagnostic summaries.

(f) Phase histories.

(g) Deferred central control fault recovery failure data.

(h) Writable store audit failure reports.

(i) Application program error analysis "past history" tables if available.

(j) Application program daily plant measurements if available.

Data which may be collected on demand includes:

(a) Raw diagnostic data.

(b) Selected traffic and plant measurement reports (from the application program).

(c) Half-hour lists of out-of-service units.

(d) Repair data which consists of manually input diagnostic repair (pack replacement) data. This data is referred to as frame repair records.

B. Data Collection

6.141 The data which is routinely collected by error analysis programs falls into the following three main classes:

(a) Maintenance Level Data: Maintenance level data, includes data collected on maintenance interrupts, maintenance interjects, and also via base level maintenance reports. Maintenance interrupt data comes from the interrupt bins and consists mainly of the saved contents of internal central control registers. Such data is intended to provide information describing the general state of the processor at the time of an interrupt. Other maintenance level data comes from the fault recovery programs.

(b) Data collected on base level from diagnostic and deferred fault recovery programs: This data is collected in order to summarize the principal results obtained by running the diagnostic and deferred fault recognition programs—especially when they are run in response to a detected system trouble.

(c) Other data, including:

(1) Data from other maintenance programs such as peripheral error analysis in the application program

(2) Repair data (frame repair records) which is manually input for information purposes

(3) Data associated with phase histories, writable store audit failure report, traffic/plant measurement reports, and out-of-service unit reports.

C. Data Retrieval

6.142 The stored data is used in manual analysis. To provide information needed for manual analysis, retrieval routines are provided to search through the data base according to certain patterns of interest and according to numerical values of specified data elements. An example of the latter is a search based on the contents of saved central control operational registers. With the aid of multiple key (keyword) matching, maintenance personnel may request, for example, information retrieval in accordance with a pattern based on a combination of particular interrupt sources, system configuration, failing addresses, and unit type member number. Such searches have been made less difficult by the formation of brief descriptors of maintenance level files known as file descriptor blocks.

6.143 All forms of output which are intended for eventual maintenance personnel use are classified as reporting. Included in these outputs (reports) are:

(a) TTY output messages which give the desired information in response to a successful retrieval

(b) Error message for an invalid request

(c) Automatic output messages which indicate that the data base is (nearly) full

(d) Summaries of data which originated during any specified time period, containing information on the current usage of storage space.

6.144 Summaries, which only occur in response to manually input messages, provide assistance to maintenance personnel in making retrieval and editing requests. They may also be used in conjunction with follow-on retrieval requests to obtain an overall office profile or history for those maintenance troubles on which data has been collected.

7. SUPPORT DOCUMENTATION (USER-TYPE DOCUMENTS)

INTRODUCTION

7.01 This part is provided as an introductory guide to the support documentation that can be ordered for any Electronic Switching System. A brief description of the general content and uses of each type of document is given.

GENERIC PROGRAM DOCUMENTS

A. PR Documents

Program Listing (PR)

7.02 A program listing is a hard copy record of a program which should include a description of the objectives of the program, a list of the instructions used to accomplish those objectives, and definitions of all data items unique to the program. Each program consists of one or more subunits called pidents (program identifications). A pident is the smallest segment of a program (or group of instructions) which is assembled by the assembly program.

B. PK Documents

Test Access Documents (PK)

7.03 The Test Access documents provide the link between software and hardware. They are utilized when the software diagnostic and trouble-locating programs fail to isolate a problem, and manual troubleshooting must be performed. These documents identify the specific hardware monitor points associated with each diagnostic test.

7.04 Test Access documents are provided for each software diagnosable frame and are designated by the same number as the primary schematic drawing (SD) for the frame, prefixed with "PK". Each document contains a section providing descriptions and examples for reference.

7.05 The pages in these Test Access documents are arranged by mode and address. The diagnostic summary data message is used to access the correct page and bit number. Each of the 24 bits represents a different monitor point. The function name or software name is given in the first two columns and expanded, under Description, in the last column. The SD number gives the last two digits of the SD. The functional schematic/symbol numbers (FS/SYM) and lead designations then give a single point to commence troubleshooting. The name of the specific register or gate in the circuit pack schematic (CPS) is given for further information.

Raw Data Document (PK)

7.06 The PK program document specifies the type of diagnostic test performed and its expected, unprocessed, raw data as they are initially stored in memory. This document is associated with one of several maintenance programs which diagnose specific equipment units.

7.07 The PK supports the TLM for those cases where the trouble number cannot be found in the TLM or where the replacement of the equipment listed in the TLM does not correct the trouble. This may occasionally happen particularly when the fault is marginal in nature.

7.08 In the event that the trouble number does not lead to the trouble, the attendant can request, via the maintenance TTY, that the diagnostic program be reexecuted and that the test results be printed in an unprocessed form. For trunks, raw data is requested via a diagnostic from the trunk and line test panel. The raw data document helps the attendant to interpret this test data by describing the various tests and their expected results.

♦ General Call Processing Related Document (PK)

7.09 There are three general call processing related documents:

(a) CIC manual (PKxA123): This manual contains a series of Compool items, macros, and pidents that are used to load orders into peripheral order buffers.

(b) CIN manual (PKxA121): This manual describes the function and use of CIN macros and characterizes the network programs.

(c) USER's Manual (PKxA120): This manual provides input and output specifications for routines making up translation programs.

Note: The "x" represents a 1 (for No. 1 ESS switch) or a 6 (for No. 1A ESS switch).◀

C. IM Documents

Input Message Manual (IM)

7.10 The Input Message Manual lists TTY messages that can be typed on the maintenance TTYs to request a system action or function. A description of the format and the use of each message, as well as cautions and expected results, are given for each message. The messages are arranged in alphanumerical order, and a topical index guides the reader to the specific message to be used.

D. OM Documents

Output Message Manual (OM)

7.11 The Output Message Manual lists in alphanumeric order all the system output messages printed by the TTY. This document contains a description of each message, the reason each message was issued, the actions to be taken, if any, as a result of the message having been issued, and alarm indications that should accompany the message.

E. TG-1A Documents

Translation Guide (TG)

7.12 The TG provides complete documentation of the software (translations) interface between, the telephone company assignment requirements for lines, trunks, routing, charging, measurements, etc. and, the Western Electric Company computer input requirements. Also, the document details the relationship of these input requirements to the actual feature, option or machine action desired, and the affect of the computer processes on the telephone company's administrative records maintained for the office.

F. ▶PG-1A Documents

Parameter Guide (PG)

7.13 The PG-1A is used in the preparation of input data for the parameter data assembler (PDA). Its functional scope for parameter data is analogous to that of the Translation Guide (TG-1A) for translation data. The scope of the PG-1A includes almost all information covered in the current PA-6A001, Volume 1. The actual layout of parameter data in unduplicated call store is not within the scope of the PG-1A; this is left to the PA-6A002. The PG-1A is not designed for manual engineering of call store.◀

G. TLM Documents

Trouble Locating Manual (TLM)

7.14 The TLM is a maintenance document which supplements the Output Message Manual to help in locating troubles within system units. A TLM usually covers one functional unit of the system (for example, program store, call store, etc). The TLM lists trouble numbers that are matched with numbers generated by the system from the diagnostic results. The suspected faulty package(s) (location and type) and any special procedure are specified adjacent to each trouble number. Except for TLM-1A001 on trunks and TLM-1A121 on TTYs, a TLM carries the same number as the SD of the functional unit with which it is associated. A few system units do not have an associated TLM.

APPLICATION DOCUMENTS—1A PROCESSOR

A. Diagnostic Program Applications—Description

7.15 This document describes for personnel in a telephone company switching office equipped with a 1A processor, the diagnostic program applications in terms of:

- Frame control switch requests
- Common TTY messages and options
- Interactive diagnostic options
- Diagnostic abort evaluation.

B. Generic Utility Program Applications—Description

7.16 This document describes the application of the GULP to the 1A processor. The information provided orients the craftsperson to the 1A processor GULP in areas not discussed in other documents.

7.17 The first part of the document is a high level discussion of the purpose, capabilities, and functions of the program. Part 2 describes the utility function verbs and provides examples to clarify the explanations. Part 3 consists of the binary layouts of the data tables and instructions used in GULP.

C. Error Analysis Program Applications—Description

7.18 This document provides information to assist telephone company personnel in the application of the 1A processor error analysis program (ERAP) and error analysis library program (ERLI). It includes a discussion of the following:

- Purpose of ERAP and ERLI
- Function and organization of ERAP
- ERAP input/output messages
- Functions of ERLI
- Preparations for running ERLI
- ERLI input/output messages
- ERAP procedures.

D. Program Listing—Software Description

7.19 This document describes the information (format and layout) contained in a program listing (PR) that is used in the Telephone Company switching offices equipped with a No. 1 or 1A processor. This section also describes the information contained in both a standard listing and a diagnostic phase program listing. The diagnostic phase program listing is different in use from other PRs and is therefore covered separately in this section.

7.20 A program listing is a software-generated hard-copy record, an output of the switching assembly program (SWAP). This section provides a discussion of this hard-copy record that contains information on the following basic topics:

- Program listing format
- Line format of instructions
- Diagnostic phase program listing
- Definitions of terms (GLOSSARY).

E. Diagnostic Language (DL-1)—Software Description

7.21 This document describes the diagnostic language (DL-1) and provides the following:

(a) Description of the basic structure of DL-1

(b) Description of statement format and definition of terms

(c) Detailed explanation of each DL-1 statement

(d) Alphabetical listing of the DL-1 statements.

8. GLOSSARY

8.01 A glossary of terms used in this section is described below:

Base Level (L-Level): The operational level in which the central control performs the majority of its work. All call processing is done on base level.

Buffer: A general purpose call store memory area used to store data when necessary to compensate for a difference in data flow rate.

Busy/Idle Bit: One of 16 bits in a call store word which denotes the status of an item (eg, a link in the network map). Normally, a busy bit is equal to zero and a idle is equal to one.

Busy/Idle Word: A word in call store which contains 16 activity bits (bit positions 6 through 21) corresponding to 16 different A, B, or C links or junctors.

Client: A program that is currently requesting service from other programs or routines.

Flag: Usually a bit that, when set, indicates a request for service.

Generate Control Pulse: An instruction used to generate direct pulses to various points in the system and provide for possible responses.

Generic Program: The program controlling all system operations including diagnostic and maintenance activities.

Global: A common address to which many pidents transfer.

H-Level: High priority J-level work.

Hopper: An area of call store to store information being referred from an input/output program to a call processing program.

J-Level: An interrupt level at which clock controlled input/output programs are executed.

K-Code: A numerical definition of the address limits of a particular store.

Line Equipment Number: A number which uniquely identifies an appearance on a line switch frame.

Link Map: An area of call store containing busy/idle status of links and junctors in the network.

Local: An address which is only available from locations defined in the same pident.

Macro: A high level statement that the assembly program interprets and expands into a predefined sequence of instructions or data.

Multiline Hunt Group: A group of lines that provide the means to supply a set of special originating and terminating services on a group basis rather than an individual basis. In addition to these services, line hunting is also provided.

Multi-MAC: Refers to the ability of MACP to run concurrent jobs.

Page: One or more file store resident program sections, each of which is functionally complete, including the subroutines called by the program sections.

Paging: The operations required to bring a paged program from file store into core memory before execution can begin.

Path Memory: A part of temporary memory where enough information about a connection is stored to enable the system to reconstruct the connection.

Queue: A call store memory area used to record a waiting list of work which temporarily cannot be completed.

Register: A call store memory area used to store information required to process a particular call in progress or to record administrative or maintenance information.

Scratch Pad: A memory area allocated to the program for temporary data storage.

Task Dispenser: A program which unloads an assigned buffer and, for each buffer entry, transfers control to another program until the buffer is empty. Control is returned to the task dispenser after each buffer entry is processed.

Task Program: A program called in by a task dispenser to process a single entry in a hopper, queue, or other buffer.

T1: An activity memory bit associated with each ferrod to indicate the state of the ferrod.

T2: A control bit associated with each T1 activity bit which indicates whether the state of the T1 bit should be reported when a change occurs.

Volume Controlled Calls: Certain types of calls which are limited in number by the software at any instant of time in a particular office.

9. ABBREVIATIONS AND ACRONYMS

9.01 The following is a defined list of abbreviations and acronyms used in this section.

ADS	Auxiliary Data System
AEX	Automatic Routine Exercise
AIFR	Fault Recognition Program
AIOD	Automatic Identified Outward Dialing
ALIT	Automatic Line Insulation Test
AMA	Automatic Message Accounting
AMAC	AMA Data Accumulation Program
AMDX	AMA Data Transfer Program
ANI	Automatic Number Identification
AOVD	Automatic Overload Control Program
APCL	Attached Processor Communication Link

API	Attached Processor Interface
APMH	Attached Processor Message Handler
APS	Attached Processor System
APT	Automatic Progression Testing
ASW	All Seems Well
ATAL	Audible, Disconnect, and Line Termination
ATTT	Automatic Trunk Test Termination
AU	Auxiliary Unit
AUBSQ	Auxiliary Unit Bus Sequence
BCD	Binary Coded Decimal
BINK	1024 words (BINary one K)
CC	Central Control
CCAD	Customer Changeable Speed Calling Program
CCIS	Common Channel Interoffice Signaling
CCOL	Chart Column
CFUP	Call Forwarding Usage Program
CHRN	Channel Request Number
CIC	Change in Circuit
CIN	Change in Network
CLID	Calling Line Identification List
CMB	Channel Memory Block
CNC	Coin Charge Register
CNLP	Centrex Console Lamp Control Program
COCN	Coin Control Program
COIN	Coin Charge Program
COPR	Report and Miscellaneous Subroutines
CPD	Central Pulse Distributor
CPDB	Central Pulse Distributor Enable Address
CPFR	Central Pulse Distributor Fault
CR	Call Register
CRFI	Common System Recorded Announcement
CS	Call Store Frame Input Analysis Program
CSDS	Circuit Switched Digital Capability
CSRAF	Common System Recorded Announcement Frame
CTYP	Call Type
CX1X	Centrex Tandem Tie Line Program
CXBV	Busy Verify-Trunk Test Program
CXDS	Disconnect for Centrex Program
CXIC	Incoming Digit Analysis for Centrex
CXIO	Centrex Input/Output Scan Program
CXKY	Centrex Key Signal Director
CXLO	Centrex Attendant Line andTrunk Seizure
CXOR	Centrex Digit Analysis Program
CXSF	Centrex Simulated Facilities Program
CXTA	Centrex Trunk Code-Call Answer Program
CXTP	Centrex Trunk Preemption Program

CXYH	Seize and Release Routines
CZO	Coin Zone Operator
DAC	Design Aid Computerized
DCON	Diagnostic Control Program
DCONMAIN	Diagnostic Control Program (pident)
DCS	Duplicated Call Store
DCT	Digital Carrier Trunk
DDD	Direct Distance Dialing
DIAG	Diagnostic
DISC	Disconnect Program
DKAD	Disk Administration Program
DKADI	Disk Administration Interface
DMA	Direct Memory Access
DMAPAPPL	Data Mapping Control and Linking Program
DMERT	Duplex Multi-Environment Real-Time
DOC	Dynamic Overload Control
DOCT	Dictionary Trouble Number Program
DP	Dial Pulse
DRE	Directional Reservation of Equipment
DRPP	Diagnostic Results Post-Processing
DSP	Dynamic Service Protection
DTST	Dial Tone Speed Test Program
DU	Data Unit
DUAD	Data Unit Administration Program
DUC	Data Unit Controller
DUS	Data Unit Selector
ECIO	Executive Control Input/Output Program
ECMP	Executive Control Main Program
EMERFULL	Emergency Mode Control Full Configuration
EMERMIN	Emergency Mode Control Minimum
EML	Emergency Manual Line
ERAP	1A Processor Error Analysis Program
ERLI	Error Analysis Library Program
FDIP	Frame Dependent Interface Program
FM	File Manager
FMI	File Manage Interface
FOR	Fault Recognition
FS	File Store
FSAP	File Store Administration Answer
FSC	File Store Controller
FSSP	File Store Administration Submit Program
FSSR	File Store Service Routine
GCP	Generate Control Pulse
GRC	Growth Recent Change
GULP	Generic Utility Program
HMTL	Hotel-Motel Program
HUC	Higher Unduplicated Call Store
ICB	Input Character Buffer
IM	Input Manual

I/O	Input-Output
IOCP	Input/Output Control Program
IOT	Intraoffice Trunk
IOU	Input/Output Unit
IOUC	Input/Output Unit Controller
IOUS	Input/Output Unit Selector
ITTT	Incoming Trunk Test Termination
KB/S	Kilobits per Second
L-L	Line to Line
L-T	Line to Trunk
LDR	Loader
LENCL-4	Line Equipment Number Class 4
LIBR	Library Control Program
LIBRTRP1	Library Control Common Traps Administrator
LIFO	Last in First Out
LLN	Line Link Network
LUC	Lower Unduplicated Call Store
LULPUTIL	Local Generic Utility Program
MAACA	No. 1A ESSS Scheduler
MAC	Maintenance Control
MACP	Maintenance Control Program
MACR	Maintenance Control Peripheral Program
MALM	System Alarm Program
MAUD	Maintenance Audit Program
MCC	Master Control Console
MCCM	Common Control and Monitor Program
MCCP	Maintenance Control Center Program
MCLM	System Alarms Program
MCTWADMN	Master Control Center Administration
MFJR	Multifrequency Signaling Junior Register
MTS	Message Telecommunications Service
MURL	Maintenance Unexpected Results List
NCD	Noncheck Dummy
NETG	Network Growth Program
NMFA	Network Fabric Routines Program
NMFL	Network Maintenance Action Program
NMIN	Network Management Indicator Program
NMMP	Network Management Maintenance Program
NMMX	Network Matrix Exercise Program
NMRF	Network Fault Recognition Program
NMTD	Transmit Dynamic Overload Control Signals
NMTG	Network Management Program
NTWK	Network Program
OFGT	Miscellaneous Outgoing to Switchboards and Desk Program
OFML	Emergency Manual Line Service Program
OFNT	Operator No Test Program
OFTR	Toll Switch and Recording Completing

OM	Output Message
OMO	Overtime Monitoring Operator
OMR	Output Message Register
OOS	Out of Service
OPCL	Outpulsing Control Register
OR	Originating Register
PAGS	Paging Supervision Program
PATT	Processor Application Transfer Table
PBX	Private Branch Exchange
PC	Processor Configuration
PDA	Parameter Data Assembler
PG	Generic Program Documentation Index
PGID	Generic identification and Compatibility
PLUG	Line Termination Denied Program
POB	Peripheral Order Buffer
PR	Program Listing
PRE	Protected Reservation of Equipment
PS	Program Store
PSDC	Public Switched Digital Capability
PT	Program Tag
PTW	Primary Translation Word
PUAB	Peripheral Unit Address Bus
QAPR	Queue and Administration Processing
QCIA	Customer Interface and Special Auditing
QSIF	Queue State Information Feature
QTAL	Give Audible, Disconnect, and Line
RACT	Relay Activity Bit
RADR	Receiver Attachment Delay Report
RAF	Recorded Announcement Frame
RAM	Random Access Memory (Read-Write)
RAMP	Recorded Announcement Machine Program
RBB	Rollback Block
RC	Recent Change
RCSS	Recent Change Subsystem
REX	Routine Exercise
RI	Register Identification
RI-PT	Register Identification-Program Tag
ROH	Receiver Off-hook
RPPS	Regional Parameter Processing System
RRT	Routine Request Table
RVRT	Reverting Call Program
SACT	Customer Program for Growth
SADT	System Audit Program
SAWS	Writable Store Audits
SCFR	Scanner Fault Recognition Program
SI	System Initialization
SIRE	System Interrupt Recovery Program
SR	System Reinitialization

SRTT	Station Ringer and TOUCH-TONE Program
SUPERV	Supervision Modernization
SWAP	Switching Assembly Program
SYPI	System Performance Indicator Program
SYUP	System Update Program
T-T	Trunk to Trunk
TAND	Tandem Connection Program
TBR	Teletypewriter Buffer Register
TBTF	Through Balance Test Facility
TCC	Trunk Class Code
TG	Translation Guide
TGC	Trunk Group Control
TLM	Trouble Locating Manual
TLN	Trunk Line Network
TODA	Ringing and Tone Plant Diagnostic
TOMK	Ringing and Tone Plant Monitor
TOPR	Toll Operator Signaling Program
TRCE	Call Trace Program
TRNS	Transition State
TSAH	Trunk Seizure and Answer Hopper
TSPS	Traffic Service Position System
TTIA	Teletypewriter Input Messages Directory
TTOX	Teletypewriter Output Program
TTPP	Teletypewriter Output Phases Program
TTWK	Teletypewriter Work Register Program
TTYM	Teletypewriter Translation Input/Output
TUC	Tape Unit Controller
TVNDX	Transfer Vector Index
TWR	Teletypewriter Work Register
TXFR	Call Forwarding Program
VFHC	Verification of H and C Register
WAIT	Call Waiting Program
WPADAPL2	Write Protect Administration Application
WPADCOMM	Write Protect Administration Common
WPADCTRL	Write Protect Administration Control
WQUE	Queue Administration Program
WRDN	Word Number
YAHA	Seize and Release Routines
YCCK	Register Link Routine
YCLK	Register Linking Routine
YFDS	Scan of Single Master Scanner Point
YFTO	Incoming Trunk to Busy Overflow
YMRG	Miscellaneous Register Subroutines
YTTO	Originating Line to Busy Overflow
ZERO	Call Store Zeroing Program