No. 1 ESS ADF:

Message Processing Program Organization

By H. G. KIENZLE, K. L. NICODEMUS, M. T. SMITH JR., E. W. WEBER and H. M. ZYDNEY

(Manuscript received August 5, 1970)

The No. 1 Electronic Switching System, Arranged with Data Features (No. 1 ESS ADF), uses the basic program organization of No. 1 ESS with additional task programs, input-output equipment, and mass storage to provide a reliable switching arrangement for store-and-forward data service. This paper describes the operational program organization used to control the reception of a message into the system from an originating data station, its placement in the message store, and its delivery to the terminating stations. In addition, it describes the service features and the unique characteristics of the system's translation and message-integrity features.

I. INTRODUCTION

The No. 1 Electronic Switching System, Arranged with Data Features (No. 1 ESS ADF), is a stored-program system which uses the basic No. 1 ESS program organization with additional task programs, input-output equipment, and mass storage to provide a reliable switching arrangement for store-and-forward data service.

To meet the needs of data switching, No. 1 ESS ADF must meet the stringent operating requirements of a telephone switching office[1] in addition to meeting requirements that are unique to a data-switching office. The system must (*i*) respond rapidly in real time to the demands for service, (*ii*) have the versatility to provide the large variety of service features demanded of a modern data-communication system, (*iii*) be reliable, (*iv*) be capable of meeting the needs of different installations without modification of the basic program, and (*v*) be economically balanced in its use of real-time and storage facilities. In addition, the No. 1 ESS ADF must meet requirements beyond those imposed on telephone switching systems.[2] Unlike line-switched systems, a message-switched system provides no end-to-end verification of the communication path or delivery of the message. The originating user is completely reliant on the system to properly deliver accepted messages to the appropriate terminators. This imposes stringent requirements on the message-switching system to provide message protection, assurance of delivery, and privacy.

In a message-switched system, communication between incompatible station arrangements is possible since the originating and terminating stations are never directly connected. Only the switching center need be compatible with all station arrangements. Thus, the service offering includes many station arrangements: (*i*) send only, receive only, and send-receive; (*ii*) single and multistation lines; (*iii*) multiline hunting groups; (*iv*) a variety of transmission rates; (*v*) a variety of information codes; and (*vi*) a variety of signaling sequences.

Other important features which are desired by users are multiple-address operation, the use of mnemonic codes and group codes for addressing, message numbering, time and date insertion, and message retrieval. A complete list of features is given in Ref. 2. The extensive set of service features offered by this system is made economically possible by employing the stored-program concept of system control.

A data message passes through three basic stages of operational processing: (*i*) detection of a request for service, (*ii*) processing of the input transmission along with storage of the message and the formation of a delivery queue for each addressee, and (*iii*) servicing of the delivery queues and transmission of each message to all terminators.

In the following sections of this paper a description of the operational program is given through both a general and a detailed account of the processing of a typical data message. While the system is capable of handling both five-level and eight-level stations, the scope of this paper will be restricted to eight-level operation since this represents the more modern station arrangements.

II. SYSTEM OPERATION

To utilize economically the transmission facilities, each line may service up to twenty eight-level stations as indicated in Fig. 1. The use of multistation lines requires that the switching center periodically interrogate or poll all stations to determine their desire to originate traffic. The polling is accomplished over the transmission facilities by an interchange of a sequence of characters within the code set of the data machines. Each station on a given line is assigned a specific station poll code to which it alone responds. The station responses

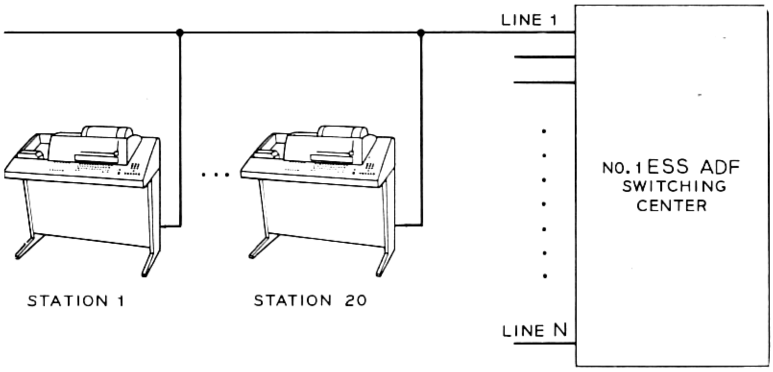


Fig. 1—The No. 1 ESS ADF system provides service to multistation lines.

which are tabulated in Table I contain information in addition to an indication of the desire to originate traffic. First, the importance of the originating traffic is indicated as priority or regular traffic. This allows for a per line priority option which involves the selective pickup of originating traffic on a per message basis. Priority options in prior switching systems have been limited to selectivity on a station basis. On any line which exercises this option, the switching center will pick up all priority traffic before accepting any regular traffic. Second, the status of the receive portion of the station is indicated. If the station receiver is out of service or made busy, the polling response indicates a not-ready-to-receive condition. This allows the switching center to hold traffic for the given station and concentrate on deliveries to other stations on the line.

On half-duplex (HDX) lines, the line can be polled continuously whenever it is neither transmitting nor receiving traffic. However, on full-duplex (FDX) lines, two independent communication channels are available for concurrent transmission and reception of traffic. Polling of FDX lines is conducted by temporarily halting any outgoing traffic on the line and sending the appropriate poll code for each station. Continuous polling of the FDX lines would interfere with message delivery. Thus, when all stations on the line have no traffic to send, polling is suspended and all stations are placed in a "suspend poll" state. If a user mounts a message tape on the station and bids for service, a polling request code is generated by the station and is recognized by the switching center which initiates a round of polling for the line. The suspend-poll mode is backed up by a

TABLE I—EIGHT-LEVEL STATION POLLING RESPONSES

Station Response	Input Traffic Available	Status of Receiver
CAN	None	Ready
NAK	None	Not Ready
P ACK	Priority	Ready
P NAK	Priority	Not Ready
R ACK	Regular	Ready
R NAK	Regular	Not Ready

time-out mechanism which insures the polling of all FDX lines at least once every 15 minutes.

A user who wishes to originate a data message prepares a paper tape in a standard format consisting of two parts, heading and text. These parts are delineated by special characters for start-of-heading, start-of-text, and end-of-text. Several messages may be cut on a single tape and the last message is followed by a end-of-transmission character. The heading is subdivided into fields for routing information, delivery priority, and personal address information and must follow specific format rules. If the format rules are violated, the originated message will not be accepted by the processor. A service message is sent to the originator indicating a control-code error or a heading format error with the erroneous heading field identified.

Having prepared the message, the user loads the message into his machine and conditions the machine to indicate that it has traffic. When the switching center detects the desire to originate traffic as a result of the polling response, it consults its translation information to obtain a description of the data machine, code and speed, and the service features subscribed to by the user for this machine. Based on the service features chosen by the user, the switching center generates and transmits to the station a printed record which typically consists of the time and date of message pickup and a serial number assigned to the message which the user can use in referring to this message. The last part of the printed record is a sequence of characters which initiates transmission of the message to the switching center. The switching center assembles the characters into blocks of memory and builds a single copy of the message on magnetic disk. As soon as the complete heading is received and properly stored, the switching center proceeds to analyze it. In so doing, it generates a centralized descriptive record of the message called the message-processing block. This record contains the identity of the originator, message descriptive information, a list of the recipients, and an indication of where the message has been stored on magnetic disk. Upon completion of the analysis of the heading, an entry consisting of the location of the message-processing block is made in each of the terminators' message-delivery queues. Each line has a separate queue in which messages are ordered by priority and time of arrival within a priority class. Deliveries to the recipients may now be initiated simultaneously with, yet independent of, the input transmission. When the complete message has been received from the input line, the switching center continues to monitor the line for receipt of either a start-of-message character indicating a multimessage transmission or an end-of-transmission character which restores the station to the idle condition.

Delivery to the stations is initiated as a result of scanning the message-delivery queues. If the line associated with a given queue entry is idle and the station is ready to receive as indicated by the polling responses, a call-in sequence of characters is transmitted to the line and conditions the desired station for reception of the message. One of these characters, the call enquiry code, is unique to each station on the line. To insure that the proper station receives the message, the station responds with a unique station identification code which is verified with translation data. In addition, the station also responds with an indication of its readiness to receive. The switching center then generates and transmits a heading to the station based on the options chosen by the user. This typically consists of the mnemonic used in the heading to address the station (this helps the station attendant to identify the message recipient at a communication center which may be addressed by many mnemonics), the time and date of delivery, a message number assigned by the switching center to the transmission, and the originating time, date and message number. The originator's heading, at the option of the user, is transmitted to the station, followed by the text of the message as derived from the copy stored on magnetic disk. The heading and text are code converted if the originating code differs from that of the terminating machine. Upon completion of the transmission, the switching center and the station engage in an exchange of signaling called roll-call to verify the proper delivery of the message. The switching center again transmits the call-enquiry code to which the station responds with its station identification code and the present status of its readiness to receive. If the station fails the roll-call checks, the message is requeued for future delivery. During the call-in and roll-call signaling sequences, input traffic on FDX lines must be halted so that the input channel can be used for responses from the station. The above discussion is directed to delivery to a single station; however, if the message is addressed to more than one station on a given line, the message is directed to all such stations in a single transmission. The call-in and roll-call sequences are expanded and the generated heading is directed to each station individually as appropriate.

On HDX lines intraline delivery capability of messages is provided. For this type of operation, the originating station halts its transmission upon completion of the heading. The switching center then analyzes the heading and restarts the station if no intraline deliveries are involved. If the message being received is addressed to stations on the same line as the originator, the addressed stations are called in, receive their generated heading and receive optionally the originator's heading. The switching center then restarts the originator's station, and the text is simultaneously transmitted to both the switching center and the receiving stations. The receiving stations are roll-called upon completion of the message.

The control logic provided by the switching center is further complicated by the need and desire of user groups to control and manage the traffic on their networks. The user may designate a particular station or stations as a control location. This control location has the capability of placing stations on skip, denying them origination capability, and on hold, denying delivery capability, or traffic may be rerouted to an alternate station. This requires that the polling process examine the skip status of all stations and send out polling codes only to those stations which are not on skip. It also requires the delivery process to examine the hold state of each station and skip over message queue entries for stations in a hold state.

The somewhat complex dialog required between the switching center and the stations it serves provides a high degree of integrity in handling messages. Any anomalies detected by the almost continuous monitoring of service are reported to transmission-plant craftsmen through service messages. This allows for rapid maintenance of the transmission plant which does not depend on user complaints to detect troubles as prior systems have done.

III. SOFTWARE STRUCTURE

3.1 *Task Schedule*

The organization of the No. 1 ESS ADF program was strongly influenced by the desire to maintain compatibility with the No. 1 ESS program to permit the possibility of future merger of these programs in a combined voice-and-data system. In designing the executive control program, the scheduling algorithm described by J. A. Harr, et al.,[1] has been used with only minor modifications.

The real-time tasks or stimuli processed by the switcher vary largely in their requirements for response time. The rapid response time necessary for the more critical tasks is achieved through an interrupt mechanism. The central control clock interrupts the base-level programs periodically and transfers control of the system momentarily to the input-output main program (J-level) which systematically searches for high-priority work. The majority of tasks, including all operational input-output tasks, can be handled with a 10-ms interrupt. Only a few infrequent maintenance tasks require a 5-ms scanning period during the interval that the maintenance task is active. The interrupt mechanism imposes a real-time overhead on the processor. First, the central control registers must be stored before performing the input-output tasks and then restored before reentry to the interrupted base-level program. Second, the process of scanning for work involves an administrative overhead, especially if done at a frequency greater than required. To minimize this overhead associated with the interrupt work, the standard 5-ms clock interrupts are subdivided by program means into odd and even interrupts. Only on odd interrupts, a 10-ms period, is control transferred to the input-output main program. Even interrupts return control to the base-level program that was interrupted immediately without the necessity of storing and restoring the central control registers.

The base-level main program scheduling algorithm is basically cyclic and nonsynchronous. All processing tasks are assigned to one of five main-program levels. These levels are executed in a prearranged order so that each level is scheduled twice as frequently as the next lower level. Each task is either permanently active or activated on request and is assigned to a given main-program level based on the real-time requirements of the task. One additional level exists, called interject, which allows for the insertion and execution of a high-priority task between any main program task. This algorithm tends to favor the real-time tasks of higher priority while insuring that no task is deferred indefinitely.

3.2 *Processing Memory*

The processing associated with a message transmission consists of a number of disjoint tasks or processing segments. The action taken by the program in response to any given stimulus is a function of the history of all preceding actions. The program uses temporarily assigned memory areas, called processing registers, to retain the state of the transmission and associated data during the intervals between processing segments. A register is temporarily assigned to a data line when a transmission is set up, remains associated with the line while the line is active, and is released soon after the transmission is terminated. The register assigned to a message origination from a data machine is called the input-processing register, and the register used for message delivery to a data machine is called the output-processing register. To provide for the simultaneous delivery of the same message to more than one station on a given line, additional units of memory must be assigned to the line to store the data associated with each station. These memory areas, called processing-register annexes, are linked to the output-processing register as needed. The formats of the input-processing register and output-processing register are shown in Fig. 2.

The first word of each register contains the program tag defining the state of the processing. This tag is a relative address defining the program routine which is expected to process the next stimulus. If abnormal responses occur or a time-out of the expected response occurs, the program tag is indexed to produce the address of the particular routines designed to process these conditions. At the conclusion of the processing segment, the program tag is updated to reflect the next expected response.

A single central record of each message must be maintained by the switching center. This record contains the identities of the originator,

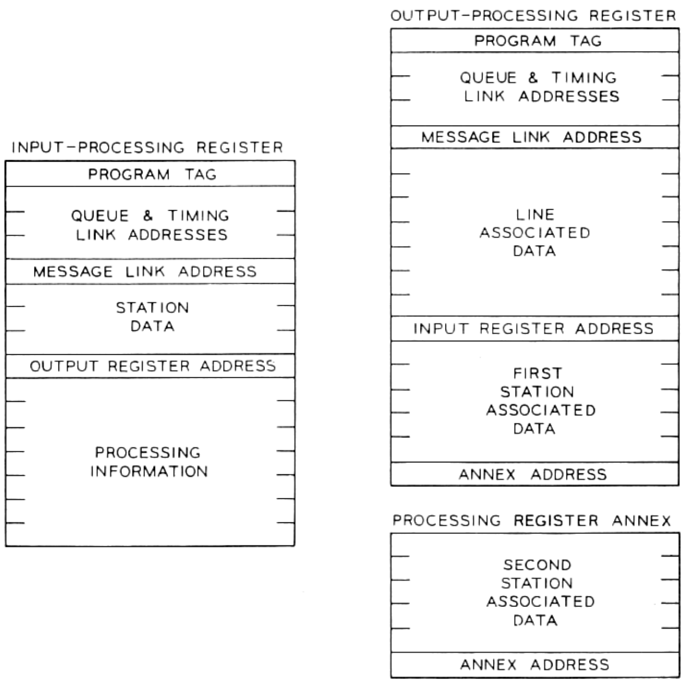

OUTPUT-PROCESSING REGISTER
PROGRAM TAG
QUEUE & TIMING LINK ADDRESSES
MESSAGE LINK ADDRESS
LINE ASSOCIATED DATA
INPUT REGISTER ADDRESS
FIRST STATION ASSOCIATED DATA
ANNEX ADDRESS
INPUT-PROCESSING REGISTER
PROGRAM TAG
QUEUE & TIMING LINK ADDRESSES
MESSAGE LINK ADDRESS
STATION DATA
OUTPUT REGISTER ADDRESS
PROCESSING INFORMATION
PROCESSING REGISTER ANNEX
SECOND STATION ASSOCIATED DATA
ANNEX ADDRESS

Fig. 2—Input- and output-processing register formats.

the message, and each terminator as well as the final disposition of each delivery. Since the number of addresses in a multiple-address message may be very large, perhaps several hundred, this central record is composed of assignable blocks of memory called message-processing blocks which are linked together in the required number. During transmission of the message, the associated processing registers are linked to the message-processing block as required. During periods of time when no active transmission is occurring for a given message, i.e., the message is merely queued for delivery, the message-processing blocks are stored on the disk memory to minimize the use of temporary memory. The format of the message-processing block is shown in Fig. 3.

During intervals of time when no active transmission is being processed for a given line, the line is constantly monitored by the polling process. A small, one-word register is used during these monitoring intervals. These registers are organized in a dedicated table, called

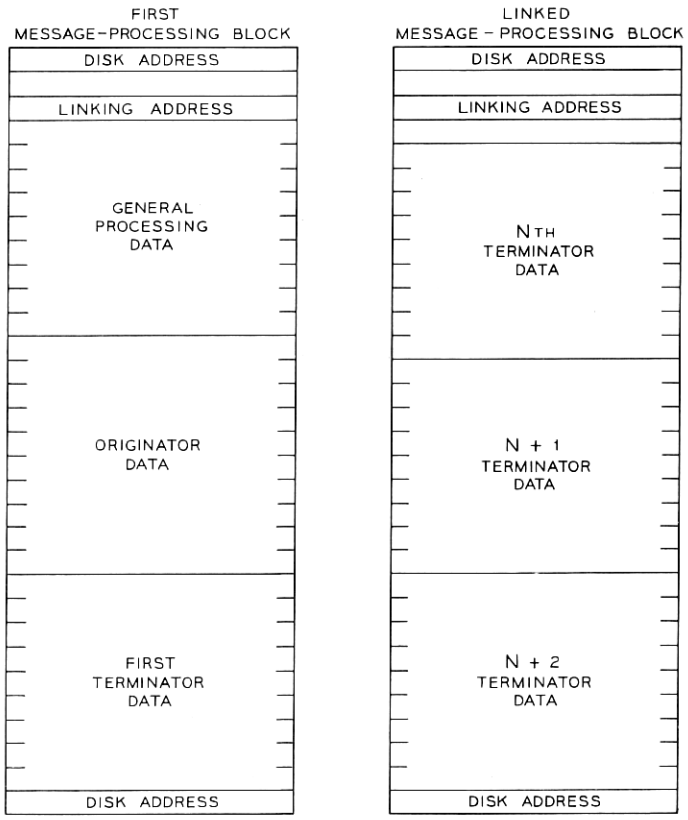


Fig. 3—Message-processing block format.

the line-status table, and indexed by the data-line number (DLN). Each word contains a line-status tag, a message waiting field which indicates the presence and highest precedence of messages awaiting output, and a polling field which is utilized by the polling process. The line-status tag is used in a manner similar to the program tag in the processing registers as a relative address to the program routine expected to process the next response from the line. The line-status table also provides the means for systematically associating the processing registers with a given line. In this case, the polling field is used to store the processing register address. The state of the line-status tag indicates the presence of a processing register so that the proper routine for processing a response can be determined from the program tag rather than from the line-status tag. The linking of the various areas of processing memory is illustrated in Fig. 4.

3.3 *Program Classification*

The programs associated with message processing may be classified in three broad categories. First, the programs which interface with the hardware or the outside world are called input-output programs. Second, the processing programs are the high-level programs for state control which advance a given process from stage to stage. To minimize the interfaces between the program designers, only three such programs were defined, one each for (*i*) polling, (*ii*) input transmission, and (*iii*) output transmission. However, to maintain programs of a manageable size, a number of second-level processing programs were defined to perform specific functions, which were large in nature but could be characterized with a minimum of interface. Third, service routines perform functions of a general nature for all processing programs and are characterized by the memory areas which they control.

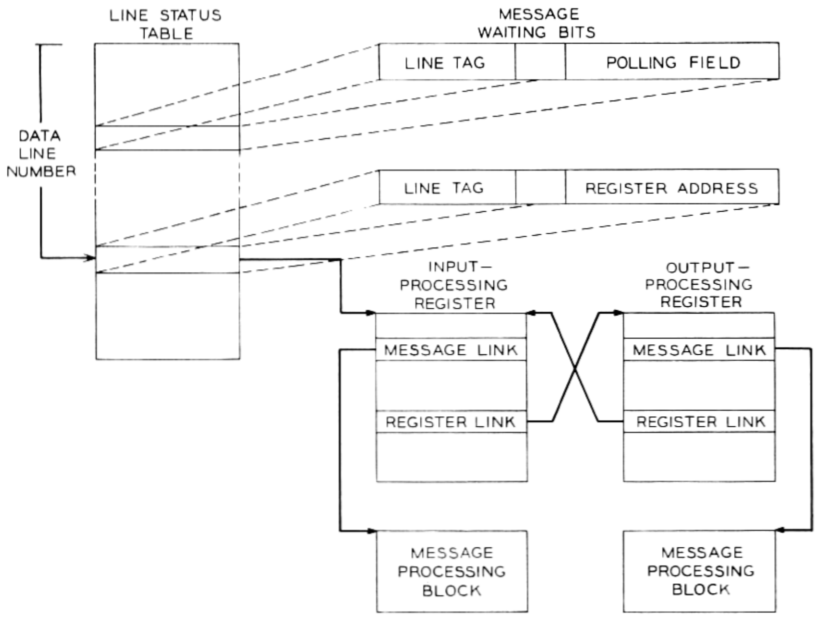


Fig. 4—Linking of processing memory areas.

3.4 *Economy of Resources*

The major resources of an electronic switching system are its storage facilities and its real-time capability. A store-and-forward switching system places greater requirements on temporary memory than a line-switched system since holding time on facilities is much greater and all data must pass through the processor. A unit of temporary memory cannot be viewed as a hardware cost alone because the total available storage affects the system capacity in much the same way as real-time capability. In the design of the No. 1 ESS ADF, each trade off between real-time capability and storage had to be viewed from a need to achieve a balance which resulted in the maximum system capacity.

IV. HARDWARE-SOFTWARE INTERFACE

One of the basic objectives in the design of the No. 1 ESS ADF was to maximize the use of No. 1 ESS equipment. The resulting equipment configuration is shown in the functional block diagram of Fig. 5. The central control, program store, and call store are equipments used in common with the No. 1 ESS. The central control guided by the program stored in the program store directs the operation of the switching center using the call store as a temporary scratchpad.

The buffer control and its associated equipment form an input-output community specifically designed for the No. 1 ESS ADF. The buffer control is a wired-logic subordinate processor which relieves the

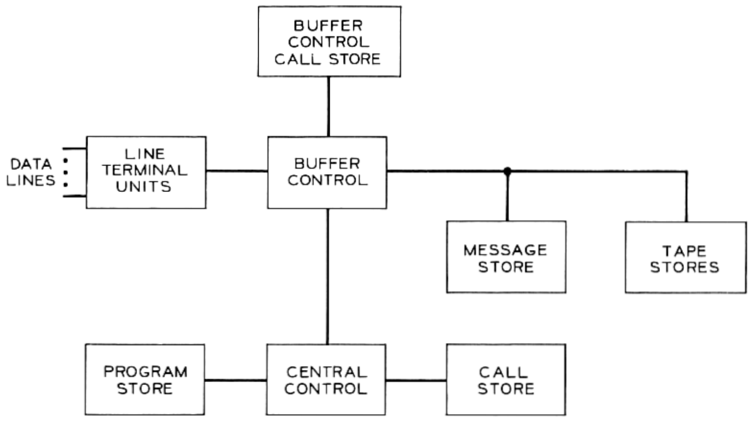


Fig. 5—Functional block diagram of No. 1 ESS ADF switching system.

central processor of many of the time-consuming taks associated with the input-output functions. The input-output community has a variable memory, called the buffer-control call store, which is used as a scratchpad in processing its own tasks and to communicate with the central control. Information destined for the central control is loaded in predetermined areas of the memory which are scanned and unloaded periodically by the central-control program. Information destined for the input-output equipment is loaded by the central-control program in other assigned areas of memory which are consulted and unloaded periodically by the wired-logic sequencers of the buffer control. Thus, this memory serves as the vehicle for the indirect interface between the input-output equipment and the central-processor program.

The program is primarily interested in communicating with three input-output communities: the data lines, the magnetic disk or message store, and the tape stores. The following sections will discuss the interfaces with the first two of these communities. The program interface with the tape system is discussed in another article in this issue.[3]

4.1 *Data Lines*

The memory-interface area for the data lines is shown in Fig. 6. Data entering the system from data lines are loaded into an area in the buffer control call store called the input-character hopper by the buffer control. This is a fixed-length common area used for all data lines. All entries in this area are time ordered and consist of two words. The first word contains the data-line number which identifies the source of the input data and a special action code. To relieve the program of the necessity of examining each character to detect signaling codes such as end-of-message, the buffer control recognizes all characters used for signaling and identifies these characters in the special-action code. The second word of the input-character hopper entry contains the three characters assembled from the line. All signaling characters produce an immediate entry in the input-character hopper. In this case, the first- and second-character positions in the second word may contain a fill character. Two pointers are maintained by the buffer control to assist in the loading and unloading of the input-character hopper. The load pointer indicates the next available entry slot. The unload pointer locates the next data entry to be unloaded by the program. The program addresses the

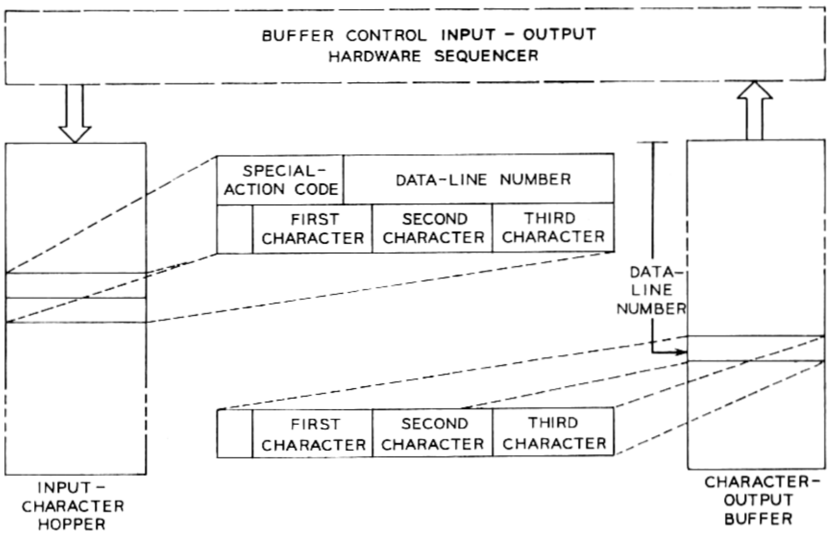


Fig. 6—Memory interface between program and data lines.

input-character hopper with a special entry address and is not cognizant of the absolute address of any given entry. This relieves the program of work associated with updating the pointers.

To transmit data to the lines, a memory area called the character-output buffer is used. This area consists of a list of words ordered by DLN, one word for each line connected to the switching center. Each word, serving as buffer storage for the associated line equipment, stores the next three characters to be transmitted on the data line. When a line equipment unit is ready to accept another set of characters for transmission, the buffer control unloads its associated word in the character-output buffer and inserts an idle code to indicate to the program that the unloading process is completed. The function of the program is to scan and load the character-output buffer at a sufficient rate to maintain the data-transmission rate on all lines that are actively transmitting.

4.2 *Message Store*

The memory interface area for the message store is illustrated in Fig. 7. To provide for greater throughput capability, the message store is organized into 16 sectors with the capability of transferring one data block of 32 words per sector per disk revolution. The message-store hardware executes the block transfers as a result of instructions in the instruction queue. In order to assist the program in maintaining the instructions in the proper order for block-transfer requests, the message-store instruction queue consists of 16 dedicated instruction slots, each slot dedicated to a particular sector. The message-store sequencer is designed to sense the physical position of the rotating disks and read the proper instruction slot. The task of the program is to follow the progress of the message-store hardware and keep the message-store instruction queue loaded with new instructions. The hardware maintains an unload pointer to indicate the next instruction to be executed. The program maintains a load pointer to indicate the next instruction slot to be loaded with a new instruction.

The format of an instruction slot is illustrated in Fig. 7. The operation code directs the action of the message-store hardware while the two addresses, the message-store address and the call-store address, specify the source and destination of the data to be transferred. Upon successfully executing an instruction, the message-store hardware overwrites the operation code with an idle code to notify the program.

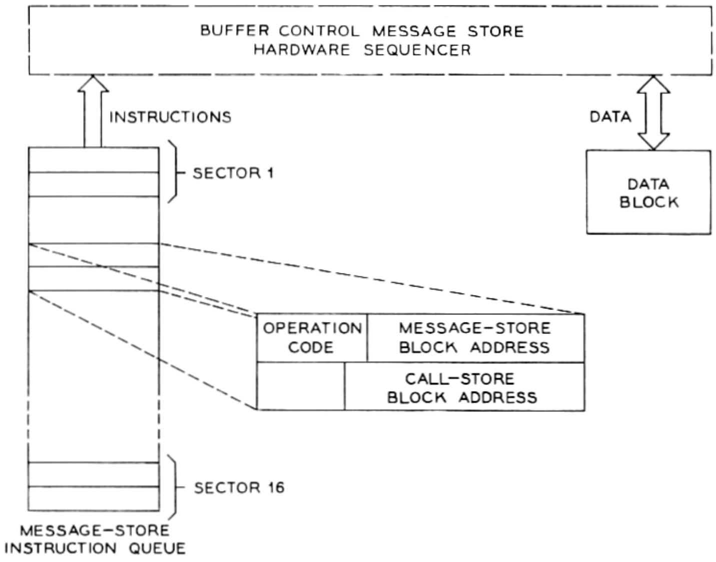


Fig. 7—Memory interface between program and message store.

V. INPUT-OUTPUT PROGRAMS

The programs which interface with the external environment are designed to be relatively simple but very efficient in their use of processor real time because of the highly repetitive nature of the functions. One is willing to sacrifice storage in the interests of maximizing the operating speed of these programs. The major input-output programs associated with the data lines and the message store are described in the following sections. Other programs, which detect carrier signals from data sets, detect maintenance states of hardware within the switching center, control-office status and alarm indicators, transmit messages to maintenance personnel, and interface with the tape stores,[3] are not covered.

5.1 *Line Input*

The input-sequence control program services the input-character hopper and does all the preliminary processing on the input data from lines. The primary function assumed by this program is the assembly of the data characters into blocks so that the message-processing programs can deal with the data in larger entities. In addition, some prefiltering of signaling characters is done before passing this information to the processing programs. The input-character hopper provides adequate buffer storage to allow this program to operate as a base-level program, thus minimizing the overhead. The input-sequence control program does not require communication buffering to the processing programs since all involved programs are operating at base level. However, time order must be maintained between the input and output stimuli handled by the processing programs. Since the output program operates on J-level, buffering of output reports is required. Thus to maintain time order between the reports, both the output program and the input-sequence control program communicate with the processing programs through a buffer called the data-service request hopper. Each entry in this hopper consists of two words, which are similar in format to the input-character hopper. The first word contains the DLN and a report-identification code. This code identifies the signaling characters in the report or processing signals such as the completion of a block of assembled data. The second word may contain the data received from the input-character hopper or a data-block address.

In processing the input data, the input-sequence control program must be cognizant of the state of the line and the next available location in the data block for the assembly of data. The word-status table provides the memory for this function. This table has one dedicated word per line indexed by the DLN. The special action indicator in the input-character hopper and the input state from the word-status table are used as a double index to determine the proper routine for processing each entry. The input state is a five-bit code of which 11 states are currently used:

(*i*) Two states are concerned with the assembly of heading and the assembly of text.

(*ii*) Two states are used to monitor the line after a start-up code is transmitted. The first three characters of heading or text are assembled in the data block, an entry is made in the data service request hopper indicating that the station has responded to the start-up code, and the line state is changed to the assemble heading or assemble-text state.

(*iii*) Two timing states are provided for the measurement of the marking interval after a FORM FEED or tab character has been received. The elapsed time is stored in the data block for use by the output program.

(*iv*) Two signal-monitoring states are used during signaling periods on the line. Both states are similar in that all received characters result in a data-service-request hopper entry. However, one state indicates that an input has been interrupted to use the input channel for station responses in which case the true-line state has been temporarily stored in the data block associated with the line.

(*v*) One state, that is peculiar to five-level stations, is provided for the assembly of heading with all information passed to the processing programs for immediate analysis of the heading information.

(*vi*) A loop-test state is provided for a loop-around test of the line-terminal hardware. All data received during this state are stored in a specified buffer area for analysis by a maintenance program.

(*vii*) A "no operation" state is used to temporarily place a line out of service and disregard all inputs. If an invalid character is received for a given state, the signal filtering provided by the input-sequence control program results in the input state being set to the no-operation state and a data-service-request hopper entry is made to initiate abort procedures on the line.

The process of assembling a three-character word into a data block represents the primary work load for the input-sequence control program and is designed as the main line thread of logic with the greatest efficiency. The characters are assembled into 30 words of a 32-word block. The first and last words are reserved for forward and reverse link addresses which link all blocks of a message together. The five least-significant bits of the data-block address indicate into which of the 32 words the assembly is to be made. After assembly, these five bits are tested to determine if the block is filled. If the end of block has not been reached, the data-block address is incremented and reinserted in the word-status table. If the last word of the block is used, a new data block is seized and its address is placed in the word-status table and the last word of the data block just assembled. An entry is made in the data-service-request hopper to notify the processing programs of the completed block. If the flow of input characters should halt, as a result of a stuck station transmitter or broken tape, the lack of stimulus and assembly of further characters would go undetected. To detect this condition, called intercharacter time-out, a timing bit is assigned to the word-status table. An executive control routine sets this timing bit every 32 seconds. The assembly process resets the bit upon updating the data-block address. If the executive control routine, during its periodic visitation, finds the bit set while the line is in an assembly state, an intercharacter time-out is declared and the originating user is informed through a service message.

The input-sequence control program performs a rather important function which tends to reduce the work load on the output program. Input data which requires special action by the output program, such as tabbing functions or end-of-text (ETX) characters, are specifically flagged so that the output program does not have to scan and test each character that is being transmitted in order to detect these functions. The flagging process results in a two-word assembly in the data block. The characters involving the special action are assembled in the first word and an output control code which defines the special action is assembled in the next word. The sign bit of the word containing the output-control code is set to distinguish this as a control word rather than data.

The processing functions performed by the input-sequence control program are illustrated in Fig. 8. Upon reception of an ETX character when only one word in the data block remains to be filled, an additional data block may be seized to assemble the output-control

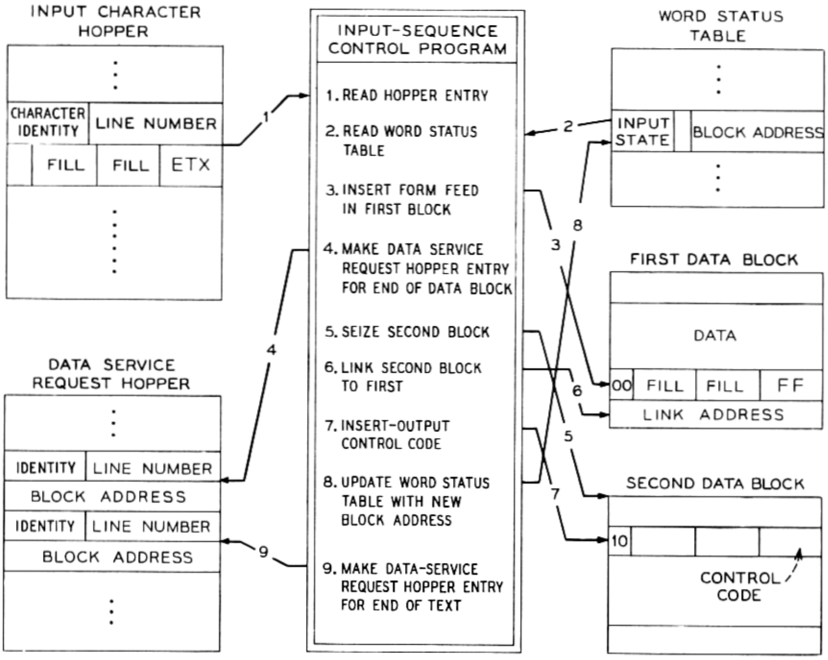

INPUT CHARACTER HOPPER
CHARACTER IDENTITY
LINE NUMBER
FILL
FILL
ETX
DATA SERVICE REQUEST HOPPER
IDENTITY
LINE NUMBER
BLOCK ADDRESS
IDENTITY
LINE NUMBER
BLOCK ADDRESS
INPUT-SEQUENCE CONTROL PROGRAM
1. READ HOPPER ENTRY
2. READ WORD STATUS TABLE
3. INSERT FORM FEED IN FIRST BLOCK
4. MAKE DATA SERVICE REQUEST HOPPER ENTRY FOR END OF DATA BLOCK
5. SEIZE SECOND BLOCK
6. LINK SECOND BLOCK TO FIRST
7. INSERT-OUTPUT CONTROL CODE
8. UPDATE WORD STATUS TABLE WITH NEW BLOCK ADDRESS
9. MAKE DATA-SERVICE REQUEST HOPPER ENTRY FOR END OF TEXT
WORD STATUS TABLE
INPUT STATE
BLOCK ADDRESS
FIRST DATA BLOCK
DATA
00
FILL
FILL
FF
LINK ADDRESS
SECOND DATA BLOCK
10
CONTROL CODE
1
2
3
4
5
6
7
8
9

Fig. 8—Illustration of functions performed by input-sequence control program.

code and two data-service-request hopper entries are made, the first indicating the end of assembly in the original data block and the second indicating the completion of the message. The work performed on this unit of input information is unusually large but illustrates the type of functions performed by this program.

5.2 *Line Output*

It is the output-sequence control program which delivers data to the lines. This involves the scanning of the character-output buffers, mentioned earlier, and the orderly word-by-word disassembly of data blocks as each idle buffer is detected. The data blocks comprise parts of messages received from input lines and retrieved from message store, service messages generated by the processing programs, or blocks of fixed data used in the signaling dialog with the stations. The output program is not cognizant of the source or purpose of the data. It merely goes about its task of processing words of data at the direction of other programs which manipulate the data blocks.

The character-output buffers must be scanned periodically at a rate fast enough to maintain the character rate of the associated line. For line rates of 150 bits per second, a visitation period of less than 200 ms is required. To achieve the required timing, the output sequence control program is operated as a J-level interrupt task with a 10-ms periodic entry. It would be desirable to scan only active lines to maximize the scanning efficiency, and equalize the work during each entry to minimize the effects of peaking. Both of these objectives are achieved by driving the output-sequence control program from a work list of active lines called the output work table. For all lines of 150 bits per second or less, this table consists of 18 subtables of n words each as shown in Fig. 9 where n is chosen to meet the output capability of the office. On each J-level entry the output-sequence control program services one subtable, so that the character-output buffer for each active line is scanned once every 180 ms. Each entry

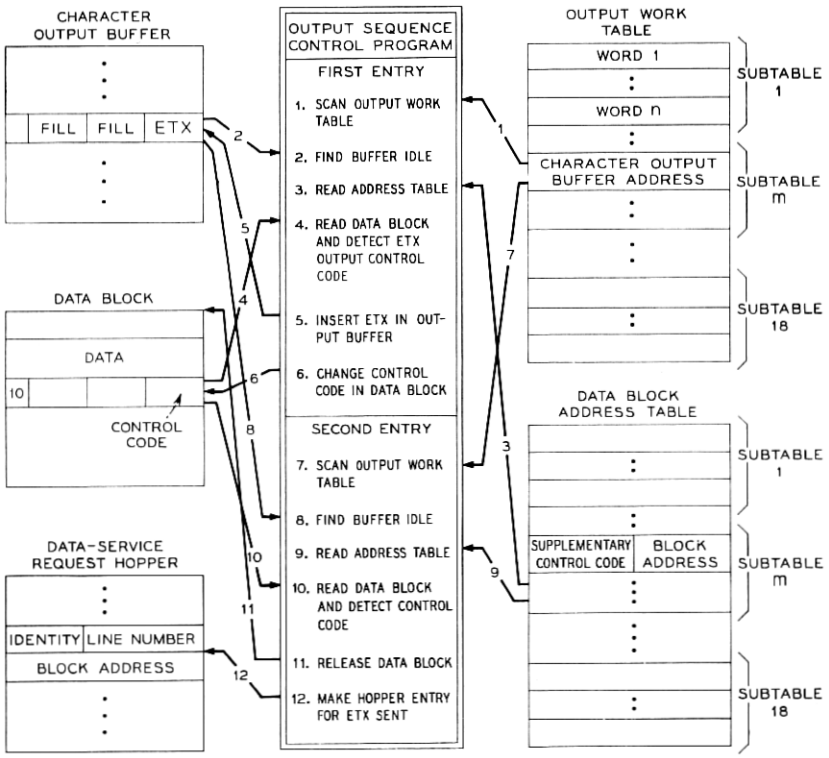


Fig. 9—Illustration of functions performed by output-sequence control program.

contains the character-output buffer address of an active line requiring scanning.

A second table, which complements the output work table, is used to store pointers to the next word to be transmitted to the active lines. For each entry in the output work table, there is a companion entry in the data-block address table in the same relative position. This relative position is defined as the data-block index.

A processing program that wishes to transmit a block or group of blocks of data to a given line must seize an idle location in the output work table. The busy-idle state of each entry is maintained in a matrix in which each column defines a subtable of the output work table and a row defines the word position in each subtable. The seizure process, a systematic search of the matrix from top to bottom, results in a concentration of work in the upper portion of each subtable and equalizes the work in each subtable. Upon seizing an entry, the processing program loads the data-block address table with the address of the data block to be transmitted and the output-work table with the character-output buffer address of the desired line.

A data-block entry which has the sign bit set to one indicates the presence of an output-control code which informs the output program that some special action is required. This allows the output-sequence control program to perform the task of block disassembly in the fastest, most routine, repetitive manner possible. Output control codes are inserted in data blocks by the input-sequence control program when assembling data blocks or by the processing programs when generating data blocks for output transmission. There are some 49 output-control codes defined to perform such diverse actions as the detection of the end of signaling sequences or units of a message, insertion of transmission-free timing intervals on the lines, transmission of break signals, and detection of data-block format errors. While some of the codes result in special action and then the continuation of the disassembly process, the majority of the codes result in a cessation of further disassembly and a report to a processing program so that it can perform some action at that point in the disassembly process. These reports are buffered to the base-level programs through the data-service-request hopper. In making entries in this hopper, the output-sequence control program must algorithmically convert the character-output buffer address to the DLN.

A secondary control mechanism is provided through the use of the supplementary-control code stored in the data-block address table. These codes deal primarily with halting further disassembly for one of a number of reasons. For example, one encoding of this item stops further output while waiting for the next block to be retrieved from the message store. In addition, if a processing program wishes to halt the output transmission to poll the station or transmit a service message concerning an input transmission, a supplementary-control code is written into the data-block address table. Detection of this code by the output program results in a halt of further disassembly and a report in the data-service-request hopper.

Figure 9 illustrates the processing functions performed by this program when it detects the output-control code for the end-of-text indication assembled by the input-sequence control program. For this particular case, the output-sequence control program transmits the ETX character and changes the output-control code in the block. It is only upon detecting this second code that a data-service-request hopper entry is made. This ensures that the character-output buffer has been unloaded before the processing programs are informed that the message delivery is complete.

5.3 *Message Store Input-Output*

The last of the major input-output programs performs both input and output transfers of data blocks to and from the message store. This program, called the message-store administration program, consists of two functionally independent parts: (*i*) the message-store input-output routine, a J-level program, which services the message-store instruction queue, and (*ii*) the message-store service routines which maintain a busy-idle status for each block of message store and perform common functions required to interface with the input-output routine.

Figure 10 illustrates the memory areas associated with the message-store program. A processing program that wishes to store a block of data on the message store makes a direct transfer to a message-store service routine. There are as many entry points to the service routines as there are services performed by these routines. The first task of the service routine is to assign a destination disk address for the data block residing in call store. This is done by consulting a busy-idle map. Each block is represented by a single bit in the map. In searching the map for an idle block, a pointer is used to keep track of the last map word that was read. In this way, those map words most likely to contain idle blocks are searched. The number of words scanned in a search is limited to prevent real-time abuses when disk usage is near capacity. If an idle block is not found within the limit,

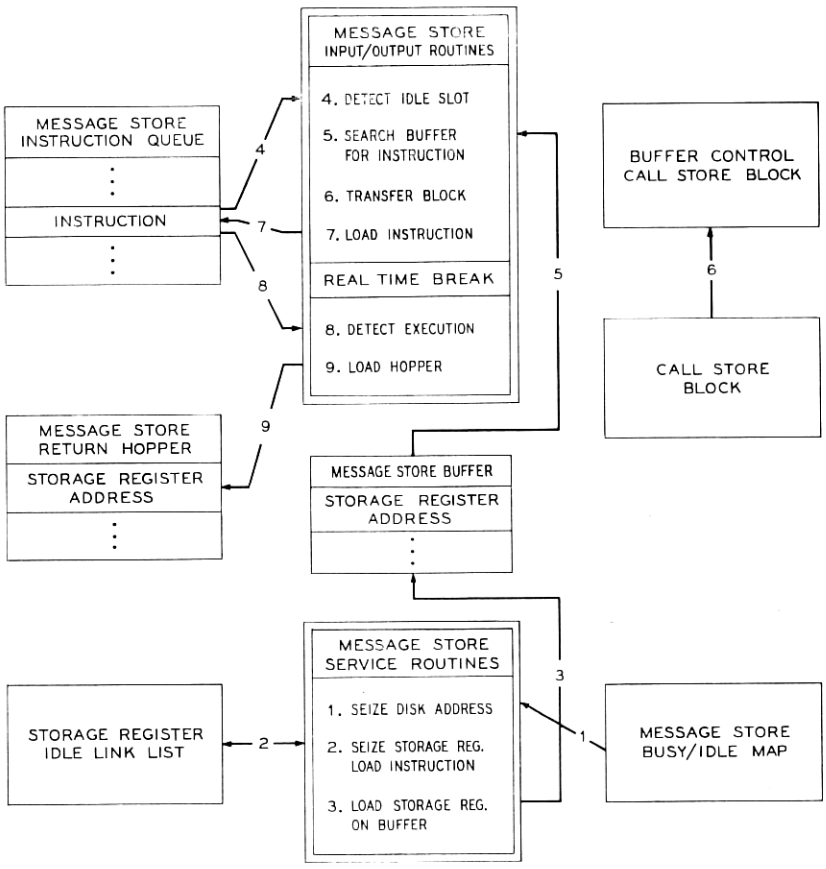

MESSAGE STORE INPUT/OUTPUT ROUTINES
4. DETECT IDLE SLOT
5. SEARCH BUFFER FOR INSTRUCTION
6. TRANSFER BLOCK
7. LOAD INSTRUCTION
REAL TIME BREAK
8. DETECT EXECUTION
9. LOAD HOPPER
MESSAGE STORE INSTRUCTION QUEUE
INSTRUCTION
BUFFER CONTROL CALL STORE BLOCK
CALL STORE BLOCK
MESSAGE STORE RETURN HOPPER
STORAGE REGISTER ADDRESS
MESSAGE STORE BUFFER
STORAGE REGISTER ADDRESS
MESSAGE STORE SERVICE ROUTINES
1. SEIZE DISK ADDRESS
2. SEIZE STORAGE REG. LOAD INSTRUCTION
3. LOAD STORAGE REG. ON BUFFER
STORAGE REGISTER IDLE LINK LIST
MESSAGE STORE BUSY/IDLE MAP
1
2
3
4
5
6
7
8
9

Fig. 10—Illustration of functions performed by message-store administration program.

a failure return is given to the processing program. The resulting address defines the sector and entry location in the message-store instruction queue into which the instruction must be loaded. Because the desired instruction slot may not be presently available, a waiting list of instructions must be maintained. This waiting list is called the message-store buffer. Since the message store is sector oriented, this buffer is also sector oriented and consists of 16 link lists. The linked-memory areas are called storage registers. The second function of the message-store service routines is to seize an idle storage register, load the instruction data, and link the register to the appropriate sector-oriented message-store buffer link list. The instruction data consists of the instruction, the message-store address and the call-store address. In addition, a processing-register address is needed to associate the transfer request with a particular data call. The notification of the successful block transfer is directed to the processing register so designated.

The message-store input-output program, a J-level program, monitors the action of the message-store hardware in executing the instructions. On finding that a given slot in the message store instruction queue is idle, the corresponding link list in the message-store buffer is consulted to determine the next block to be transferred. Since the message-store hardware can only access the buffer-control call store, the data block must first be transferred from its call-store location to a buffer-control call store. A dedicated area, one block per sector, is defined for this purpose. The instruction is then generated and stored in the message-store instruction queue.

At some later time, the input-output routine will detect that the aforementioned instruction was executed. At that time, the storage register, used initially for the transfer request, is entered into a link list called the message-store-return hopper. This hopper acts as a buffer between the J-level input-output routine and the base-level processing programs for notification that the requested transfer has been successfully completed.

The message store is organized on a block basis to achieve adequate throughput with a mechanically rotating device having large access time. The message-store administration program provides the capability of single-word transfers. In the case of a single-word write instruction, the input-output routine generates two instructions: (*i*) read the data block in which the one word of data is written, and (*ii*) write the resultant data block back to the message store.

In storing messages and message-processing blocks on disk, the blocks are linked together so that the data can be referenced with a minimum number of addresses stored in call store. This linking scheme is illustrated in Fig. 11. The first and last words of each block contain reverse and forward link addresses and only the 30 central words contain data characters or processing information. The double linking is provided so that the program can retrieve the data using either the initial data-block address or the final address. In storing a block of data on message store, the address of the next block must be known to insert the proper link address in the data block. Thus the service routines provide separate entries for a first-block transfer, two message-store addresses are seized; for a middle-block transfer, one mes-

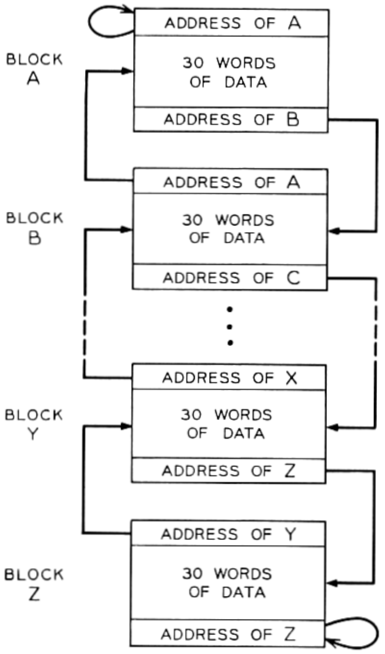


Fig. 11—Illustration of double linking of data blocks.

sage-store address is seized; and for a last block transfer, no seizure is required.

VI. MESSAGE-PROCESSING PROGRAMS

The message-processing programs, each of which handles a given phase of processing, are relatively complex because the design covers many station arrangements, line types, codes, and optional features. The decision to incorporate all station characteristics and options within a single program was made to achieve efficient use of common logic functions. In the following sections the processing of a typical message is used to describe the functions of these programs.

6.1 *Polling*

The polling program consists of several major functions which provide the segmented control for this cyclic process, as illustrated in Fig. 12. To observe the operation of this program, the cycle will be entered at the point of generating the polling code. The polling pro-

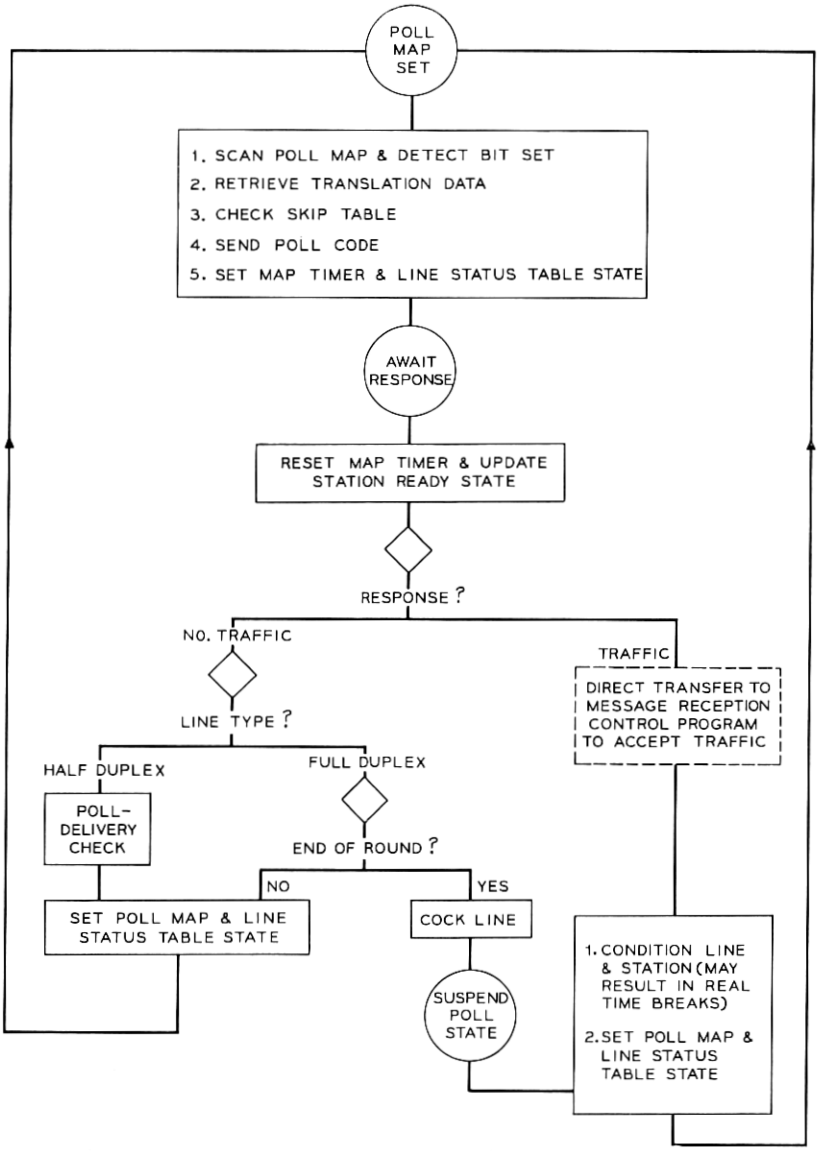

POLL MAP SET
1. SCAN POLL MAP & DETECT BIT SET
2. RETRIEVE TRANSLATION DATA
3. CHECK SKIP TABLE
4. SEND POLL CODE
5. SET MAP TIMER & LINE STATUS TABLE STATE
AWAIT RESPONSE
RESET MAP TIMER & UPDATE STATION READY STATE
RESPONSE ?
NO. TRAFFIC
TRAFFIC
DIRECT TRANSFER TO MESSAGE RECEPTION CONTROL PROGRAM TO ACCEPT TRAFFIC
LINE TYPE ?
HALF DUPLEX
FULL DUPLEX
POLL-DELIVERY CHECK
END OF ROUND ?
NO
YES
SET POLL MAP & LINE STATUS TABLE STATE
COCK LINE
SUSPEND POLL STATE
1. CONDITION LINE & STATION (MAY RESULT IN REAL TIME BREAKS)
2. SET POLL MAP & LINE STATUS TABLE STATE

Fig. 12—State diagram of polling program.

gram is entered periodically as a base-level task by the executive control program to poll a given number of lines, the number being an office parameter. The particular lines to be polled are defined in the poll map, a dedicated area of call store in which each line is represented by a single bit. The program scans the poll map; and when it detects a bit set to one, it interprets this as an indication of a line to be polled. The translation service routines are used to retrieve the characteristic data about the line: code, type, and options available to the line. The program then consults the skip table for the line to determine the next station to be polled. The basic function of the skip table is to indicate, with a particular bit pattern, which stations on a line are to be polled or skipped and, with another bit pattern, which stations have indicated in the past a receiver not-ready status. The skip table also contains a counter, called the poll-table index, which indicates which of the stations on the line should be polled next. The skip-table bits are examined for the station referenced by the poll-table index and the station is polled if the station is not on skip or if it is not-ready. Stations on skip, but not-ready, must still be polled to detect the transition from the not-ready to ready state so that traffic may be delivered to the station. The poll-table index defines the specific poll code to be sent to the line. This code is loaded into the character-output buffer directly and a per-line timer is set to make certain that stimulus on the line is not lost if the station should fail to respond. The line-status table state is updated so that the response will be passed to the proper polling routine for analyzing the response. The poll-code generation segment is then complete.

It has been noted that the station response to polling is timed. In fact, all expected signaling responses are timed. This function is administered by an executive control timing routine which makes use of two bits of memory per line organized into two maps similar to the poll map. The timing routine examines each map alternately at the end of three-second timing periods. During odd intervals the first map is set by the processing programs for response timing. During even intervals while the second map is being set, the first map is allowed to time the responses. Any successful response results in the processing-program resetting the corresponding bits of both maps. At the end of the even interval, the timing routine will examine the first map; and any bits still set indicate a response time-out. The line-status table state is indexed to produce the proper entry point in the processing program required to produce the necessary actions as a result of the time-out. The map-timing technique achieves a large economy in storage allocation in comparison to conventional link list timing facilities and ensures that a timing facility is always available for every line. The penalty is a sacrifice in accuracy, $\pm\frac{1}{3}$ of the average response time-out.

Under normal conditions, the polling response will be received by the line terminal hardware and entered in the input-character hopper. Upon its periodic visitation of the input-character hopper, the input-sequence control program will detect the entry and, based on the input state of the word status table, will make an entry in the data-service-request hopper. This entry will be passed to the polling program for analysis of the response. If the polling response indicates a request to originate traffic and the priority options of the line allow the present request to be serviced, control is relinquished by a direct transfer to the message-reception control program, which services the request after updating the ready status in the skip table. The polling process has thus been interrupted to accept the message origination. The polling cycle for this line will be restarted at the conclusion of the message origination.

If a no-traffic response is received for a full-duplex line, the polling program must decide whether or not to poll the other stations on the line. If each station has not been polled once since the cycle was started, the poll map is set to initiate polling of the next station. If the polling round is complete, the polling program will suspend further polling by loading the character-output buffer directly with a set of characters to "cock" the line. The line-status table state is updated to reflect an idle state and an additional bit is set to indicate that the line is in the suspend-poll state. The polling will restart on the next quarter hour by an executive control task which examines the suspend poll bit in each line-status table entry. An overview of the polling process, illustrating the relationship between the program and the involved memory areas, is shown in Fig. 13.

On HDX lines, traffic can proceed in only one direction at a time and so the input and output of traffic are necessarily in contention for use of the line. On reception of a no-traffic response, the polling program must determine its next action based on a poll-delivery criteria which establishes a precedence order between the four delivery priorities and the two origination priorities as follows:

(*i*) deliver all urgent and rush traffic,
(*ii*) pick up priority traffic (lead-round polling),
(*iii*) deliver all normal traffic,

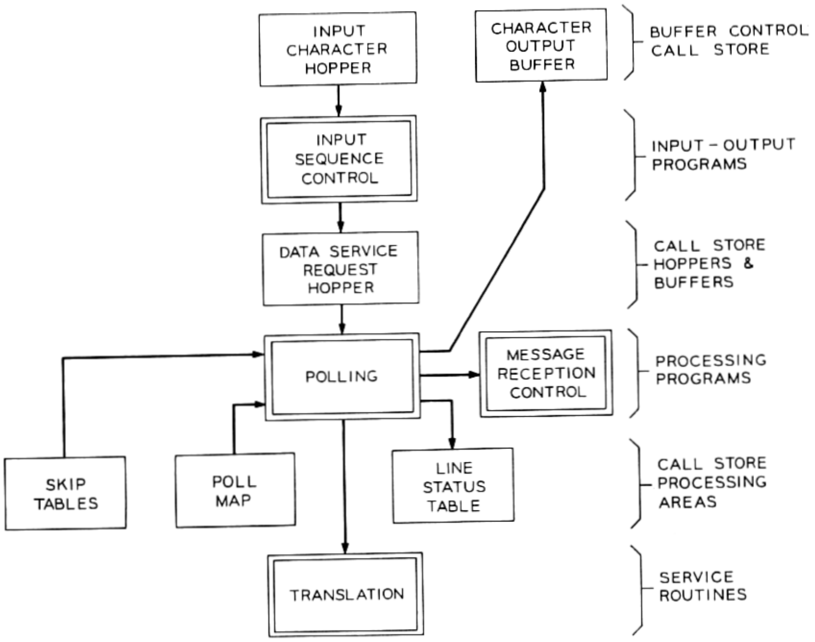


Fig. 13—Relationship between program and memory areas used in polling process.

(*iv*) pick up of regular traffic (general-round polling),
(*v*) deliver all deferred traffic, and
(*vi*) continue polling (subordinate round polling).

To make the proper decision, the polling process for HDX lines is subdivided into three levels or polling rounds, as indicated in the poll-delivery criteria: lead round, general round, and subordinate round. Memory of current polling round is maintained as an item in the polling field of the line-status table. This item, in conjunction with the message waiting bits, determines the next action to be taken by the polling program. If a delivery of a message is indicated, the polling program relinquishes control of the line and initiates the delivery process by setting a bit in the nomination map, which is discussed in greater detail in Section 6.7. If polling is to continue, the poll map is set to initiate the polling of the next station on the line.

6.2 *Heading Origination*

As noted in the preceding section, the message-reception control program is entered by the polling program when it is found that a station is ready to originate a message. Using a set of general service routines, the message-reception control program: (*i*) seizes an input-processing register and links it to the line-status table, (*ii*) updates the line-status table state to indicate that the polling field now contains a register address, (*iii*) seizes input capacity—a fictitious facility used to control the input load presented to the system, and (*iv*) seizes a message-processing block and links it to the input-processing register. Any failure to seize a given item results in queuing for that item. The translation program is then entered to retrieve translation data concerning the originating station. The data defining the line characteristics and station options are loaded into the input-processing register, and the message-processing block is loaded with the originator's identity and the user's identity. The user in this context does not refer to an individual station but rather to the community of stations composing a given network. In addition, the program also seizes an output-work table assignment for later use in transmitting the generated heading to the originator. As noted earlier, this assignment is in the form of a data-block index which is stored in the input-processing register. Next the identity of the originating station is verified. The message-reception control program loads the character-output buffer with the originator's call-inquiry code obtained from the translation data and sets the response timer. If the proper station has been identified, the station will respond with a start-of-heading character.

When the station response is received through the data-service-request hopper, the program initiates the construction and delivery of the generated heading to the originator. A data block is seized and a direct transfer is made to the heading-generation program. Based on the station options specified in the translation data stored in the input-processing register, the heading-generation program assembles the desired heading information in the data block. This information will generally consist of a message number, time and date, and a start-of-text (STX) control character which restarts the originator's transmitter, allowing the heading portion of the message to be transmitted to the switching center. In addition, the heading-generation program inserts the message number, time and date into the message-processing block independent of the station options. Upon return to the processing program, the message-processing block is stored on the message store. The service routines provided by the message-store administration program are used to seize a block address in a specified area of the message store and load a storage register on the message-store buffer to transfer the message-processing block to disk.

The message-processing block now contains an identification of the originator and message to be originated. The purpose of this operation is message protection. If the message is lost in the system for any reason, a record of the message can be obtained from the message store. Next the data block address is loaded into the data-block address table, and the DLN is algorithmically converted to a character-output buffer address and loaded into the output work table to initiate the delivery of the generated heading to the originator.

The output-sequence control program systematically transmits the data to the station. After transmitting the restart code to the station, the program detects an ouput-control code inserted in the data block by the heading-generation program. This code directs the program to terminate the transmission by inserting an idle state in the supplementary control code of the data-block address table. The data-block address is removed from this table and inserted in the word-status table so that the block can be used for the assembly of the input heading. A data-service-request hopper entry is made to inform the processing program that the station transmitter has been started. In response to this entry, the message-reception control program sets the response timer to monitor the line for the receipt of the first three characters of heading as an indication that the station has responded to the restart code. The program tag in the input-processing register is updated for this monitoring state.

Upon receiving the first three characters of heading, the input-sequence control program assembles the characters into the data block and makes a data-service-request hopper entry. In response to this entry, the message-reception control program resets the response timer, releases the output-work table assignment, and updates the program tag to the receive-heading state. The input-sequence control program continues to assemble the heading into data blocks. As each block is completed, the message-reception control program is informed through a data-service-request hopper entry. Each block is ordered to the message store. Upon acknowledgment from the message-store administration program that a given block has been transferred to disk, the data blocks are relinked and retained in call store for the analysis of the heading. When the end-of-heading indication is received, the last block of heading is ordered to disk and the program tag is updated to a receive-text state. For FDX lines, the assembly of the text will continue concurrently with the analysis of heading. When all heading blocks have been acknowledged, the message-reception control program initiates the analysis of the heading indirectly by loading the input-processing register address on the heading-analysis hopper.

6.3 *Heading Analysis*

The heading-analysis program services the heading-analysis hopper. The analysis consists of an examination of each character of heading assembled in the data blocks, formatting of certain of the characters for translation of the data, and the storing of information in the message-processing blocks in a form appropriate for future action by other programs in the processing of the message. When the analysis is complete, no other program will need to examine the data characters on an individual basis.

The heading format for a data message is quite rigid and may consist of the following units of information separated by delimiters:

(*i*) a 10-digit directory number,
(*ii*) a mnemonic code, which is selected by the user to address a single station or group of stations in the user's realm of interest,
(*iii*) a precedence designator, which defines the priority order of delivery to each terminator, and
(*iv*) personal address information, which can be used to define an individual or department being served by a given station.

The delimiters and maximum number of characters per unit of information are shown in Table II. For action requests which ask that the switching center perform given functions rather than the delivery of a message, additional units of information are defined as shown in Table II. The input processing of an action request is indistinguishable from message processing, with the exception of the heading analysis. At the completion of the transmission of an action request, second-level processing programs are called upon to execute the request based on information loaded in the message-processing block by the heading-analysis program.

The analysis of the heading is a table-driven process. Each character is used as an index into a 128-word table, one word for each character in the ASCII code set. If the word in the table is not flagged (sign bit = 0), the word contains a six-bit stripped ASCII character which merely compresses the subset of alphanumeric characters to conserve storage. If the word in the table is flagged (sign bit = 1), the word contains a set of indexes used with other tables. The particular choice of index and second table depends on the unit

TABLE II—THE UNITS OF INFORMATION, THE DELIMITERS, AND THE CHARACTERS PER UNIT THAT MAY BE INCLUDED IN A HEADING

Unit of Information	Starting Delimiter	Ending Delimiter	Maximum Count of Characters
For Message or Action-Request Headings			
Mnemonic	Alphanumeric or Hyphen	CR, LF, Space or STX	7
Directory Number	<	>	10
Precedence	Alphanumeric or Hyphen	CR, LF, Space or STX	7
Personal Address Information	[	]	63
For Action Requests Only			
Action-Request Order	/	/	7
Data Line Number	Numeric	CR, LF, or Space	4
Circuit Number	Alphanumeric or Hyphen	CR, LF, Space or STX	14

of information being processed. These second-level tables contain addresses of the particular routine necessary to process the given character.

When a given unit of information has been assembled and identified, it is usually necessary to translate and store the information in the message-processing block in a form more useful for the other programs. For example, the result of a mnemonic translation is a normalized directory number, which is a compressed form of the directory number used for routing traffic within the switching center. Since mnemonics are chosen completely at the discretion of the user, a set of mnemonic translation tables are required per user set of stations. To translate a mnemonic, the heading-analysis program provides the translation program with the mnemonic and the user identity which was stored in the message-processing block by the message-reception control program.

As each addressee in the message is determined, a terminator slot in the message-processing block is loaded. As additional storage is required, the heading-analysis program seizes additional message-processing blocks and links them to the original block. A mnemonic may translate to either a single terminator or a group of terminators. For each terminator the normalized-directory number is loaded into the terminator slot. Before loading a terminator slot, certain screening to eliminate duplicate addresses is done. The originator is always screened from a group code; and if a terminator has a single-copy option, the resulting normalized-directory number is screened against all previously established terminator slots. Also loaded into the terminator slot is the relative address of the first character of the mnemonic and the number of characters. This information is required by the heading generation program to retrieve the relevant address in constructing the terminator's generated heading. The location of the first character of the mnemonic is a relative address consisting of the heading block number, word position, and character position in the word. The absolute address cannot be inserted in a terminator slot because each time a program needs the heading block the call-store address may be different, having been written from message store to call store. If a given addressee has been eliminated because of the single-copy option, a terminator slot is still established to store the relative address of the mnemonic. This allows the heading generation program to retrieve all the relevant address information used in the heading to address a single station. When a personal-addresss information unit is identified, its relative address is also inserted in the appropriate terminator slots.

At the completion of heading analysis, the state of processing the incoming message is unknown since the assembly of text occurs concurrently with the analysis. The heading-analysis program therefore transfers indirectly to the message-reception control program using the program tag in the input-processing register. If a format error or nontranslatable mnemonic is detected by the heading-analysis program, an indication of this condition is given to the message-reception control program. In this way the input message may be aborted with a service message to the originator indicating the field or unit of information that was found to be in error. As a result of this transfer, the message-reception control program updates the program tag to indicate the completion of heading analysis and releases the call-store copy of the heading.

6.4 *End of Originated Message*

The message-reception control program continues the processing of the input message. A data-service-request hopper entry indicating the completion of text assembly in a given data block results in the data block being ordered to disk. As each block transfer to disk is acknowledged by the message-store administration program, the message-reception control program releases the call-store copy of the data block. When the first block of text is stored on disk, and if the heading has been analyzed, the message-reception control program calls upon the message-queue insertion program as a service routine to generate a delivery-queue entry for each terminator whose line speed is less than or equal to that of the orignator. This allows for rapid speed of delivery, i.e., the delivery of the message can be started while the same message is still being originated. The process of generating the delivery queues are discussed in greater detail in the next section.

When an end-of-text entry is received by the message-reception control program, the complete message has been received from the station. Continued processing of the message can occur without further association with the line. The input-processing register is disassociated from the line by removing the register address from the line status table and setting the line status tag to an await-End of Transmission (EOT) state. The last text block is ordered to disk and the program tag in the input-processing register is updated to an end-of-message state. The message-reception control program continues to control both the line and the message independently through the line-status table and the input-processing register, respectively.

The await-EOT state is used to monitor the line for multimessage transmission. A bit in the line status table is set to initiate intermessage timing on the line. If no further stimulus is received from the line, an intermessage time-out will occur in approximately 48 seconds. The message-reception control program initiates a line-abort procedure which transmits an emergency-stop sequence of characters on the line which in turn normalizes the stations on the line. If a start-of-heading character is received indicating that the station wishes to transmit an additional message, the message-reception control logic for premessage initialization is entered. If an end-of-transmission character is received, the intermessage-timing bit is reset and input capacity which has been held during the duration of the transmission is released. For FDX lines, the polling program is entered to condition the line for polling and to set the poll map which initiates polling on the line. For HDX lines, the choice between delivering a message to the line or polling the line for additional input traffic is based on the poll-delivery criteria discussed earlier. Based on the message-waiting bits in the line status table and the current polling round, either the poll map is set to initiate polling or the nomination map is set to initiate the delivery of a message.

The state word in the input-processing register is used to control further processing of the message. The entries that are handled during the end-of-message state consist of message-store returns which acknowledge the successful transfer of data blocks to disk and, in the case of messages with very short text length, heading-analysis returns indicating the successful completion of the processing of the heading information. After all data blocks have been acknowledged and the heading successfully analyzed, the message-reception control program relinquishes control of the message to three programs which complete the input processing. First, the permanent file program is entered to update the retrieval tables stored on the message store for each station, which aids in the rapid retrieval of messages requested by the users. This program is one of a group of programs which control the tape subsystem.[3] Second, the message queue insertion program is entered to established a delivery request, i.e., a message queue entry for each addressee that had not been processed when the first block of text was received from the originating station. Third, the message-termination program is entered to complete the processing of the input message.

The primary functions performed by the message-termination program are billing, traffic counts, and the final disposition of all call-store facilities associated with the message. The program transfers control to the automatic-message-accounting (AMA) program as a service routine which updates customer summary-billing registers associated with the originator. This data is transferred to an AMA tape once per day. The message-termination program then uses several subroutines to update system and user-traffic statistic counters. The user-traffic statistic data concerning all stations in the user set are periodically delivered to the user-control stations so designated by the user. When these tasks are completed, the message-termination program determines from an item of storage in the message-processing block if the call-store copy of the message processing block is being used by any other program; e.g., the message is being delivered to one or more of the addressees. If it is not being used by another program, the message-termination program transfers the message-processing block from call store to disk, using the message-store administration-service routines. When this transfer is completed, the input-processing register and the call-store copy of the message-processing block are released. If another program is using the message-processing block, then only the input-processing register is released. The transfer of the message-processing block to disk is thus left to the last program using the call-store copy.

6.5 *Generation of Delivery Queues*

The heart and substance of a store-and-forward switching system is the message queues, whereby the system may hold many messages for a particular terminator to insure maximum utilization of the transmission facilities. The message-queue insertion program generates the message queues and administers other special functions which affect the message queues. For example, one such special function is the shifting of entries from one queue to another when the original terminator is placed on alternate delivery. The message-nomination program, which is discussed in a later section, selects entries from the queues and initiates the actions required to deliver messages to the terminating stations.

The requirements and objectives imposed on the message queues lead directly to the memory organization. First, the entries in the message queue must be time ordered within four precedence categories: urgent, rush, normal, and deferred. Second, the holding time on the call-store facilities which provide the message-queue function is very large, so that the efficient use of such facilities is highly desirable. In addition, the different functional areas should be administered from a single pool of call-store facilities to provide the greatest efficiency under different traffic mixes. Third, the structure of the message queue must lead to a single call-store copy of the message-processing block when processing concurrent deliveries to several lines. Besides the economic reasons, this is necessary to insure that the results of one transmission do not overwrite and destroy the results of another transmission when attempting to update the disposition information on the disk copy of the message-processing block.

Each data line or multiline hunting group is assigned a unique queue index which for convenience is identical to the poll index. This index identifies the memory in two fixed tables associated with each line or multiline hunting group. The first of these tables is the nomination map which has a format similar to the poll map. It contains one bit for each queue index, i.e., for each message queue, and provides the stimulus to service a given queue. A given bit which is set to the "1" state indicates that the associated queue contains one or more messages and that the line is available for transmitting traffic. The second table is the message-queue table which contains two words for each queue index, as shown in Fig. 14. The first word contains a pointer to the queue-location register if a queue currently exists; otherwise the word is zero. The second word contains a count of the number of messages currently in the queue, which is used in making queue reports to the user-control location. The queue-location register is a three-word call-store facility which contains pointers to the subqueues which are organized by precedence. The first word points to the high-priority subqueue which contains both urgent and rush messages; the second word points to a normal-priority subqueue;

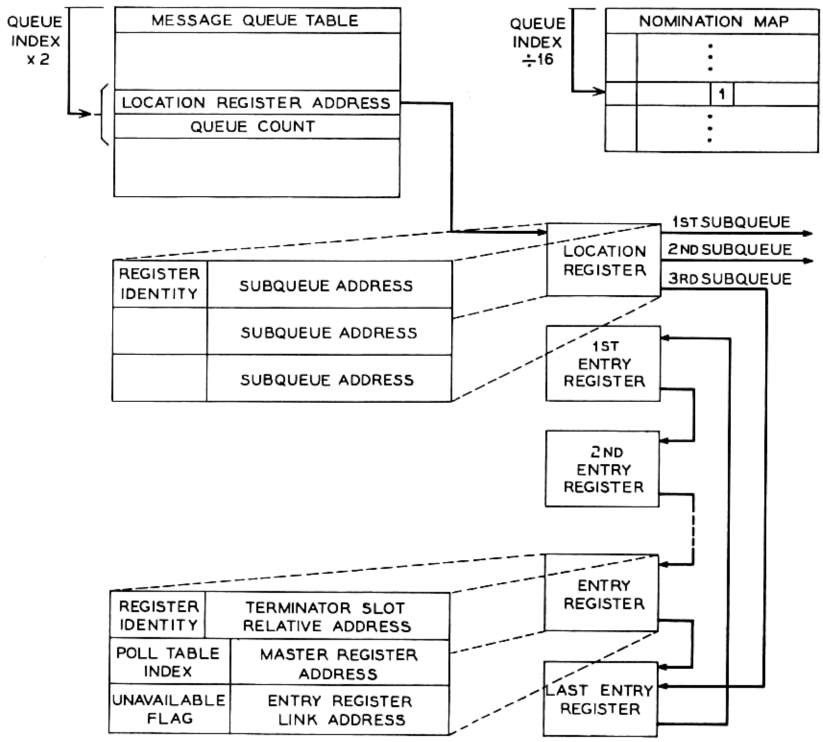


Fig. 14—Organization of message queue memory areas showing the circular linking of entry registers.

the third word points to a deferred-priority subqueue. The urgent and rush messages are combined in a single subqueue so that the queue-location registers can be assigned from a common pool of three-word registers used for all message-queue functions. The percentage of messages using the two higher-precedence categories is very low, so that combining these precedences in a single subqueue does not add an appreciable work load for the processing associated with this subqueue.

Each subqueue is made up of three-word facilities called entry registers which are linked together to form a one-way circular link list as shown in Fig. 14. One entry register is used to refer to each addressee in a given message. The subqueue pointer in the queue-location register points to the last-entry register in the subqueue. The circular nature of the link list makes it possible to get to the last entry for adding additional entries to the queue or to the first entry for servicing the queue in a very few logical steps with a minimum of storage.

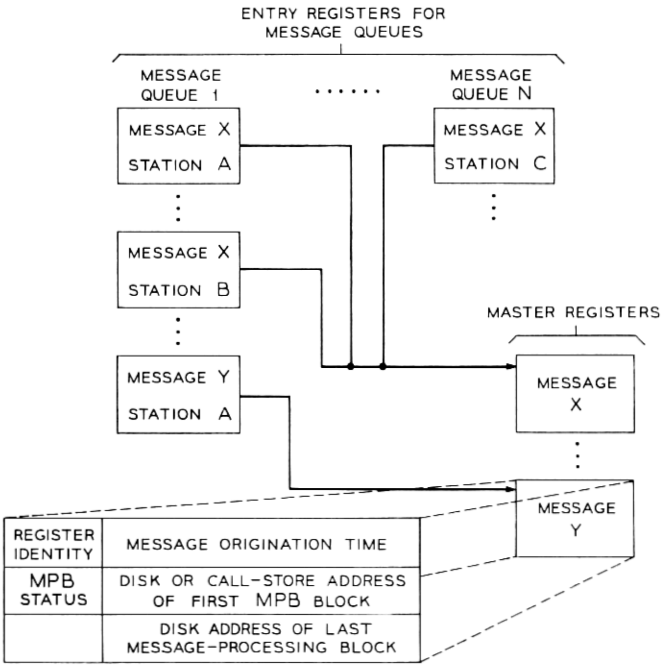


Fig. 15—The master register provides a single common linkage to the message-processing block for all message-queue entries associated with a given message.

Associated with each message is a three-word call-store facility called a master register. The master register indicates whether the message-processing block is located in call store, message store, or in the process of transfer to or from call store and the address of the first block of the message-processing block. The entry registers for each addressee in the message point to the master register, as shown in Fig. 15. The master register provides a common linkage to the message-processing block for all message queue entries. This allows deliveries to all terminators to be initiated independently while maintaining at most a single copy of the message-processing block in call store. As with queue-location registers and entry registers, the master registers are seized from a common idle-link list of three-word call-store facilities called message-queue registers. An item in the register is used to identify the function assigned to a given facility.

In the normal processing of an input message, the message-queue insertion program is entered by the message-reception control program for the purpose of forming the message queues. This occurs after the first block of text is assembled and also at the completion of the input transmission. To perform its functions, this program requires an available processing register which is linked to the message-processing block. If a master register does not exist, the message-queue insertion program seizes one and initializes it with the appropriate data. The program then proceeds to process sequentially each terminator slot in the message-processing block that was established by the heading-analysis program. When an unprocessed-terminator slot is found, the addressee's directory number is used as an input to the translation subroutines to obtain all of the station related information required for queuing. A privacy check is made, based on the translation data, to determine if the addressee will accept messages from the originator. If the privacy check fails, the message is sent to the originator's user-control location with a service message indicating why the message was undeliverable.

The next operation performed by the program is to check the status of the addressee. If the addressee is on alternate delivery to another station, the disposition item in the terminator slot is marked to indicate the alternate delivery and a new terminator slot is formed for the alternate addressee. If the message to be alternate delivered has already been alternate delivered from another station, an illegal condition exists which may result in "ring-around-the-rosy." In this case, the message is sent to the terminator's user-control location. If the status of the addressee indicates that the station is in a hold state or in a not-ready state, the entry register for this addressee is marked unavailable for delivery. This will cause the message-nomination program to skip over this entry when processing the queue for the given line.

After the station-status checks are made and it has been determined that the current terminator slot should be queued, the message-queue table is examined to determine whether or not a message queue exists for the terminator's queue index. If a message queue does not exist, then a queue-location register is seized and linked to the message-queue table. An entry register is then seized and linked to the appropriate subqueue through the queue-location register. The entry register is linked to the master register and to the terminator slot in the message-processing block using the relative slot address. The station identity consisting of the station's poll-table index is also inserted in the entry register. The queue counter in the second word of the message-queue table is incremented and, if the station status is such that immediate delivery can take place, the message-waiting bits in the line-status table are set to indicate the appropriate precedence, and the nomination map may be set.

After the entry register is linked in the message queue, the associated terminator slot is marked to indicate a queued disposition. The program then moves on to the processing of the next terminator slot. If the message-processing block should contain many addressees, the message-queue insertion program must segment its work and relinquish control so that other tasks may be performed. This is done by placing the processing register on a timing list. When the time-out occurs, the program will continue the processing until all terminator slots have been examined and a message-queue entry has been formed for each addressee, or until another time break is necessary.

6.6 *Summary of Input Processing*

Figure 16 summarizes the input-message processing sequence. The poll-administration program generated the station-polling codes and analyzed the station responses to determine the desire to originate traffic. On a positive response, the poll-administration program transferred control of the line to the message-reception control program which proceeded to seize and initialize the necessary temporary memory areas for the incoming message. The heading-generation program was used as a service routine to assemble a generated heading which typically included time, date, and message number. At the conclusion of the delivery of this information to the originating station, a

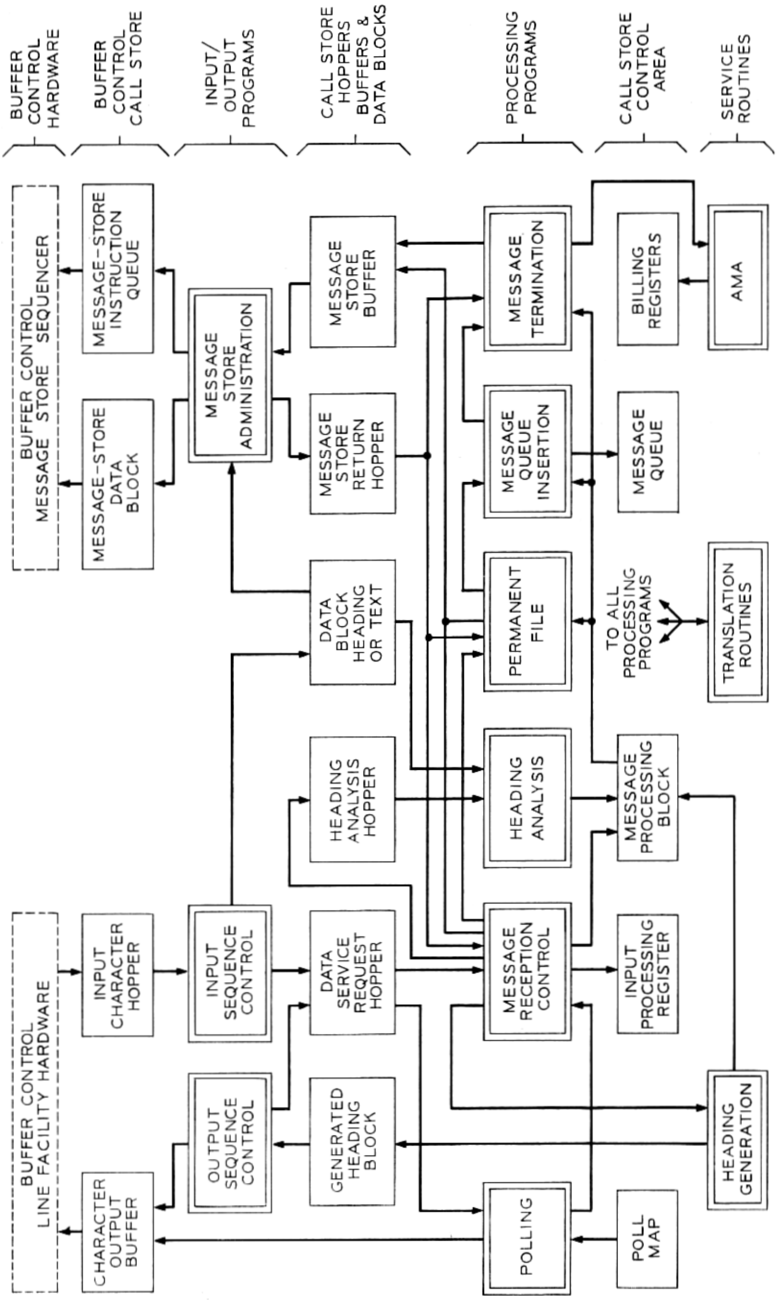


Fig. 16—Relationship between programs and memory areas concerned with the input-message processing.

sequence of control characters triggered the station to transmit its message to the switching center. The heading-analysis program was later entered when the complete heading information was received and stored on the disk. It proceeded to analyze the heading and build the message-processing block, establishing a terminator slot for each addressee in the heading. The message-reception control then guided the progress of the transmission until the end-of-message sequence was received. At that time, the message-reception control program divorced the message from the line, initiated further processing on the line, and relinquished further control of the message. The message-queue insertion program constructed a delivery request for each addressee listed in the message-processing block and added these requests to the appropriate message queues. When this task was completed, the message-termination program proceeded to update billing information and traffic-statistic counters, to transfer the call-store copy of the message-processing block to disk, and to release the call-store facilities.

6.7 *Message Nomination*

The message-nomination program is responsible for servicing the nomination map and initiating the transmissions to the terminating stations. This program is entered periodically by the executive control program and scans the nomination map. As mentioned earlier, a bit in this map, when set, corresponds to a data line that is available to receive traffic and in addition has messages waiting for delivery in its queue. The position of the bit in the nomination map uniquely defines the message queue by its corresponding queue index. With this unique identification, the linked list message queue is located through the message-queue table. The most eligible entry register is selected based on the priority and time ordered structure of the message queue. Entry registers which are marked as unavailable are passed over, i.e., the station intended for delivery is currently in a hold or a not-ready state. The poll-table index stored in the entry register uniquely identifies the station to which the message is to be delivered.

At this point the nomination program seizes an output-processing register which will follow the output processing through the remainder of the transmission. Using the system subroutines, output capacity to send the message is reserved and a data-block index is seized which reserves an output-work table entry for the transmission. The queue index and poll-table index are used as input data to the translation subroutines to retrieve the station related data which is loaded into the output-processing register. As a result of the translation process, the DLN of the desired station is identified. The DLN is used to inspect the line-status table to see that the line is in a state which is capable of receiving traffic. At this point the program knows that a delivery may be made to the given line; the selected entry register is removed from the message queue and released to the idle-link list. The nomination program then scans the entry registers in the message queue to determine if the given message is destined for more than one station on the given line. If an entry register in the message queue contains the address of the master register associated with the message currently being processed, the program will initiate delivery of the message to the recipient stations simultaneously. To facilitate this, a processing register annex is seized for each additional station and linked to the output-processing register. The poll-table index stored in each entry register is used to retrieve the station translation data (station call-in code, heading-format number, and station options), which are stored in the processing-register annex.

Once the output-processing register and all associated processing-register annexes have been loaded with the station translation data and the line-state checks have been made, the message-processing block for the message must be located. The associated master register indicates the location of the message-processing block: (*i*) in call store, (*ii*) in message store, (*iii*) in transit from call store to message store, or (*iv*) in transit from message store to call store. The message-processing block must be in call store before processing may begin. If the message-processing block is in transit to message store, a call-store copy of it still exists and the nomination program requests the transfer to stop by making a disconnect entry to the program making the transfer through the processing register whose address is stored in the master register. If the message-processing block is in transit to call store, it is because some processing register had requested the transfer. The nomination program uses the general purpose call-register timing routines to time until the transfer is complete. When the message-processing-block state indicates that it is residing in message store and not in a transit state, the nomination program makes use of the message-store-administration routines to transfer it to call store.

When the message-processing block is finally located in call store, the count of stations waiting delivery is decremented by a count equal to the number of stations which will receive the delivery; i.e., the number of processing register annexes plus the output processing register.

In addition, an item in the message-processing block is incremented to indicate that another program is currently using the message-processing block. As noted previously, this item is used by the message-termination program in its determination of the final disposition of the message-processing block. The message-processing block is checked to see if the delivery is a service message rather than a data message. If so, the service-message program is entered as a service routine to generate the appropriate service message based on data in the message-processing block.

The next task in the output-processing sequence is the generation of the personalized header information to be delivered to each of the stations involved in the delivery. This is the function of the heading-generation program which is discussed in greater detail in Section 7.2. In summary, the heading-generation program seizes a data block and loads it with the appropriate time, date, and message number for each station. This information is also loaded into the message-processing block. If the user has elected the proper heading options, the heading-generation program retrieves the originator's heading from disk and inserts the relevant address information into the data block. In addition, the program prefixes this information with a call-in code and control codes so that each station receives only its own generated-heading information. The heading-generation program transfers back to the nomination program which initiates the delivery by inserting the data-block address in the data-block address table and the character-output buffer address into the output-work table. The line-status table is updated to reflect that an output transmission is about to take place on the line by calling a subroutine in the message-transmission control program. The address of the output-processing register is placed in the line-status table.

6.8 *Message Transmission*

The message-transmission control program is responsible for transmitting the message to the line and retrieving data blocks from the message store as required. The output-sequence-control program acts in a subordinate role to disassemble data blocks and transmit characters to the line. The process is initiated when the output-sequence-control program detects the load-next-word output-control code in the data block associated with the output-work table entry (see Fig. 17). The data machine call-in sequence in the following word of the block is transmitted to the line and the program temporarily ceases transmitting to the line while waiting for the station response. In addition, an

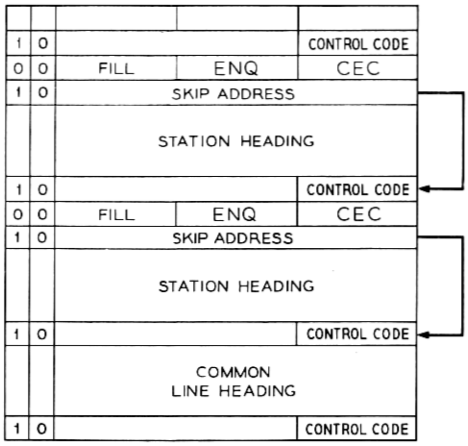


Fig. 17—Format of generated-heading data for a delivery to two stations on the same line.

entry is placed in the data-service-request hopper as a notification that the call-in sequence has been transmitted. When the data-service-request hopper is serviced, the message-transmission control program is informed of the transmission and accordingly updates the call status in the output-processing register and sets the response timer.

The data machine's response to the call-in sequence, its station identity code, is received by the input-sequence control program when it services the input-character hopper. The status bits in the word-status table direct the input-sequence-control program to load the response in the data-service-request hopper. When the data-service-request hopper is serviced, the response is analyzed by the message-transmission control program. If the response is not valid, the call-in sequence is repeated. If the response to this second attempt is also invalid, the message-transmission control program can use the skip address in the data block to skip over the station-heading information to transmit the call-in sequence for the next station on the line. In addition, the station-status table information is updated to reflect that the station is not ready, and a service message is transmitted to the transmission-plant craftsmen to indicate that the station has failed call-in. If the response to the call-in sequence is valid, the program updates the call state in the output-processing register. In addition, the program directs the output-sequence-control program to resume transmitting by advancing the data-block address in the data-block address table to the first word of the station-heading information.

The output-sequence-control program proceeds to transmit the station-heading information which may consist of relevant address, time, date, and message number based on the station options. The process continues with the transmission of further call-in sequences and station headings until all stations to receive the message have received their, and only their, particular heading information. An output-control code in the data block directs the output-sequence-control program to make a data-service request entry. This entry informs the message-transmission-control program that the call-in sequence is complete. The program then directs the output-sequence-control program to transmit the line heading to all stations. The line heading is common information received by all stations, such as the originator's time, date, and message number based on line options. The first portion of this data consists of a control character sequence which turns on all stations that had previously received the call-in sequence. The heading data which was generated by the heading generation program is terminated by a control code which indicates whether the line is to receive the originator's heading or only the text of the message. The data-block address is retained in the data-block address table for transmission of the message to the line. The supplementary control code in the data-block address table is set to a no-op state until the first-message block can be retrieved from disk and a data-service-request hopper entry is made.

When this entry is serviced by the message-transmission control program, it consults the message-processing block to determine the disk address of the first block of the link list of message blocks stored on the message store. Using the message-store service routines, the program transfers the first heading block to the call-store block retained in the data-block address table. If the line is not to receive the originator's heading but only the text, the message-transmission control program uses a special message-store administration routine to progressively read the link addresses in the link list of message blocks stored on disk until the address of the first text block is retrieved. The first text block is then transferred to the data block retained in the data-block address table. In accomplishing this task, the message-transmission control program uses the forward-link addresses to retrieve the data blocks and uses the reverse-link addresses as a check that a valid block has been retrieved from disk. When the program is notified of the successful transfer of the requested block through the message store return hopper, transmission to the line is initiated by updating the supplementary-control code in the data-block address table. This process continues until the complete message is transmitted to the line.

6.9 *End of Delivered Message*

The end of text control code in the last data block is detected by the output-sequence-control program. The data block is released and a data-service-request hopper entry is made. When the message-transmission control program receives this entry, it proceeds to confirm that each station which received the message is still functioning properly by transmitting a special character sequence, which is called roll-call, to each station. When these responses are received and validated, the program proceeds to divorce the message from the line and restore the line to the idle condition. The output-processing register address is removed from the line status table and the line status is set to the idle state. The output-work table slot and the output capacity are released. The message-transmission control program determines what program action should next take place on the line, based on the type of line and the poll-delivery criteria. Accordingly, the program will either set a bit in the polling map or the nomination map, thus completing the disconnect treatment on the line.

The remaining processing of the output message is similar to that for the input-message transmission. The permanent-file program is entered to update the station-retrieval tables stored on disk. The message-termination program is then called upon to update the billing information and the station and system traffic counters. If no other program is currently using the message-processing block and further deliveries of the message are required, the message-processing block is returned to disk. The call-store copy of the message-processing block and the output-processing register are released. If all deliveries of the message have been completed, the disk copy of the message-processing block is released and the generation of a permanent-file magnetic-tape record of the message, and a journal-file magnetic-tape record of the message transactions, are initiated. This is done by converting the message-processing block from a passive unit of memory to a processing register and entering this newly formed processing register on the permanent file hopper. The programs which perform the tape functions are discussed in an accompanying article in this issue.[2]

6.10 *Summary of Output Processing*

Figure 18 summarizes the sequencing of the programs involved in the output-message processing. The message-nomination program initiated the message delivery as a result of scanning the nomination map and brought the copy of the message-processing block into call store. The program then called upon the heading-generation program to construct the generated heading, consisting of the call-in sequence and the station heading for each station and the line heading. The response of each station to the call-in sequence was analyzed and the heading transmitted to each station. The message-transmission control program retrieved the message blocks and controlled the delivery until the end of the message. At this time, a special roll-call sequence was sent to each station to verify that the station had received the message.

After completion of the message delivery, the message-transmission control program restored the line to an idle state and initiated further processing on the line by setting either the poll map or the nomination map. The permanent file program was given control of the processing register to update the station-retrieval tables stored on disk. The message-termination program then ordered the appropriate billing logic and either returned the message processing block to disk or initiated the permanent file processing.

VII. SERVICE ROUTINES

7.1 *Translations*

The translation data base is located in the permanent magnet twistor program store. It contains all the necessary information to define the configuration of user's stations on lines, lines within a user network, and users' networks within a No. 1 ESS ADF office. Translation service routines are provided for use by the system programs to retrieve this information. The service routines eliminate the duplicate program code that would exist if each program requiring translation data retrieved it itself but, more importantly, they make the system programs insensitive to the organization of the data base. Each of the translation data structures discussed in Section 7.1.1 has associated with it at least one program subroutine which, when entered with the proper input data, will obtain from the data base the associated information. In many cases, more than one program subroutine is provided, some of which accept alternate forms of input information or derive output information from the data base for more than one translator.

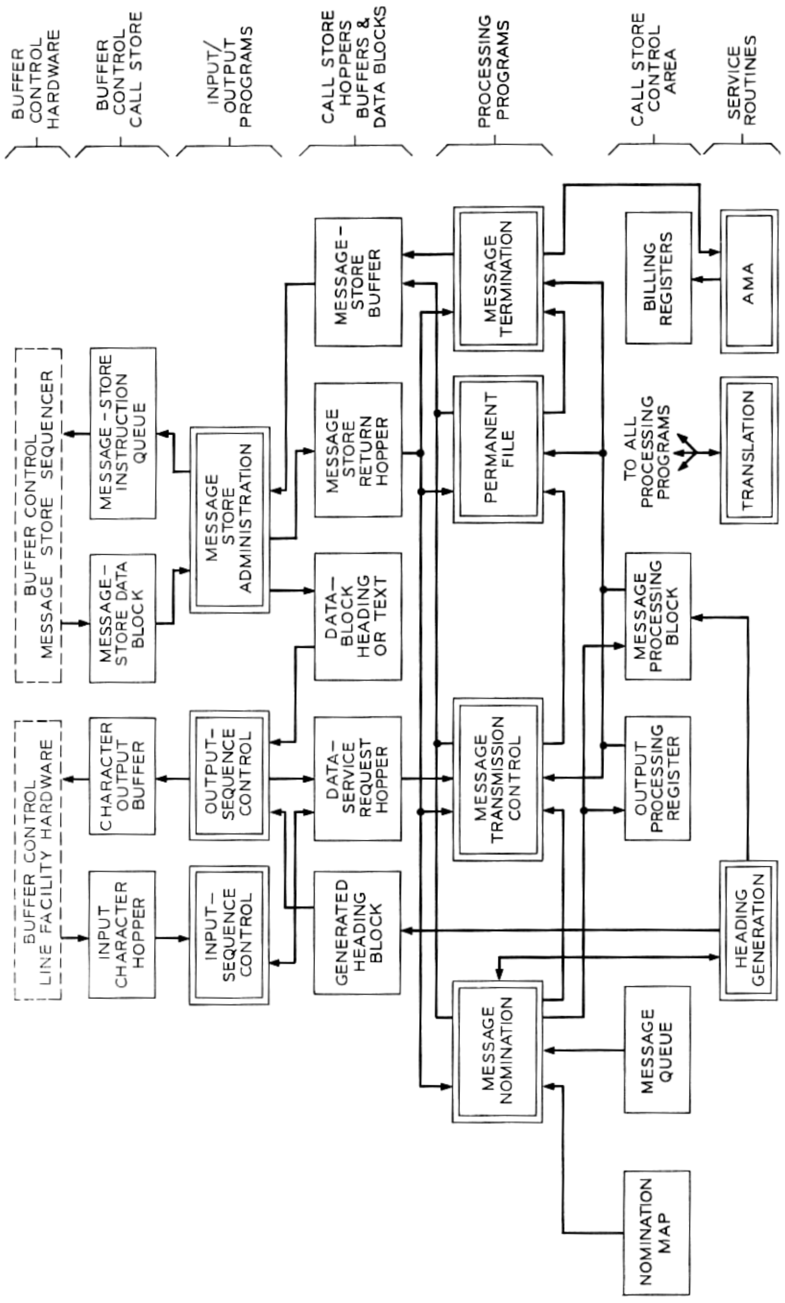


Fig. 18—Relationship between programs and memory areas concerned with the output-message processing.

Many of the basic concepts in the No. 1 ESS ADF translations are the same as those in No. 1 ESS:[4] the data base is in program store, changes are introduced via recent changes stored in call store, a card-writing process updates the program store, and after having been put into program store, the recent changes are removed from call store. Differences do exist, however, which are made possible and, in some cases necessary, by the differences in system organization and requirements. The relatively low-cost mass storage provided by the message store (disk) is used to provide a backup for the recent changes stored in call store; hence, errors introduced in the call store can be both detected and corrected.

The mnemonic translator is unique in that it utilizes a scatter technique to store its data. This technique makes the storage requirements and the recent-change program procedures for the mnemonic translator compatible with those of the other translators.

The following sections describe the contents and organization of the No. 1 ESS ADF translations, and the verification and audit procedures used to protect them.

7.1.1 *The No. 1 ESS ADF Translators*

7.1.1.1 *Basic Line Translation.* Each line in a No. 1 ESS ADF office is identified by a terminal equipment number known as a data-line number (DLN). This translator provides information on a per DLN basis. For a normally assigned line, the translator contains such information as a line class word, user-group identity, queue index, and a list of stations that are on the line.

The queue index can be used to locate the call store head cell for the queue of messages to be delivered to the line. The user identity is just that; it identifies the user network to which the line belongs. The line class word identifies such things as the speed of the line, the duplex nature (half or full) of the line, and the code (ASCII or Baudot) of the line. The list of stations on the line which this translator provides can be used to identify each of the stations on the line and to locate translators which identify the particular characteristics of each station.

7.1.1.2 *Basic Station Translation.* Information on each station in the system is contained in the basic station translator. The quantity used to locate a particular station within this translator is a station identity as obtained from the basic line translator. Each assigned station has the following information associated with it: normalized directory number, principal station mnemonic, alternate delivery directory number, station class word, call-enquiry code, heading-format identity, and privacy list. The normalized directory number is the directory number of the station, normalized to take advantage of the restricted set of directory numbers that is possible within an office. The station mnemonic is a 1-to-7 character alphanumeric that is used by the system to identify the station in any messages that might be initiated by the system and delivered to the user. The alternate delivery directory number is the directory number to which, upon request, the system will alternate deliver all messages addressed to the station. The station class word contains such information as the originating and terminating heading format options for the station, the originator's heading forwarding option, the originator's precedence level insertion, and the action-request authority of the station.

Each station has associated with it particular characters which are unique to that station on the line; that is, they may be duplicated between stations on different lines but not between stations on the same line. These characters are used for control purposes in polling the station for traffic and calling the station in to deliver messages. The special characters for calling the station in to deliver messages (call-enquiry code) are contained in this translater.

Because a single ADF may serve the networks for more than one user, certain privacy facilities are provided. Each station has associated with it a privacy-screening code. This code may indicate: (*i*) that this station can receive messages only from other stations within its own user's network; or (*ii*) that this station can receive messages from any station regardless of its user affiliation; or (*iii*) that this station can receive messages from any station within its own user's network and, in addition, from any station within other selected users' networks; or (*iv*) that this station can receive messages from any station within its own user's network and, in addition, from any station identified in a list of directory numbers which is provided; or (*v*) that this station can receive messages from any station within its own user network and, in addition, from a combination of *iii* and *iv*.

A heading-format identifier is provided for each station. The identifier assigned to a particular station is dependent upon the code (ASCII or Baudot) of the station and upon the generated-heading format desired by the station. It is this identifier modified by options contained in the station class word that determines the format of the originating and terminating generated headings received by the station.

7.1.1.3 *Heading Format Translation.* This translator drives the heading-generation program (Section 7.2) in constructing the originating and terminating generated headings delivered to stations upon the origination and receipt of messages. The particular heading-format table used is dependent upon the heading-format identifier associated with the station in the basic station translator, and upon whether the heading desired is for an originating message, and intraline terminating message, or an interline terminating message. Each table is a list of fixed data and subroutine calls. The order and contents of the information in this table determine the order and contents of the generated heading.

7.1.1.4 *Mnemonic Translation.* All messages addressed by users in the No. 1 ESS ADF system are addressed using 1-to-7 character alphanumeric mnemonics. This translator is used by the heading-analysis program to convert mnemonics in a message heading to the appropriate directory number or other information. A mnemonic may translate to:

(*i*) The directory number of a station.
(*ii*) A list of directory numbers. The message should be sent to all of these stations.
(*iii*) A DLN. This identifies a particular line and only has application in an action request.
(*iv*) A list of DLNs. This identifies a list of lines and only has application in an action request.
(*v*) Action-request indication. This identifies a message as one addressed to the No. 1 ESS ADF from a user's station. Such messages are known as action requests.
(*vi*) Message delivery precedence. This indicates that all deliveries, as a result of mnemonics in the message heading following this mnemonic, should be made with the precedence (one through four) to which this mnemonic translates.

Each user must define its own set of mnemonics; two or more users using the same mnemonic must each define it in the way that it is to be used by that user.

Defined as a special class of mnemonics in this translator are the mnemonics used to identify specific action requests. Each of these mnemonics translates to an index that is used by the program to determine the action to be performed. These mnemonics are defined only once, can be used by all users, and do not restrict the user's freedom to select mnemonics for his own network.

It is possible for a No. 1 ESS ADF office to have from one to 511 user networks. Every mnemonic requires three words of storage and each user may have from a few to many thousand mnemonics defined in translations. The input to this translator (user identity plus mnemonic) is 51 bits. The special memory organization problems of the mnemonic translator are discussed in Section 7.1.2.

7.1.1.5 *Basic Directory Number Translation.* This translator is organized according to the normalized directory number. For each assigned directory number it provides the identity of a station and of the line on which the station resides.

This translator also provides a cross-reference file address. The cross-reference file is an area on the message store where a record of all originations and terminations is kept for message-retrieval purposes. Within the cross-reference file area, each station has a dedicated address which is provided by this translator.

7.1.1.6 *Basic Queue Translation.* This translator is organized according to the message-queue index. For each assigned message queue index it provides the associated DLN. Its purpose is to provide a means of translating between bits in matrices maintained for polling and message queuing and the associated DLN. In the case of multiline hunting groups, this translator provides a list of all of the DLNs associated with the hunt group.

7.1.1.7 *User Control Locations Translation.* Each user can assign certain stations within its network to be used for administrative control purposes. These stations are collectively known as the user's control location. This translator provides a list of the directory numbers associated with these stations.

7.1.1.8 *Report List Translation.* Each user segments his network into, at most, seven disjoint subsets of lines. It is possible for each of the user control location stations to maintain administrative control over one or more of these subsets. In addition, certain automatic traffic reports are generated by the system on a subset basis and are sent to these control stations. This translator provides a list of all of the data lines and multiline hunt groups in each subset for each user.

7.1.2 *Memory Organization*

Figure 19 illustrates the standard head table-subtable-auxiliary block structure of a translator. The input quantity is divided into two parts, a selector and a level. The selector is used to index the head

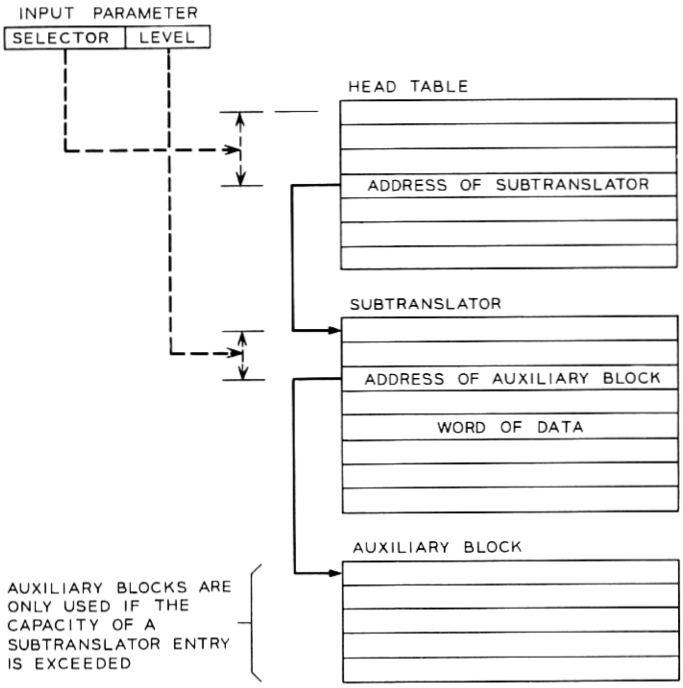

INPUT PARAMETER
SELECTOR
LEVEL
HEAD TABLE
ADDRESS OF SUBTRANSLATOR
SUBTRANSLATOR
ADDRESS OF AUXILIARY BLOCK
WORD OF DATA
AUXILIARY BLOCK
AUXILIARY BLOCKS ARE ONLY USED IF THE CAPACITY OF A SUBTRANSLATOR ENTRY IS EXCEEDED

Fig. 19—Head table—subtranslator—auxiliary block organization of a translator.

table and the level to index the subtable. The subtable contains the required data or the address of an auxiliary block which contains the data. Every position in a subtable uniquely defines a specific input to the translator and, in a fully expanded translator, every input has a position in a subtranslator.

The mnemonic translator presents a problem in that its input information is a 51-bit quantity (9-bit user identity and 42 bits of ASCII alphanumeric characters). The 9-bit user identity may or may not be a densely packed set; clearly, the 42 bits of ASCII alphanumeric characters are very sparsely packed. A compression function is used to generate a 13-bit pseudorandom number from the 42 bits of alphanumeric characters. To this is added the 9-bit user identity. The 13 low-order bits of the sum are a 13-bit index used as input to a translator with the head table-subtable-auxiliary block organization (see Fig. 20).

The number of mnemonics that are defined in an office will usually exceed 8192 (2^{13}). Because of this, more than one 51-bit input will

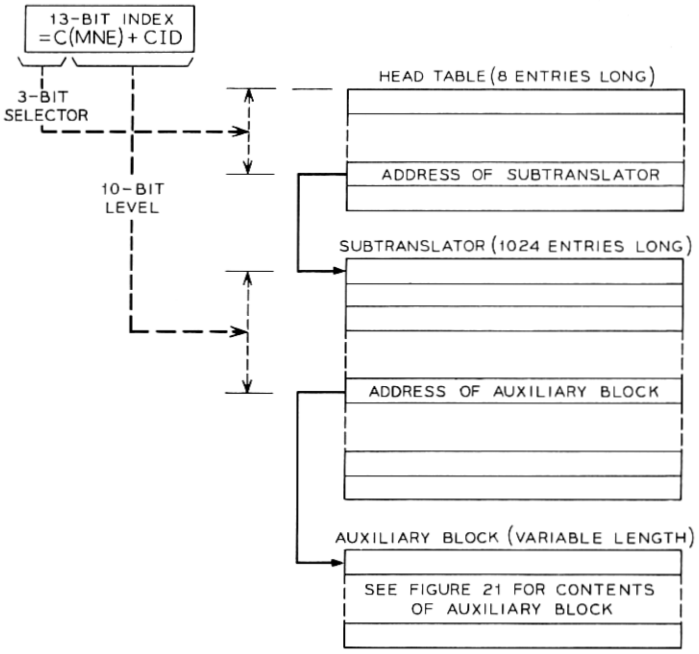

13-BIT INDEX =C(MNE)+CID
3-BIT SELECTOR
10-BIT LEVEL
HEAD TABLE (8 ENTRIES LONG)
ADDRESS OF SUBTRANSLATOR
SUBTRANSLATOR (1024 ENTRIES LONG)
ADDRESS OF AUXILIARY BLOCK
AUXILIARY BLOCK (VARIABLE LENGTH)
SEE FIGURE 21 FOR CONTENTS OF AUXILIARY BLOCK

Fig. 20—Organization of mnemonic translator. C(MNE) = 13-bit pseudo random number generated by the compression function acting on 7 ASCII characters. CID = 9-bit user identity.

result in the same 13-bit index. These "collisions" are resolved by using an auxiliary block to store the data for all of the mnemonics with the same 13-bit index. Because the 13-bit index is not unique to a mnemonic, it is necessary to store the mnemonics in the auxiliary blocks along with their associated data (Fig. 21). Having found the proper auxiliary block by using the 13-bit index, the data is identified by doing a binary hunt over the mnemonics stored in the auxiliary block. By making the subtables longer than 2^9 (maximum number of users) it is not necessary to store the 9-bit user-group identity in the auxiliary block. This is so because the same mnemonic for two different users cannot fall in the same auxiliary block.

The mnemonic translator uses one eight-word head table and up to eight 1024-word subtables. In order to conserve translation program store space, the eight subtables are on the left side of program store*

* Words in program store have a 14-bit left-side portion and a 23-bit right-side portion. Most translation data is stored on the right side.

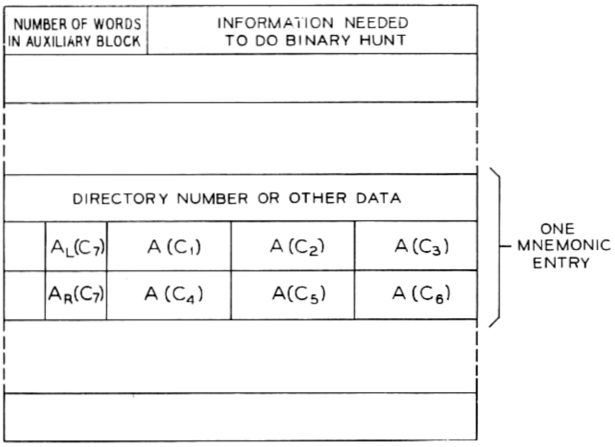

NUMBER OF WORDS IN AUXILIARY BLOCK
INFORMATION NEEDED TO DO BINARY HUNT
DIRECTORY NUMBER OR OTHER DATA
$A_L(C_7)$
$A(C_1)$
$A(C_2)$
$A(C_3)$
$A_R(C_7)$
$A(C_4)$
$A(C_5)$
$A(C_6)$
ONE MNEMONIC ENTRY

Fig. 21—Auxiliary block for mnemonic translator.
$C_1C_2C_3C_4C_5C_6C_7$ = 7 character alphanumeric mnemonic.
$A(C_i)$ = 6-bit stripped ASCII representation of the alphanumeric character C_i
$A_L(C_i)$ = 3 high-order bits of $A(C_i)$
$A_R(C_i)$ = 3 low-order bits of $A(C_i)$.
Entries are ordered according to the value of the 21-bit number $A_L(C_7)A(C_1)A(C_2)A(C_3)$. This allows a binary search to be used to locate mnemonics when there are many entries in the auxiliary block.

which would otherwise be "wasted." Since only one mnemonic translation is done per terminator, per message, the real-time penalty is not great. The head table and the auxiliary blocks for the mnemonic translator are on the right side of program store, as they are for most other translators.

The "scatter storage" organization for the mnemonic translator described above is no more costly in translation space than other schemes that were considered. Its primary feature is that, even considering imperfections in the compression function (unused values of the 13-bit index) the mnemonics are distributed over a large number of auxiliary blocks. The sizes of these auxiliary blocks are of the same magnitude as the sizes of auxiliary blocks for other translators. Hence, no special memory areas or special recent change techniques are necessary to administer the mnemonic translator.

7.1.3 *Recent Change Procedures*

A primary requirement is that the translation data base be readily changeable. This is accomplished by entering changes for the data base into call store via recent-change service orders, and requiring the translation service routines to hunt the call store for recent changes to the data base before returning the requested information to their clients. No. 1 ESS ADF uses a field-oriented input format for recent changes from both the service order and maintenance teletypewriters. All recent changes are active immediately upon entry into the system. Certain translators that are infrequently changed are not checked for recent changes by their retrieval routine. In these cases, the new data will not be used until the program store is updated.

The card-writing procedure for updating the permanent magnet twistor cards in program store is basically the same as in the No. 1 ESS.[4]

7.1.4 *Translation Memory Protection*

Errors that are introduced into the call store recent-change area and that go undetected are eventually transcribed into the program store. It is possible for these errors to propagate and do extensive damage to the translation data base before being discovered. In an effort to guard against this problem, three audits as well as a backup for the recent-change call store are provided. The audits verify the format and order of the recent-change call store, cross-check the translation data for consistency, and verify that each word in the translation program store is used in one and only one translator. The backup for the recent-change call store is a copy on the message store of all permanent recent changes.

7.1.4.1 *Translations Audits.* The recent-change call store audit validates the format of the recent-change call store area. It checks counters and pointers used to administer the recent-change call store as well as checking the address order of the recent changes themselves.

As a part of the recent-change verification program, an audit is provided which cross-checks much of the operational translation data for consistency. This audit is initiated via a teletypewriter request and is driven from the mnemonic translation table. The translation data associated with all lines and stations which have mnemonics associated with them are cross-checked for consistency. While this is not a 100-percent consistency check of the translation area, it does check a large portion of the data. This verification will print out the inconsistencies and errors that it finds. Not all detected errors are by any means indicative of destruction in the translation area. In most cases, they are oversights on the part of the recent-change personnel and are correctable with additional recent changes.

Every station and line in the system is not required to have a mnemonic associated with it in the mnemonic translator. It is desirable, however, if for no other reason than to enhance the capability of this audit.* Another audit, which is also a part of the translation verification program, can be called via a teletypewriter request to verify that all lines and stations do have a mnemonic in the mnemonic translator. It can be used to identify those lines or stations for which mnemonics are missing.

A basic sanity test for the translation program store is that each word is used once, and only once, and that all addresses in all head tables and subtranslators are within the translation program store range. This audit is a part of the translation-verification program and is called via a teletypewriter input message.

The program seizes auxiliary recent-change call store area to keep a busy-idle map for the address range of store being checked. Parameters are used to both define the address range of the translation program stores and the address ranges of subsets into which it is divided. A minimum of two subsets, defining the right and left sides of program store, is required. The busy-idle map is constructed for one subset at a time; hence, the subsets can be sized so that the busy-idle map will fit in the auxiliary recent-change area.

Errors found by this audit are printed out on the requesting teletypewriter. Each printout identifies the kind of error, the translator in which it was found, and the index into the translator when the error was detected.

7.1.4.2 *Message Store Backup for Recent Change Call Store.* In an effort to provide protection for the recent-change call store area and at the same time to take advantage of the storage media available in No. 1 ESS ADF, the message store (disk) is used as a backup for the recent-change primary and auxiliary call store areas. Input messages are provided which can be used from either the maintenance or the service-order teletypewriters to cause the permanent recent-change information in the recent-change call store to be copied onto the disk. Temporary recent changes are not copied onto the message store.

In response to an input from either the maintenance teletypewriter or from the service-order teletypewriter, the permanent recent-change information in the call store recent-change area is audited from the

* From the user's viewpoint, even stations and lines which never receive messages (send-only stations) should have mnemonics in order to use all the avaliable action requests.

information on the disk. The call store information is updated to agree with the information on the disk.

Two copies of the recent-change call store are maintained on the disk. When a request to copy the call store onto the disk is received, it is written onto the oldest copy. In this manner, should something happen to upset the copying process, the previous copy is not destroyed.

The recent-change information on the message store is protected by a check sum over 32-word segments. Should the check sum fail over a particular segment, that segment will not be used to change any call store information.

The procedure for inserting recent changes into the No. 1 ESS ADF requires that first, the recent-change call store be audited from the disk. The second step is to insert the recent changes into the call store. Thirdly, after appropriate verifications, the recent-change area should be copied onto the disk. At that point, the updated call store information is on the disk and, should the call store be destroyed, it can be reconstructed from the disk. Periodically (every hour) the recent-change call store is audited from the disk. The disk audit is also called automatically under certain conditions where, during the insertion of recent changes, the recent-change program discovers an impossible or unrecoverable situation. This allows the system to back up to the point where it was before the recent change was inserted. Protection against copying bad recent-change call store information onto the disk is provided by automatically changing any request for a copy into a request for an audit if there has been a maintenance interrupt since the last audit was run, or if any of the call store recent-change audits have discovered discrepancies.

7.2 *Heading Generation*

The heading generating service routines are used by the message reception control program upon the start of a message into No. 1 ESS ADF, and by the message nomination program when the delivery of a message to a station is initiated. These routines construct the generated heading by placing the appropriate characters and output control codes in data blocks in call store. The address of the first block in the linked list of blocks is returned to the calling program. That program proceeds to cause the generated heading to be sent to the station via the output-sequence-control program.

The format of an originating page copy is shown in Fig. 22. The originating message number, date and time are known as the generated heading. Figure 23 shows a typical terminating message. Its

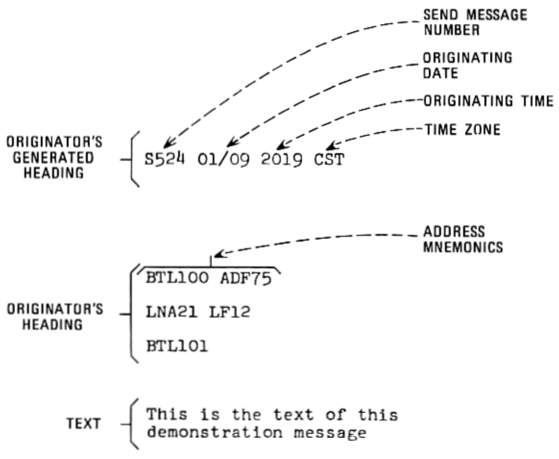


Fig. 22—Originator's page copy.

generated heading consists of the relevant address, the terminator's receive-message number, date and time, and the originator's send-message number, date and time. In addition, in the case of a termination, the generated heading contains those nonprinting character sequences necessary to call the station in for delivery.

There is great flexibility as to what a particular station is able to have in its generated headings. Optionally, a station need not receive all of the items shown in Figs. 22 and 23. Although it is not part of the generated heading, the originator's heading is an integral part of the heading options available to a terminating station. The terminating station is free to receive or not to receive the originator's heading, unless the originator specifically prohibits it via his translation option.

In addition to allowing a station the option of receiving those components of the generated heading shown in Figs. 22 and 23, and in allowing a station to choose whether or not to receive the originator's heading, it is possible for a station to specify additional fixed sequences of characters which are to be sent to it as a part of its originating or terminating generated heading. Also, it is possible for a station upon termination of a message to elect to receive a sequence of characters, if and only if, the address mnemonic with which the station was addressed was followed by a "+" symbol. A different sequence of characters can be sent to the station, if and only if, the

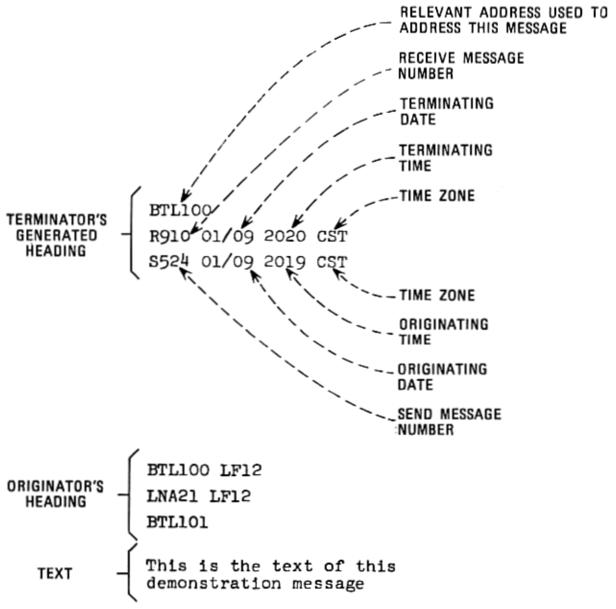


Fig. 23—Terminator's page copy.

address mnemonic with which the message was addressed was followed by a "*" symbol.[†]

In addition to optionally receiving the preceding components of the generated heading, it is possible for a station to receive them in nearly any order. The basic station translator contains a station class word, as well as a heading-format identifier. The station class word contains a series of bits to indicate the selection and nonselection of various items in the generated heading. The heading-format identifier is used to locate in the heading-format translator a data table that contains a list of fixed data and subroutine calls. The entries in the heading-format data table (HFDT), as well as their order, and the values of the bits in the station class word are used by the heading-generation program to determine the format for the generated heading.

The layout of entries in a HFDT is illustrated in Fig. 24. As an example, consider the HFDT shown in Fig. 25. This HFDT is for an ASCII HDX station which receives originator's message number, date and time, and time zone as its originating generated heading. By

[†] The "*" and "+" suffixes are used to cause sequences of characters to be sent to an addressee for the purpose of controlling auxiliary devices on the station equipment.

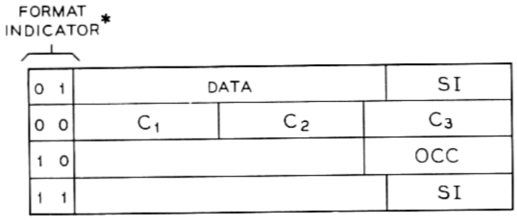


Fig. 24—Heading format data table layout.
SI Subroutine Identity—Control is transferred to the subroutine identified by this index.
DATA Certain subroutines need additional data such as the identity of a subformat table to use or the number of entries in the HFDT to be skipped based on a conditional test.
C_1, C_2, C_3 Teletypewriter characters properly coded such that they can be sent to the station exactly as they appear in this table. C_1 is sent first, C_3, last.
OCC Output control code—a code used to give information to the output-sequence-control program.

* The format indicators 01 and 11 are equivalent except that 11 indicates the last entry in the HFDT.

changing entry number 4 in the HFDT in Fig. 25, the HFDT shown in Fig. 26 results. This HFDT will result in the originator's send message number only in the originating generated heading.

Certain entries in the HFDT are required at fixed locations. For example, entry number 1 in Fig. 25 must always be first and entries 6, 7, and 8 must always be in that order and be last in the HFDT. This is because they specify necessary control characters for the station and a control routine for the heading-generation program. Between these entries, however, any fixed data may be specified and any subroutines which are appropriate may be called.

HFDTs for terminating generated headings are in general longer and more complex than for originating generated headings. They are, however, constructed in exactly the same way as originating HFDTs.

The heading-generation program is table driven from data in translations that can be changed via recent changes. This, in effect, allows the operating company the limited ability to program the machine in order to provide generated headings which contain, in almost any order, the date and time of transmission, message number, the relevant address on deliveries, and conditional and nonconditional printing and nonprinting fixed character sequences. Up to 128 different HFDTs can be defined for each of the three types: originating, intraline terminating, and interline terminating. Additional variations can be obtained by the settings of the items in the station class word.

WORD NO.					
1	0 0	FILL	DC1	CARRIAGE RETURN	ASCII TTY CHARACTERS
2	0 0	LINE FEED	DELETE	S	ASCII TTY CHARACTERS
3	0 1	CODE TABLE INDEX = 0		SI = 4	SUBROUTINE : OBTAIN MN; ENTER IN MPB AND GH
4	0 1	SUBFORMAT TABLE INDEX = 0		SI = 1	SUBROUTINE : OBTAIN D/T; ENTER IN MPB AND GH
5	0 0	CARRIAGE RETURN	LINE FEED	DELETE	ASCII TTY CHARACTERS
6	0 0	FILL	START OF TEXT	FILL	ASCII TTY CHARACTERS
7	1 0			OCC = 32	OCC: START OF TEXT SENT
8	1 1			SI = 12	SUBROUTINE : CLEAN UP

THIS GENERATED
HEADING CONSISTS OF : ORIGINATOR'S MESSAGE NUMBER
ORIGINATING DATE, TIME, TIME ZONE

EXAMPLE : S153 01/25 1150 CST

Fig. 25—Example of an originating HFDT for an ASCII half-duplex station. TTY, teletypewriter; MN, message number; MPB, message-processing block; GH, generated heading; and D/T, date and time.

VIII. ACTION REQUESTS AND SERVICE MESSAGES

An action request is a message from a station to the No. 1 ESS ADF. A service message is a message from the No. 1 ESS ADF to a station. As the name implies, an action request is used to request that the No. 1 ESS ADF perform some action. That action may be to collect and deliver certain data to a station, or it may be to alter the flow of messages through the network. Service messages are of two types. The first type is a message (queued service message) generated by the No. 1 ESS ADF in response to an action request, a reportable occurrence within the network, or according to a schedule. The second type (on-line service message) is a service message inserted into or appended onto a regular customer's message, either originating or terminating. Action requests and service messages are the tools which enable the telephone company (TELCO) to maintain administrative control over the entire network.

8.1 *Action Requests*

Figure 27 shows the action-request format. It should be noted that the entire action request is in the heading of the message. The heading-analysis program analyzes the mnemonics in the action request and

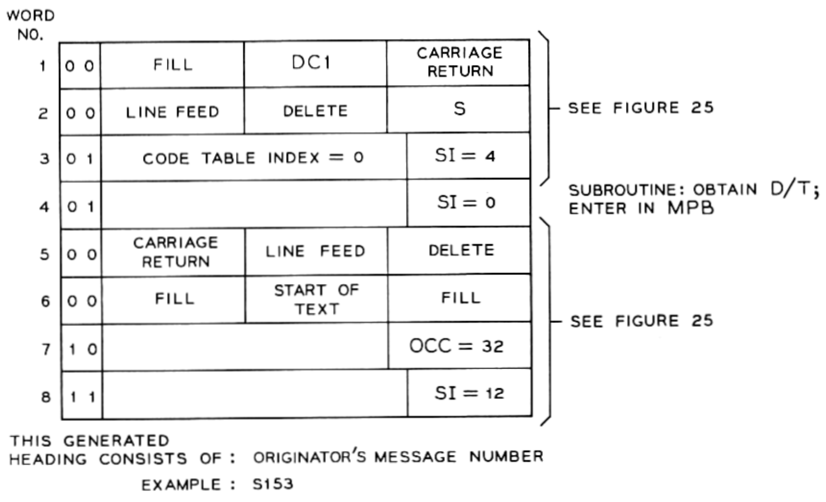

WORD NO.
1 | 0 0 | FILL | DC1 | CARRIAGE RETURN
2 | 0 0 | LINE FEED | DELETE | S
3 | 0 1 | CODE TABLE INDEX = 0 | SI = 4
4 | 0 1 | | SI = 0
5 | 0 0 | CARRIAGE RETURN | LINE FEED | DELETE
6 | 0 0 | FILL | START OF TEXT | FILL
7 | 1 0 | | OCC = 32
8 | 1 1 | | SI = 12
SEE FIGURE 25
SUBROUTINE: OBTAIN D/T; ENTER IN MPB
SEE FIGURE 25
THIS GENERATED HEADING CONSISTS OF : ORIGINATOR'S MESSAGE NUMBER
EXAMPLE : S153

Fig. 26—Example of an originating HFDT for an ASCII half-duplex station. D/T, data and time; MPB, message-processing block.

sets up a message-processing block to indicate the contents of the request. The various action-request programs then do further analysis and validity checking and perform the requested action.

Figure 28 is a partial list of the action requests that it is possible to input to the No. 1 ESS ADF. It should be noted that not all action requests can be submitted from all stations. Certain ones are reserved for TELCO use or for use by selected users' stations which maintain administrative control over the users' networks. These stations are known as the user-control locations.

The scope of the action requests is of particular note. Using them, it is possible to: stop message origination from a station (put the station on SKIP); stop message delivery to a station and hold the messages in a queue (put the station on HOLD); stop message delivery to a station and alternate deliver the message to some other station (put the station on ALT). The preceding action requests all affect the flow of messages through the network. In addition, it is possible to request data from the No. 1 ESS ADF. In particular, it is possible to request a report indicating the number of messages queued for a line and to request the status of a line to determine the stations that are on SKIP, HOLD, or ALT, or that are NOT-READY to receive (e.g., have low paper).

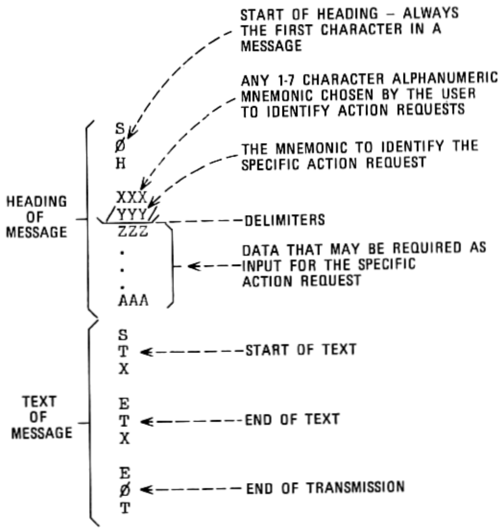

START OF HEADING – ALWAYS THE FIRST CHARACTER IN A MESSAGE
ANY 1-7 CHARACTER ALPHANUMERIC MNEMONIC CHOSEN BY THE USER TO IDENTIFY ACTION REQUESTS
THE MNEMONIC TO IDENTIFY THE SPECIFIC ACTION REQUEST
S Ø H
XXX
/YYY/
ZZZ
AAA
DELIMITERS
DATA THAT MAY BE REQUIRED AS INPUT FOR THE SPECIFIC ACTION REQUEST
HEADING OF MESSAGE
S T X
START OF TEXT
TEXT OF MESSAGE
E T X
END OF TEXT
E Ø T
END OF TRANSMISSION

Fig. 27—Action-request format.

8.2 *Queued Service Messages*

8.2.1 *Service Messages in Response to Action Requests*

Most action requests receive a direct reply in the form of a service message. Figure 29 shows typical responses; it should be noted that all action requests which affect the ability of a station to either originate or terminate messages stimulate two service messages to be generated. The first message is a notification to the affected station. The second service message is a notification to the user's administrative control center (user-control location) that some action affecting the station's ability to originate or terminate messages has been taken. Both of these service messages are generated regardless of whether the action is requested by the station itself, by the user-control location, or by TELCO. This enables the user-control location to stay abreast of the latest status of every station for which it is responsible and, hence, administer the flow of messages through the network. Other action requests stimulate service messages which are sent either to the action-request originator or to some preassigned user-control location station, depending upon the type of action request.

All examples of service messages, except those in Fig. 30, show only the printing characters. Figure 30 shows an example of a service message

ACTION REQUEST ORDER	NECESSARY INPUT DATA	USED BY	MEANING
SKIP	STA ID(S)	UCL, TELCO, CSTC	STOP POLLING STATION FOR ORIGINATING TRAFFIC
OFFSKIP	STA ID(S)	UCL, TELCO, CSTC	SUSPEND SKIP CONDITION
HOLD	STA ID(S)	A	STOP DELIVERING MESSAGES TO STATION – RETAIN IN QUEUE
OFFHOLD	STA ID(S)	A	SUSPEND HOLD CONDITION
ALT	STA ID(S) ALTERNATE STA ID	A	REROUTE MESSAGES TO ALTERNATE STATION
OFFALT	STA ID(S)	A	SUSPEND ALT CONDITION
Q Q LENGTH	LINE ID(S)	UCL TELCO, CSTC	SEND A SERVICE MESSAGE TO THE REQUESTING STATION INDICATING THE NUMBER OF MESSAGES IN THE QUEUE FOR EACH LINE
DIST	LINE ID	CSTC	MEASURE THE DISTORTION INCOMING ON THE LINE – REPORT TO REQUESTING STATION
STATUS	LINE ID	TELCO, CSTC	REPORT TO THE REQUESTING STATION THOSE STATIONS ON THE LINE THAT ARE ON SKIP, ON HOLD, ON ALT, OR IN A NOT READY CONDITION AND WHETHER OR NOT THE LINE IS BEING SERVED ON ITS NORMALLY ASSIGNED DATA LINE PORT

STA ID(S)	– STATION IDENTITY(S) – NON TELEPHONE COMPANY USERS USE MNEMONICS, THE TELEPHONE COMPANY USES DIRECTORY NUMBERS
LINE ID(S)	– LINE IDENTITY(S) – NON TELEPHONE COMPANY USERS USE MNEMONICS, THE TELEPHONE COMPANY USES DATA LINE NUMBERS
A	– ANY STATION
TELCO	– TELEPHONE COMPANY INTERCEPT CENTER, TRAFFIC DEPARTMENT
CSTC	– CONTROL SERVING TEST CENTER, PLANT DEPARTMENT
UCL	– USER-CONTROL LOCATION

NOTE: ONLY THE UCL, TELCO, AND CSTC CAN USE AN ACTION REQUEST TO PLACE A STATION, OTHER THAN THE REQUESTING STATION, ON HOLD OR ALT

Fig. 28—Partial list of action requests.

with some of its nonprinting characters. It should be noted that the sequence

$$\begin{array}{c}E\\S\\C\end{array}\begin{array}{c}\\S\\O\end{array}\ \cdots\ 01\text{–}01\ \cdots$$

and the sequence $\begin{array}{c}E\\S\\C\end{array}\begin{array}{c}\\S\\I\end{array}$ bracket the service message. This is for convenience of computer-type terminals; the $\begin{array}{c}E\\S\\C\end{array}\begin{array}{c}\\S\\O\end{array}$ sequence allows the computer

```
ACTION REQUEST                              SERVICE MESSAGE(S)
ACT            TO BTL100:                                      TO USER-CONTROL LOCATION:
/SKIP/              ...01-09...                                     ...01-10...
BTL100              THIS STATIØN ØN SKIP                            PLACED ØN SKIP
                                                                    BTL100

ACT            TO BTL100:                                      TO USER-CONTROL LOCATION.
/OFFSKIP/           ...01-11...                                     ...01-12...
BTL100              THIS STATIØN ØFF SKIP                           PLACED ØFF SKIP
                                                                    BTL100

ACT            TO THE REQUESTING STATION:
/Q/                 ...01-17...
L621                Q REPORT
L619                LINE          Q
                    L621          4
                    L619          5

ACT            TO THE REQUESTING STATION:
/STATUS/            STATUS 1602 OOTTY00077-619 NORMAL
1602                /CID/0002

                    ...01-14...
                    SKIP REPORT
                    BTL100

                    ...01-15...
                    HOLD REPORT
                    BTL100   ADF75

                    ...01-18...
                    NOT READY REPORT
                    LLIN2
```

Fig. 29—Typical service messages in response to action requests.

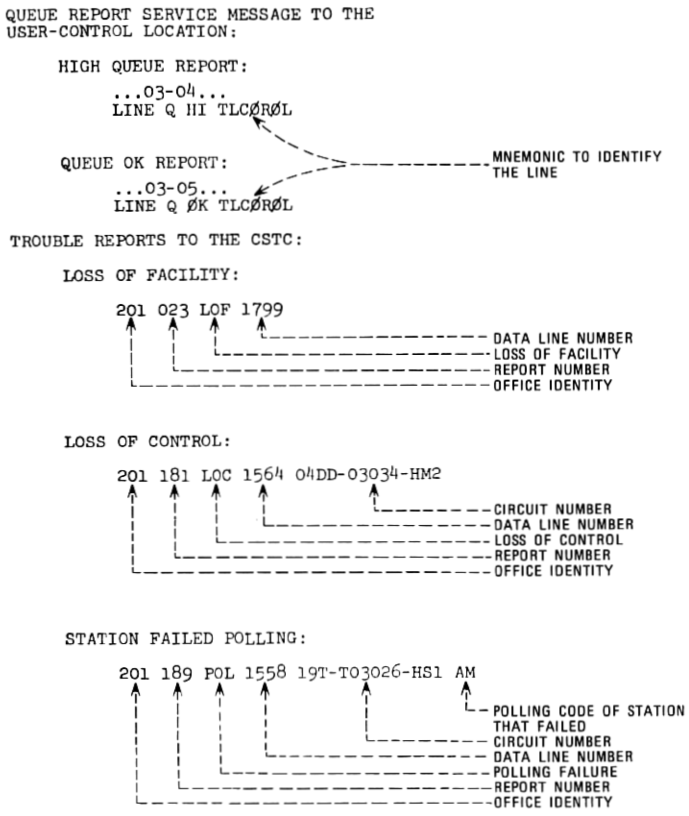
to recognize the beginning of a service message and to shift into an alternate mode where it may be properly handled; the number sequences allow easy identification; the $\begin{matrix} & E & \\ S & & S \\ C & & I \end{matrix}$ sequence terminates the service message.

8.2.2 *Service Messages in Response to System-Recognized Occurrences*

Typical service messages generated in response to system-recognized events are shown in Fig. 31. There are two types of these service messages. The first are the queue high-low reports. The queue-high service message notifies the user-control location that some predetermined threshold for the number of messages in the queue for a

```
NONPRINTING
 ___|___
  E
  S S
  C Ø ...01-01...                E
                                 S S
        THIS STATIØN ØN HØLD     C I
                                ___|___
                              NONPRINTING
```

Fig. 30—Service message showing nonprinting characters.



```
QUEUE REPORT SERVICE MESSAGE TO THE
USER-CONTROL LOCATION:

    HIGH QUEUE REPORT:
         ...03-04...
         LINE Q HI TLCØRØL

    QUEUE OK REPORT:
         ...03-05...
         LINE Q ØK TLCØRØL

TROUBLE REPORTS TO THE CSTC:

    LOSS OF FACILITY:
         201 023 LOF 1799

    LOSS OF CONTROL:
         201 181 LOC 1564 04DD-03034-HM2

    STATION FAILED POLLING:
         201 189 POL 1558 19T-T03026-HS1 AM
```

Fig. 31—Service messages in response to system-recognized occurrences.

data line has been exceeded. Its purpose is to alert the user-control location to some possible abnormal condition. A subsequent queue-low report is sent to the user-control location when the queue for the data line has returned to normal.

The second type of service messages in this category is the trouble report sent to TELCO at the Control Serving Test Center. These trouble reports notify TELCO of trouble conditions encountered in polling a station, originating traffic, or in delivering traffic to a station. They alert TELCO to problems with the user stations, many times before the user himself is aware of it. This is an aid to fast repair time with, of course, minimum down time for the user. It is an especially important feature when many of the user stations may be operating unattended.

8.2.3 *Periodic Service Messages*

As an aid in administering the network on a day-to-day basis and in collecting information to engineer the network, it is possible for both TELCO and for the user-control locations to receive scheduled service messages (see Fig. 32) from the No. 1 ESS ADF.

These service messages are of two types: status and traffic reports. The status report is generated and delivered according to some preselected hourly schedule to designated stations. Each station that receives a status report need not subscribe to the same hourly schedule. In addition, within certain restrictions, it is possible for each station that receives this report to receive information only concerning those stations in which it has an interest. Hence, it is possible for user-control location stations to receive periodic reports only on those parts of the user's network for which they have responsibility. The status report indicates all of those stations which are to be reported upon and which are on SKIP and hence cannot originate, or are on HOLD and hence cannot receive messages.

The second service message is the traffic report. As was the case

```
HOURLY STATUS REPORT
SERVICE MESSAGE:

      ...03-01...
      PERIODIC REPORT 02/02 1000 EST

      ...01-14...
      SKIP REPORT
      TRLC TRLB TRLF TRLE TRLD

      ...01-15...
      HOLD REPORT
      TRLB BTL100


DAILY TRAFFIC REPORT
SERVICE MESSAGE:

      ...03-03...
      DAILY TRAFFIC REPORT 02/02 1900 EST
      STATIONS          ORIGINATIONS                 TERMINATIONS
                    MESSAGES   CHARACTERS       MESSAGES    CHARACTERS

       54-BA             20       24030              2         10590
       53-UA              3        3540              2          3880
       52-CA              2        2880             26         75030
       MHG-1              0           0              0             0
       TRLC              50       75030              0             0
       TRLB               1         320              0             0
       TRLF               1         490              1         11790
       TRLE              35       48010              5           380
       TRLD               0           0            120         55820

               TOTALS   112      154300            156        157490
```

Fig. 32—Scheduled service messages.

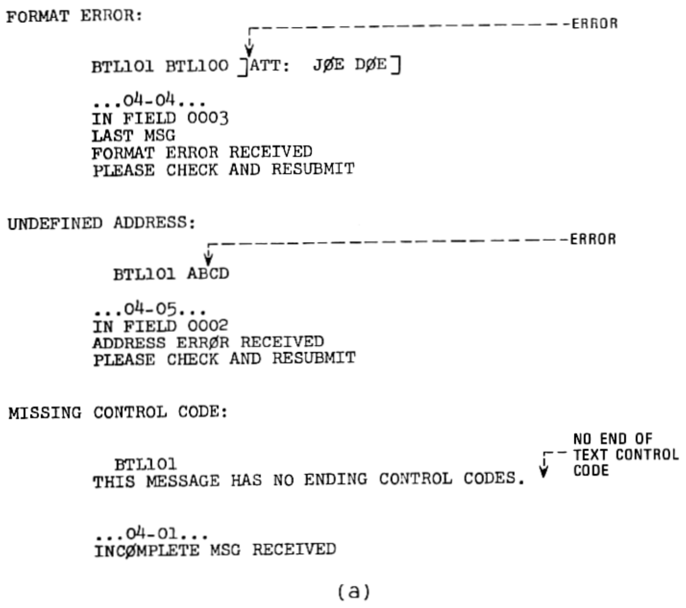
with the status reports, a traffic report may cover a subset of a user's set of stations. For each station included in the traffic report, counts of the number of messages originated and the number of messages terminated to the station and counts of the number of characters originated and the number of characters terminated to the station are given. The total counts are then included at the bottom of the report. The traffic report may be generated daily, monthly, or both. A monthly report may be generated on any day of the month that has been designated in translations. The daily report may be generated at any hour of the day that is specified in translations. If both reports are to be generated, the monthly report must be generated at the same hour of the day as the daily report.

8.3 *On-Line Service Messages*

These service messages are inserted into, or appended onto the end of the originating or terminating page copy on a user station. Their purpose is to alert the originator or the terminator of some peculiarity regarding either the originating or terminating transmission. Figures 33a and 33b illustrate typical service messages. Those service messages delivered on the originator's station interrupt the originating message. They indicate that for some reason the originating message may not be properly processed and delivered to all of the terminators.

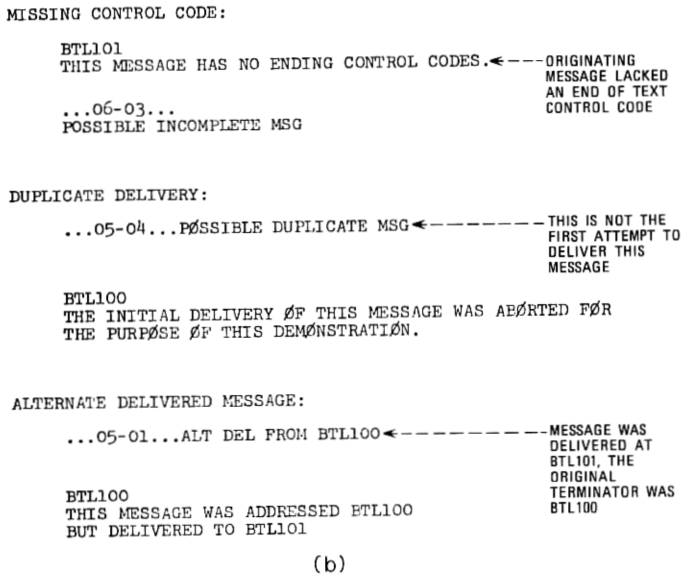
Those service messages inserted into the beginning of a user's terminating message indicate some abnormality in the delivery, but do not nullify its validity. For example, the service message inserted at the beginning of a message delivery may indicate that the message was originally addressed to some other station, but has been alternate delivered to the station where it is now being delivered. Or, the service message may indicate that this is a possible duplicate message. That is, a previous attempt was made to deliver the message either to this or some other station.

Those service messages appended to a terminating message may or may not indicate that the delivered message is valid. For example, the service message may indicate that the delivered message should be disregarded. It may, however, indicate that the originator started this message into the system but never sent an end-of-text (ETX) character. Hence, the No. 1 ESS ADF does not know whether the originator transmitted the entire message. In this case, the delivery is followed by a possible incomplete service message. Figures 33a and 33b show examples of the way these service messages may appear on the terminator's or originator's page copy.



```
FORMAT ERROR:
                                  ----------------------ERROR
        BTL101 BTL100 ]ATT:  JØE DØE]

        ...04-04...
        IN FIELD 0003
        LAST MSG
        FORMAT ERROR RECEIVED
        PLEASE CHECK AND RESUBMIT


UNDEFINED ADDRESS:
                               ----------------------ERROR
          BTL101 ABCD

        ...04-05...
        IN FIELD 0002
        ADDRESS ERRØR RECEIVED
        PLEASE CHECK AND RESUBMIT


MISSING CONTROL CODE:
                                                       NO END OF
          BTL101                                    -- TEXT CONTROL
        THIS MESSAGE HAS NO ENDING CONTROL CODES.      CODE

        ...04-01...
        INCØMPLETE MSG RECEIVED

                        (a)
```

Fig. 33a—Typical service messages inserted on-line on originator's page copy.



```
MISSING CONTROL CODE:

     BTL101
     THIS MESSAGE HAS NO ENDING CONTROL CODES.<---ORIGINATING
                                                  MESSAGE LACKED
                                                  AN END OF TEXT
     ...06-03...                                  CONTROL CODE
     POSSIBLE INCOMPLETE MSG


DUPLICATE DELIVERY:

     ...05-04...PØSSIBLE DUPLICATE MSG<--------THIS IS NOT THE
                                               FIRST ATTEMPT TO
                                               DELIVER THIS
                                               MESSAGE

     BTL100
     THE INITIAL DELIVERY ØF THIS MESSAGE WAS ABØRTED FØR
     THE PURPØSE ØF THIS DEMØNSTRATIØN.


ALTERNATE DELIVERED MESSAGE:

     ...05-01...ALT DEL FROM BTL100<---------MESSAGE WAS
                                             DELIVERED AT
                                             BTL101, THE
                                             ORIGINAL
     BTL100                                  TERMINATOR WAS
     THIS MESSAGE WAS ADDRESSED BTL100       BTL100
     BUT DELIVERED TO BTL101

                        (b)
```

Fig. 33b—Typical service messages inserted on-line on terminator's page copy.

8.4 *Service-Message Generation*

8.4.1 *Generation of Queued Service Messages*

All service messages, other than the on-line service messages, are delivered in a fashion as similar as possible to the manner in which regular messages are delivered. The program initiating a service message forms a message-processing block in which is indicated all of the terminators. This special message-processing block is then flagged as a service-message message-processing block and additional data needed for the generation of the appropriate service message is entered into it; in some cases, additional data may be placed on the message store. In these instances, the additional data in linked to the message-processing block in exactly the same fashion that the heading and text of a normal message are linked to it. The message-processing block is used to queue the service message for delivery in the same manner that any other message is queued for delivery. It should be noted that all of the data collection and generation necessary for the service message is done before the service message is queued for delivery. Also, all actions (e.g., placing a station on SKIP) are performed before the service message is queued.

The process of nominating a queued service message for delivery is the same as that for nominating any other message. Once the receiving station is called in, the message-transmission control program reads the first data block from the disk (if there is such data) and then enters the service-message generation program. This program, using data from the message-processing block and data which the message-transmission control program read from the disk (if there is such data), generates in call store a data block that contains the message that is to be transmitted. The service-message generation program returns to the message-transmission control program with this data block and the message-transmission control program uses it in exactly the same fashion that it would use a text block read from the message store for an ordinary message; the data block is sent to the station via output-sequence control.

After a complete data block has been transmitted to the station, the normal procedure is for the message-transmission control program to fetch the next data block from the message store. If this service message is one that has data blocks on message store, the procedure just described is followed. In any event, either immediately, or after having obtained the next data block from message store, the service-message generation program is again entered and it generates in call store a data block that contains the next characters to be sent out to the terminating station. In effect, the service-message generation program just replaces or supplements the logic used to pull the next text block from the message store. The service-message generation program's function is primarily one of reforming or expanding data which may be in the message-processing block or in data blocks on the message store.

8.4.2 *Generation of On-Line Service Messages*

On-line service messages are generated when, in the message-originating process, or in the message-delivery process, some program finds an error or an abnormal situation which requires that the station be notified in order to maintain system integrity. The mechanism for generating these service messages is basically independent of whether it is to be sent to an originator or to a terminator. In either case, the cognizant program enters the service-message generation program indicating the kind of service message that should be generated. The service-message program generates the necessary characters in data blocks in call store for the service message to be sent to the station. In the case of a service message appended to the end of a message, the service message is sent to the station and that terminates the transmission. In the case of a service message inserted at the beginning of a regular message, the data blocks containing the service message are incorporated into the generated heading.

IX. MESSAGE INTEGRITY FEATURES

It was recognized early in the development of No. 1 ESS ADF that hardware and software problems would occur that could affect the content and delivery of the users' messages. A message-switching system is especially sensitive to the effects of processing errors since the user is not supervising all phases of the delivery of his messages. For this reason, it is not sufficient to provide audit programs which only protect the capability of the system to transmit messages. It is also necessary that audit programs assist in accounting for all the messages stored in the system. The audit programs could not perform this message protection without the provision of operational procedures and memory layouts which facilitate message accounting.

As noted in the above description of the ADF operational program, certain in-line processing steps have been taken and the disk-memory assignment has been organized to facilitate accounting for all messages. The first step in the procedure is to store a preliminary copy of the first message-processing block on the disk at the time of the initial decision to start accepting the message into the system. At this point in the sequence of accepting a message, the first message-processing block contains the preliminary originator information consisting of the originator's identity, message number, and time and date. This is sufficient information with which to generate a service message indicating a system failure on the specific user's message. At a later time when the input message is complete, and further terminations are necessary but not in progress, the complete message-processing block is stored on the disk. In order to be able to distinguish between first message-processing blocks that are actively associated with messages and those that are idle, the blocks are written with an idle code when they are released. This step insures that the audit programs can recognize the existence of the proper message-queue stimulus for all those messages which have not been delivered to all terminators.

Several uses are made by the audits of the first message-processing block information as stored on the disk. The first use is the periodic search of the disk area in which the first message-processing blocks are stored. This is called the daily message audit, although it is performed twice daily. This audit checks each busy first message-processing block for a proper call-store stimulus. If a first message-processing block is marked preliminary, then there should be a call-store copy of the message-processing block. If it is marked complete, there should be a master register. If there is no call-store copy corresponding to the preliminary disk copy, then the program generates a service message to be sent to the TELCO position. This service message gives the information necessary to notify the originator of the specific message that may not have been delivered to all terminators by the system. These service messages are sent to the TELCO position primarily for two reasons. First, they are an indication of the grade of service being provided by the system. Second, this permits delivery, even if the user's station is in an out-of-service state. If there is no master register for a complete message-processing block, then the message-processing block is read into call store and new entries are made in the message queues for all undelivered terminators.

The second use of the first message-processing block information stored on the disk is for the message-recovery process after a severe system disturbance. The first message-processing block area on the disk is scanned for two purposes. One is to rebuild the busy-idle map of block usage on the disk, and the other is to requeue messages that are complete on the disk and to send service messages for incomplete messages. In this case, the service messages are sent to the originator. The reason for directing the service message to the originator instead of to the TELCO position is twofold. First, it is assumed that the message was incomplete because the originating message was interrupted by the emergency-action process, and the user is served better by an immediate notification. Secondly, the TELCO position would otherwise be flooded with a large number of messages.

The typical method of clearing the problem that caused a severe emergency-action phase is to obtain a system which operates in a sane manner by clearing the system's history and using initialization or restart procedures. In order to provide message protection and a continuity to the processing, it is necessary to save information through the system-recovery process. When selecting information to be saved, it is necessary to weigh the factors of data value, the sensitivity of the system to errors in the data, and the ability to correct errors in the data to be saved. In the No. 1 ESS ADF system, some of the information selectively saved through call-store zeroing are the message-number tables and station-status tables. Both of these are necessary to provide continuity of processing from the user's viewpoint and the system sensitivity to errors in these tables is quite low. The effect of an error in a message number should be no worse for the user than a zeroing of the message number. The station-status tables can be checked for invalid or mutually exclusive states and placed in a valid state if necessary. Periodic reports will indicate to the user the status of his stations and if incorrect states are noticed, the user may put the stations back into the desired state.

The combination of audits and operational defensive checks as used in the No. 1 ESS ADF provide a message-switching system with a high degree of message integrity. Since cutover of the system, a number of lost-message complaints from users have been analyzed using the message and journal file retrieval capabilities. Consistently, it has been found that the message was properly delivered and lost by the station attendant or that a service message concerning the given message was disregarded by the station attendant.

X. CONCLUSION

The No. 1 ESS ADF is a modern store-and-forward data-switching system. It is an extension of the No. 1 ESS hardware and program technology to provide message-switching service to data users. This paper has described in detail the operational programs and storage organization used to poll stations for originating traffic, to process the message transmissions to and from the stations, and to queue and store messages waiting to be delivered. Also, it described the unique characteristics of the program which provide for a high level of message integrity and provide the user with the ability to exercise administrative control over his network.

XI. ACKNOWLEDGMENT

The development described here represents the combined efforts of many people in Bell Telephone Laboratories. The authors wish to acknowledge their indebtedness to the members of the Data Communications Engineering Center for their efforts in defining the system requirements and their many helpful suggestions in the area of program strategy. The design of the operational program was accomplished jointly by two development laboratories which were never collocated. We wish to express our thanks to our colleagues in these laboratories for the unity of purpose and free communication which overcame the complex interfaces required between these two organizations.

REFERENCES

1. Harr, J. A., Hoover, Mrs. E. S., and Smith, R. B., "Organization of the No. 1 ESS Stored Program," B.S.T.J., *43,* No. 5, Part 1 (September 1964), pp. 1923–1959.
2. Ewin, J. C., and Giloth, P. K., "No. 1 ESS ADF: System Organization and Objectives," B.S.T.J., this issue, pp. 2733–2752.
3. Potter, J. L., Strebendt, Mrs. F. B., and Williams, J. R., "No. 1 ESS ADF: Magnetic Tape Subsystem," B.S.T.J., this issue, pp. 2915–2940.
4. Ulrich, W., and Vellenzer, Mrs. H. M., "Translations in the No. 1 Electronic Switching System," B.S.T.J., *43,* No. 5, Part 2 (September 1964), pp. 2533–2573.