MTS/3000
I.M.S.

Data and Image Systems

HP PRIVATE

Product No. 32193A

Karel Siler
March 25, 1978

SUMMARY

The Multipoint Terminal Software (subsequently called MTS or MTS/3000) is a data communication product which provides access to HP multipoint terminals by way of the device - independent MPE file system. MTS/3000 consists of four separately compiled modules:

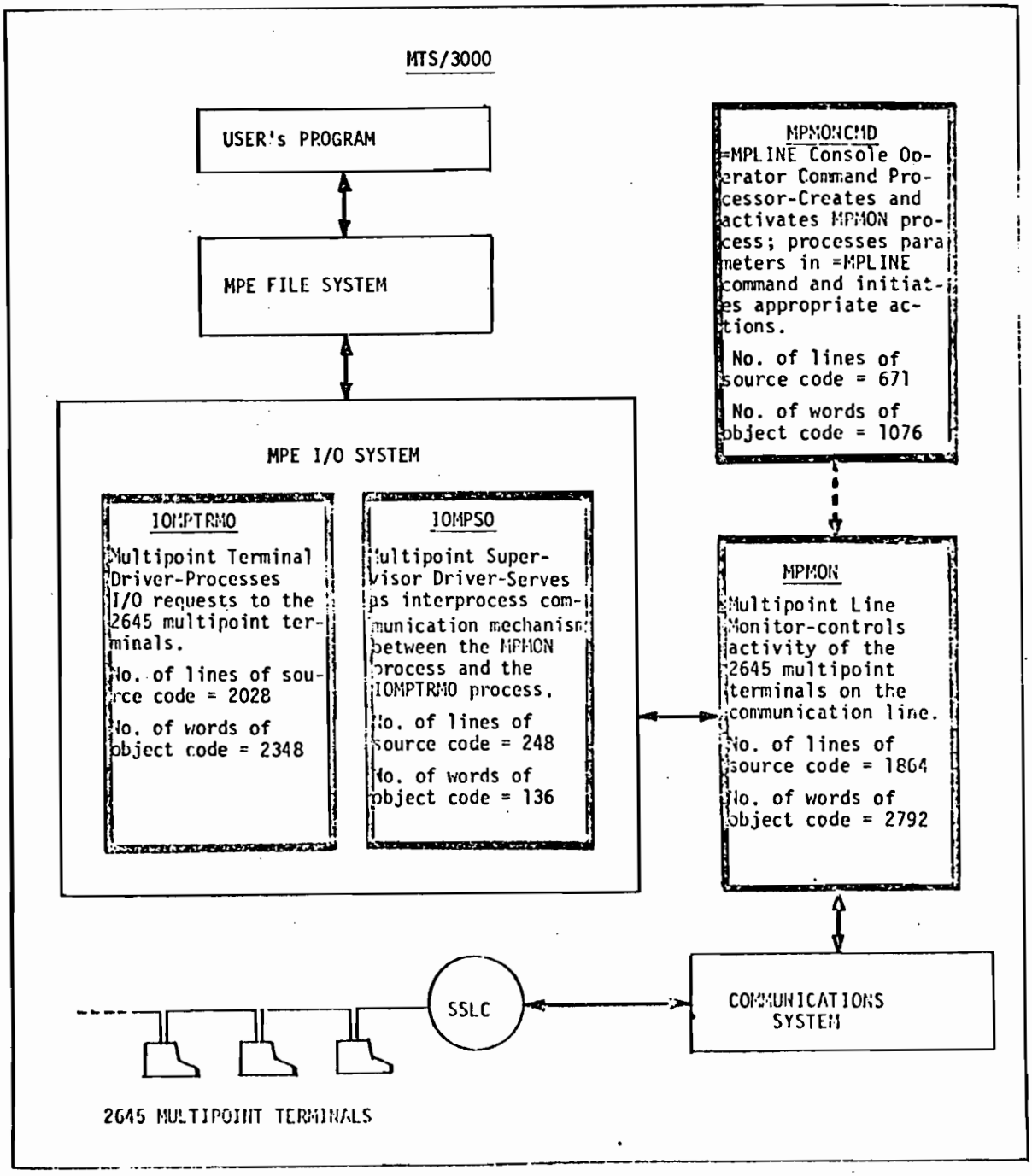
MPMON - Multipoint Line Monitor;

IOMPTRMO - Multipoint Terminal Driver;

IOMPSO - Multipoint Supervisor Driver;

MPMONCMD - =MPLINE Console Operator Command Processor.

The following block diagram schematically shows main characteristics of these modules and their relationship with other parts of the system.



I. INTRODUCTION

This document is the internal maintenance specifications (IMS) for the Multipoint Terminal Software (hereafter called MTS or MTS/3000). This document should be used with the MTS/3000 Reference Manual (32193-90002) and the source listings for the following MTS/3000 modules:

MPMON - Multipoint Line Monitor
(source name: S00S193A.HP32193.SUPPORT);

IOMPTRM0 - Multipoint Terminal Driver
(source name: S01S193A.HP32193.SUPPORT);

IOMPS0 - Multipoint Supervisor Driver
(source name: S02S193A.HP32193.SUPPORT);

MPMONCMD - =MPLINE Console Operator Command Processor
(source name: S03S193A.HP32193.SUPPORT).

The reader should also be familiar with the MPE File System, the MPE I/O System (in particular with SIO Device Monitor and SIO driver structure), and the Communications System (CS/3000).

MTS/3000 is installed by streaming the install file I00I193A.HP32193.SUPPORT in the product account.

The MTS/3000 Reference Manual (32193-90002) explains what MTS/3000 is, how to use terminals with MTS/3000, and how to configure MPE to include MTS/3000. The MTS/3000 Reference Manual also contains information about MTS/3000 hardware.

HP Computer Museum
www.hpmuseum.net

For research and education purposes only.

This section describes internal MTS/3000 functions, mutual relationship of MTS/3000 modules, and general flow of control and data.

Mutual relationship of MTS/3000 modules and their interaction with MPE I/O System and Communication System (CS) is schematically shown in Figures 1, 2, and 3. MTS/3000 modules are indicated by heavy lines. For clarity, the overall description of major MTS/3000 functions is divided in three parts:

1. Line Monitoring State;
2. Read Request Processing;
3. Write Request Processing.

The following discussion assumes MTS/3000 normal mode of operation; possible deviations will be discussed under functional description of individual modules. Also, unnecessary details are omitted. It is also assumed that Interrupt Acknowledge flag in an appropriate Device Information Table is set by the caller before the call to AWAKEIO is issued.

In the following discussion, reference numbers in paranthesis in the text, correspond to reference numbers in small circles in Figures 1, 2, and 3.

2.1 LINE MONITORING STATE

Line monitoring state (See Figure 1) is the state in which MTS/3000 is ready either to receive data from the terminals or to transmit data to the terminals.

After the console operator enters the console command =MPLINE ldn, OPEN (1), the =MPLINE Command Processor MPMONCMD (2) will create and activate the Multipoint Line Monitor process MPMON (3). MPMON constructs the SUPLIST array which contains necessary information about the terminals on the communication line. MPMON then sets up initial values of certain variables and opens the line through COPEN intrinsic call. The name of the SUPLIST array is passed to COPEN as one of the parameters. MPMON also opens a file on the Multipoint Supervisor associated with the line through the FOPEN intrinsic call and allocates the Multipoint Supervisor through FCONTROL, controlcode 37. MPMON then issues a read I/O request against the multipoint line through CREAD intrinsic call (12). The name of the line read buffer (21), in which the data from the terminals on the line will be received, is passed to CREAD as one of the parameters. MPMON also issues a dummy read I/O request against the Multipoint Supervisor (11) through FREAD intrinsic call (4). No real read buffer is associated with this request because no actual data transfer will take place between the Multipoint Supervisor and MPMON. The CREAD call against the line and the FREAD call against the Multipoint Supervisor is followed by a call to IOWAIT (13) and (13) and MPMON becomes inactive.

After CREAD (12) is called by MPMON (3), CS (14) calls the ATTACHIO procedure (15) for the line device controller (19). The ATTACHIO procedure is the entry point to the MPE I/O system (15) which constructs a read IOQ element (24) and links it to the device queue for the line device controller (19). If this is the first element in the queue, the AWAKEIO procedure (17) is called for the line device controller (19). AWAKEIO (17) will cause the CS multipoint driver (18) to be executed. This driver constructs and starts an appropriate SIO program for continuous polling of all the configured multipoint terminals (20) on the line.

Similarly, after the FREAD (4) is called by MPMON, the File System (5) calls the ATTACHIO procedure (6) for the Multipoint Supervisor(11) which results in constructing a read IOQ element (23) and linking it to the device queue for the Multipoint Supervisor(11). The AWAKEIO procedure (8) is then called for the Multipoint Supervisor (11) causing the I/O device monitor (SIODM) (9) and the Multipoint Supervisor logical driver (IOMPS0) (10) to be executed. This driver initiates a dummy read requests (29) against the Multipoint Supervisor (11), then it releases all the resources and enters the waiting state. MTS is now ready to process read and write I/O requests to the terminal.

2.2 READ REQUEST PROCESSING

An executing user process (30) (See Figure 2) generates a read request to the File System (31) through the call to FREAD intrinsic. The name of the user buffer (39) is passed as one of the parameters to FREAD. The File System (31) calls the ATTACHIO procedure (32) for the Multipoint terminal (20) which results in constructing a read IOQ element (34) and linking it to the device queue for the multipoint terminal (20). The AWAKEIO procedure (35) is then called for the multipoint terminal (20) causing the I/O device monitor (SIODM) (36) and the multipoint terminal logical driver (IOMPTRM0) (37) to be executed. This driver initiates a read request (38) against the multipoint terminal (20). Then a check is made by IOMPTRM0 (37) whether data has been received from the terminal (20), and is ready in the terminal read buffer (28). If not, IOMPTRM0 (37) releases all the resources and enters the waiting state.

If a terminal user (40) presses the ENTER key on the terminal keyboard, the terminal (20) responds to a poll with data and the CS multipoint driver (18) initiates data transmission (41) from the terminal (20) to the CS buffer (22) via the line device controller (19).

Communications System (14) then transfers data (42) from the CS buffer (22) to the line read buffer in the MPMON stack (21) and indicates CREAD completion (43) to MPMON (3). MPMON releases the line read buffer (21) by transferring data (44) from the line read buffer (21) to the terminal read buffer (28) and indicates in the terminal DIT (DITT) (27) that data has been received. MPMON then calls (45) the AWAKEIO procedure (35) for the multipoint terminal (20) causing the I/O device monitor (SIODM) (36) and the multipoint terminal logical driver (IOMPTRM0) (37) to be executed. IOMPTRM0 checks a flag in the terminal DIT (DITT) (27) whether data has been received from the terminal (20). Since the data has already been received and is prepared in the terminal read buffer (28), IOMPTRM0 transfers data (46) from the terminal read buffer (28) to the user buffer (39) and completes the read request.

In the meantime, after AWAKEIO (35) has been called, MPMON (3) reissues a read I/O request against the multipoint line through CREAD intrinsic call (12) independently of the MPE I/O system (33) action and is prepared to receive data from other terminals on the line. The call to CREAD (12) is followed by a call to IOWAIT (13), MPMON (3) becomes inactive, and MTS enters again the line monitoring state.

It should be noted that MPMON (3) performs certain editing operations on data received from the terminals. For example, CR and LF characters are normally filtered out of the input character string unless this operation is explicitly disabled

from the user's program. The type of editing depends on the request (e.g. ASCII read versus binary read). Such information is contained in P1 and P2 parameters in the IOQ element. Since it may happen that the terminal user (40) enters data before the user process (30) issues a read request to the File System (31) and therefore no IOQ element (34) currently exists, it is sometimes necessary to keep both edited and unedited data. If this occurs, unedited data is kept in the secondary read buffer (47). When the IOQ element (34) is finally constructed, IOMPTRM0 (37) decides which data will be transfered to the user buffer (39) according to the information in the IOQ element (34).

2.3 WRITE REQUEST PROCESSING

An executing user process (48) (See Figure 3) generates a write request to the File System (49) through the call to FWRITE intrinsic. The name of the user buffer (61) is passed as one of the parameters to FWRITE. The File System (49) calls the ATTACHIO procedure (50) for the multipoint terminal (20) which results in constructing a write IOQ element (52) and linking it to the device queue for the multipoint terminals (20). If this is the first element in the queue or the request specifies preemption, the AWAKEIO procedure (53) is called for the multipoint terminal (20) causing the I/O device monitor (SIODM) (54) and the multipoint terminal logical driver (IOMPTRM0) (55) to be executed. IOMPTRM0 initiates a write request (56) against the multipoint terminal (20), transfers data (57) from the user buffer (61) to the line write buffer (62) in the MPMON stack, sets a flag in the DIT for the Multipoint Supervisor (DITS) (26) indicating that MPMON (3) should be activated, calls (58) the AWAKEIO procedure for the Multipoint Supervisor (8), and logically completes the write request (56) against the multipoint terminal (20).

AWAKEIO (8) will cause the I/O device monitor (SIODM) (9) and the Multipoint Supervisor logical driver (IOMPS0) (10) to be executed. IOMPS0 (10) completes the dummy read request (59) against Multipoint Supervisor (11) which results in activating MPMON process (3) and completion of dummy FREAD (60) previously issued from MPMON. MPMON (3) then checks if there is any data in the line write buffer (62) prepared to be written to a multipoint terminal (20). If yes, MPMON (3) aborts currently pending CREAD (63) by issuing CCONTROL, controlcode 0, against the line. After CS (14) signals to MPMON (3) that CREAD has been successfully aborted (64), MPMON (3) issues a write I/O request against the line through CWRITE intrinsic call (65). The name of the line write buffer (62) is passed to CWRITE as one of the parameters.

After the CWRITE (65) is called by MPMON (3), CS (14) transfers data to be written (66) from the line write buffer (62) to the CS buffer (22). CS (14) then calls the ATTACHIO procedure (15) for the line device controller (19) which results in constructing a write IOQ element (24) and linking it to the device queue for the line device controller (19). The AWAKEIO procedure (17) is then called for the line device controller (19) causing the CS multipoint driver (18) to be executed. This driver constructs and starts an SIO program which transfers data (67) from the CS buffer (22) to the multipoint terminal (20) via the line device controller (19). After the data has been successfully written, CS (14) informs MPMON (3) about FWRITE completion.

MPMON (3) writes data to the terminal on the line until either the line write buffer (62) is empty or the amount of written data exceeds a certain limit. Note that whenever the dummy FREAD is completed (60), MPMON (3) immediately reissues a new dummy FREAD (4) against the Multipoint Supervisor (11).

If the line write buffer (62) is empty or the amount of written data exceeds a certain limit, MPMON (3) reissues a read I/O request against the line through CREAD intrinsic call (12). IOWAIT (13) is then called, MPMON (3) becomes inactive, and MTS enters the line monitoring state.

III. FUNCTIONAL DESCRIPTION OF MTS/3000 MODULES

MTS/3000 consists of the following modules:

MPMON - Multipoint Line Monitor

IOMPTRM0 - Multipoint Terminal Driver

IOMPS0 - Mutlipoint Supervisor Driver

MPMONCMD - =MPLINE Console Operator Command Processor

3.1 MPON - MULTIPOINT LINE MONITOR

MPMON is a system program designed to control activity of the 2645 multipoint terminals on the communication line. It regulates all transmissions on the line through calls to CS intrinsics. CS in turn calls an appropriate communication driver which accesses terminals on the line by means of polling and selection.

MPMON runs in its own process which is created and activated for each line through the operator console command =MPLINE (see below). MPMON interfaces to IOMPS0 through file system intrinsic calls and to IOMPTRM0 through calls to AWAKEIO.

When MPMON starts executing, it constructs a byte array SUPLIST which contains all the necessary information about the terminals on the line. The general format of SUPLIST is described in the CS/3000 ERS. During the construction of SUPLIST mutual relationship of station numbers (used by CS) and SYSDB relative DIT pointers (used by MPE I/O system) is remembered in SDITMAP (Station - DIT map).

MPMON then sets up initial values of some variables and opens the line through the call to COPEN. MPMON also opens and allocates the Multipoint Supervisor through calls to FOPEN and FCONTROL, controlcode 37, respectively.

If BUSYEHEAD is false, output buffer (OUTBUF) is empty, i.e., no data is prepared to be written to the terminals.

If SHORTREAD is true then only one pass through the poll list will be made, otherwise 100 passes will be made. 100 passes is specified instead of infinite number of passes in order to ensure proper servicing of a terminal group which might have been previously powered down.

SHORTREAD is set to true whenever a read or write request is completed. One pass through the poll list then ensures that remaining terminals on the line will also get attention if they have anything to send.

Two counters RSDOWNCNT and WSDOWNCNT are maintained in order to periodically bring all stations (terminals) up. This ensures proper servicing of a group which might have been temporarily down.

TRACEONREQ and TRACEOFFREQ are used to indicate whether the trace facility should be turned on or off, respectively.
Two important subroutines are SERVICEREAD and SERVICEWRITE. SERVICEREAD is a subroutine used to service a read request after the data from a terminal on the line has been received. This subroutine obtains SYSDB relative DIT pointer of the responding

terminal from station - DIT map (array SDITMAP) using the station number (STATIONI) passed to MPMON by CS on read completion.

If a single Cancel character (%30) is received from a terminal and the terminal is not in attention state (ATTENTERM is false), this situation is interpreted as a request to set attention state. If the terminal is already in attention state (ATTENTERM is true), and a single Cancel character is received, MPMON requests a system break. If the terminal is in attention state and other data than a single Cancel character is received, MPMON requests a subsystem break.

Depending on information contained in the read IOQ element, CR and LF may be filtered off the input character string. Subroutine FILTERCRLF is used for this purpose. If no IOQ element has yet been constructed by MPE I/O system for the terminal which sent data, unfiltered data is kept in an auxilary data segment (refer to (47) in Figure 2) and filtered data is kept in the regular terminal read buffer (refer to (28) in Figure 2).

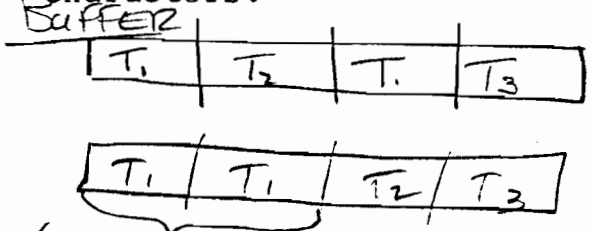
Data from the terminals is always received in the line input buffer (INBUF) and is then moved to appropriate data segment representing the read buffer for a particular terminal (refer to (28) in Figure 2).

MPMON also takes care of device recognition. If a terminal is not allocated, no IOQ element exists for this terminal, and some data has been received from this terminal (terminal user pressed the ENTER key to log on), then the device recognition procedure is started by setting SIODM state in the terminal DIT to 6 and calling AWAKEIO for this terminal.

SERVICEWRITE is a subroutine used to service write requests to the terminals on the line. SERVICEWRITE operates on data entries contained in the line output buffer (OUTBUF). Each data entry consists of a four-word header followed by data to be written to the terminal. The format of the header is shown in Figure 4. The header is constructed and moved with data to OUTBUF by IOMPTRM0 (see below). Index of the first data entry in OUTBUF is held in HEADI, index of the last data entry in OUTBUF is held in TAILI.

Figure 5 schematically shows OTBUF containing several data entries.

Before CWRITE intrinsic is called, a check is made whether subsequent data entries in OUTBUF belongs to the same terminal. If yes, data belonging to several subsequent data entries is concatenated into one data block which may contain up to 1200 characters.



MAX UTILIZATION OF A LINE ADJACENT BUFFERS (TWO WRITES TOGETHER) ARE DONE W ONE WRITE

Normally, a check is also made whether data to be written contains incomplete or incorrect escape sequences. If yes, MPMON does not append ESC_ sequence to the output character string.

Note that ENDI holds index of last available word in OUTBUF. Area beyond ENDI is overlayed by SDITMAP.

3.2 IOMPTRM0 - MULTIPOINT TERMINAL DRIVER

IOMPTRM0 is an SIO-like driver designed to process I/O requests to the 2645 mutlipoint terminals. It should be emphasized that only SIO driver structure is implemented but no SIO transfers are directly controlled by IOMPTRM0. The driver interfaces to the operating system via the SIO Device Monitor (SIODM). SIODM is core resident, contains a routine for processing preemptive requests, and also contains mechanism for device recognition.

IOMPTRM0 is a type 2 driver. It constitutes together with SIODM a type 2 I/O system monitor which is able to execute either in a user process or in the I/O process.

IOMPTRM0 supports most of the functions currently applicable for the Asynchronous Terminal Controller Driver (IOTERM0) through calls to ATTACHIO. The functions are summarized in Table I.

The driver returns information to the caller via the IOQ element. A status is placed into QSTAT word (word 10) of the IOQ element to inform the caller what happened to his request. Status returns are shown in Table II.

In order to maintain compatibility with earlier written programs, some requests not applicable for multipoint environment are returned as successfully completed although no action takes place (e.g., enable echo). However, requests for those operations which are not supported will be returned as invalid.

The amount of data transferred is returned in the QWBCT word (word 7) of the IOQ element. A byte transfer is expressed as a negative number. The return value has the same sign as the original value. If no data is transferred a zero count is returned. Parameters P1 and P2 (words 8 and 9 of the IOQ element) are used to convey request dependent information to the driver as shown in Table I.

There is one Device Information Table (DIT) for each multipoint terminal on the line. In order to distinguish the DIT for multipoint terminals from the DIT for the Multipoint Supervisor, the terminal DIT is denoted by DITT and the DIT for the Multipont Supervisor is denoted by DITS. DITT contains information about a particular terminal and I/O requests issued against this terminal. The format of DITT is shown in Figure 6. The explanation of individual fields in DITT follows:

DFLAG - Flags and SIODM state.

.ACTIVE - SIODM is currently active servicing this device.

.REQUEST - Service for this device was requested while SIODM was active.

.PREMPT - Preemptive request flag.

.IAK - Response has occurred (interrupt acknowledge flag).

.STATE - SIODM state.

DLINK - SYSDB relative pointer to the DIT for the next device requesting service or this resource.

DIOQP - SYSDB relative pointer to the first IOQ element in the request list for this device.

DLDEVT - Logical device number and unit number.

.LDEVNT - Logical device number of the multipoint terminal.

.UNIT - Unit number representing terminal address (group and device ID).

DDLTP - SYSDB relative pointer to Driver Linkage Table (DLT).

DILTP - SYSDB relative pointer to dummy Interrupt Linkage Table (ILT) to satisfy SIODM requirements (no real ILT is associated with multipoint terminals).

DTIME - Timer flags.

.READTOF - Read timeout has occurred.

.LOGONTOF - Log on timeout has occurred.

DMISCT - Miscellaneous flags.

.GSIN - Last character received from the terminal was the GS character.

.READERROR - Read error has occurred.

.FILTERCRLFOK - Proper editing of input data with respect to CR and LF characters has already been made.

.MARKED - This DITT has already been processed during construction of SUPLIST.

.WPOSTP - Current write request has been postponed.

.READPEND - Read request is pending against this terminal.

.DATAREADY - Input data has been received and is ready in the terminal read buffer.

.UP - Device has been initialized through the log on procedure or has been allocated.

.PRESPACEF - Last write operation was with a prespace request. If the next write operation is with a post space request, output CR and LF before data.

.READTIMERF - Read timing requested and not yet in progress.

.TIMING - Current read request is being timed.

.BRKOK - System break is enabled.

.SSBRKOK - Subsystem break is enabled.

.FLUSH - This flag is set whenever a break has been detected and accepted. While it is set, writes are returned completed without any I/O being done. Reads are returned with an unusual condition status %173. It also holds off any further break service requests. It is reset with a function code 25 operation.

.LASTPREMPT - Last request was a preemptive request.

DLDEVL

.LDEVNL - Logical device number of the controller servicing the multipoint line.

DDSBUF - Data segment number of the terminal read buffer.

DWLIM - Write limit counter.

DFORMAT

.FORMATF - This field holds information about vertical format specification for writes obtained from P1 parameter of the IOQ element or from the first data byte.

DNEXT - SYSDB relative pointer to the DITT for the next terminal on the same line.

DNWRITE - SYSDB relative pointer to the DITT for the next terminal with postponed write.

DSTATION - Flags and station number.

.LFLUSH - This flag is set to indicate that data for this terminal already scheduled to be written from the output buffer should not be physically sent to the terminal (break or subsystem break environment).

.DISCONREQ - Request to disconnect the terminal.

.BREAKMODE - Terminal is in break mode.

.ATTENTERM - Terminal is in attention mode.

.SSBMODE - Terminal is in subsystem break mode.

.WLQUEUE - A write request was forced to be queued by MPE I/O system.

.DJSTATE - State of terminal straps D and J.
0 - Initial state.
1 - Straps D and J are open or will be open before the next write.
2 - Undefined D and J setting.

.STATIONINDIT - Station number assigned to this terminal by CS.

DFIRST - Storage for first word for ASCII writes if veritcal format is specified by first data byte.

DBCNT - Actual byte count for reads.

DTIND - Timer indexes.

.READTINDEXF - Read timer index.

.LOGONTINDEXF - Log on timer index.

DRTIME(DRTIMED) - During a timed read, this is the reading of the timer at the initiation of the read. After a timed read is completed, the time in 1/100 of a second is saved in DRTIME as a single word. If it is -1 then the time was greater than 32K.

DRTMAX - When a read operation timeout is requested, this quantity represents the maximum time in seconds allowed for the read to be completed.

DTYPE - Terminal type and speed.

.TERMINALTYPE - Configured terminal type. Multipoint terminal is type 14.

.SPEED - Reserved field for configured terminal speed (not used for multipoint terminals).

DDSHEL - DST number of data segment holding "HELLO" message (or backspaced data).

DHBCNT - Byte count for "HELLO" message (or backspaceed data).

DDSBUF2 - Data segment number of secondary read buffer.

DBCNT2 - Byte count for read if secondary read buffer is used.

DMOD1VER - Current version number of the multipoint terminal driver (IOMPTRM0).

DUNMODE - Unedited mode characters.

.ATTENCHAR - Attention character.

.ENDCHAR - End-of-record character. (Effective as a control character if set to %137, otherwise not used).

3.3 IOMPS0 - MULTIPOINT SUPERVISOR DRIVER

IOMPS0 is also an SIO-like driver designed to serve as an inter-process communication mechanism between the MPMON process and the IOMPTRM0 process. IOMPS0 interfaces to the operating system via SIODM in a similar way as IOMPTRM0.

IOMPS0 is a type 1 driver. It constitutes together with SIODM a type 1 I/O system monitor which is able to execute on any stack.

IOMPS0 is configured in the system as a driver for the Multipoint Supervisor (a virtual device associated with each line).

The main task of IOMPS0 is to initiate and complete dummy read requests which serve as a mechanism for activating MPMON process (exit from IOWAIT).

Since the Multipoint Supervisor is configured in the system as a regular I/O device with a unique logical device number, there exists one DIT for each Multipoint Supervisor (denoted by DITS). DITS contains information relevant to all terminals on the line and also information pertinent to I/O requests issued against the Multipoint Supervisor. The format of DITS is shown in Figure 7. The explanation of individual fields in DITS follows:

DFLAG-
DLINK-
DIOQP- } Same as for DITT

DLDEVS - Logical device number and unit number.

.LDEVNS - Logical device number of the Multipoint Supervisor.

.UNIT - Unit number (always 0).

DDLTP -
DILTP - } Same as for DITT

DTIME - Reserved for timer flags.

DMISCS - Miscellaneous flags.

.MPOK - If set, then IOMPS0 is allowed to process I/O requests against the Multipoint Supervisor.

.DEBUGON - If set, then DEBUG will be called from MPMON. This flag is set through the =MPLINE command.

.TRACEON - Trace facility is enabled.

.TRACEOFFREQ - Trace facility is to be disabled.

.TRACEONREQ - Trace facility is to be enabled.

.SHUTNOW - Request to shut the line immediately.

SHUTREQ - Request to shut line after all terminals are released. New sessions are not allowed to be initiated.

BUSYHEAD - The line write buffer contains data to be written to a terminal on the line.

.MPMONACT - MPMON process is active.

.MPMONUP - MPMON process has been created and activated.

.GENWPOSTP - A write request for one or more terminals on the line has been postponed.

.GENDISCON - Request to disconnect the line.

.COMPLREQ - Request to complete dummy read pending against the Multipoint Supervisor.

DLDEVL

.LDEVNL - Logical device number of the controller servicing the multipoint line.

DDITSP - SYSDB relative pointer to the DIT for the Multipoint Supervisor (DITS).

DTBUFOFFS - Offset to the trace buffer in MPMON stack.

DWLCON - Write limit constant.

DNEXT - SYSDB relative pointer to the DITT for the first terminal on the line (the terminal with the lowest logical device number).

DNWRITE - SYSDB relative pointer to the DITT for the first terminal with postponed write.

DMOD2VER - Current version number of the Multipoint Supervisor driver (IOMPS0).

DINBUFA - Address of the line read buffer in MPMON stack.

DOUTBUFA - Address of the line write buffer in MPMON stack.

DOSPEED - Output speed.

DHEADI - Index of head entry in the line write buffer.

DTAILI - Index of tail entry in the line write buffer.

DENDI - Index of last available word in the line write buffer.

DTYPE

.TERMINALTYPE - Configured terminal type. Multipoint Supervisor is type 14 (same type as multipoint terminals).

.SUPER - This device is a Multipoint Supervisor.

.DITSOK - DIT's for the multipont terminals and the Multipoint Supervisor on this line have been rearranged and their format corresponds to standard DIT format for SIO devices.

.DSPEED - Reserved field for configured terminal speed (not used for Multipoint Supervisor.).

DMOD3VER - Current version number of the =MPLINE command processor (MPMONCMD).

DMONDSTN - Data segment number of MPMON stack.

DLSPEED(DLSPEEDD) - If not equal to 0, then the line is opened with speed specified in this double word.

3.4 MPMONCMD - =MPLINE CONSOLE OPERATOR COMMAND PROCESSOR

The format of the =MPLINE command is described in the MTS/3000 Reference Manual. In addition, the operator can also use the DEBUG parameter as follows:

$$\text{=MPLINE ldn, DEBUG } \left[\text{, } \left\{ \begin{matrix} \text{on} \\ \text{off} \end{matrix} \right\} \right]$$

Entering =MPLINE ldn, DEBUG results in calling DEBUG from MPMONCMD. If =MPLINE ldn, DEBUG, $\left\{ \begin{matrix} \text{on} \\ \text{off} \end{matrix} \right\}$ is entered, the

DMISCS.DEBUGON bit is set on or off, respectively, in the DITS. This bit is checked by MPMON (if MPMON is active) and if set, DEBUG is called from MPMON. If MPMON is in line monitoring state (i.e. waiting to receive data from a terminal and therefore inactive) it may be necessary to press the ENTER key on some terminal in order to activate MPMON.

MPMON checks validity of parameters specified in =MPLINE commands and initiates appropriate action.

When OPEN parameter is specified and the line is being opened for the first time, MPMONCMD collects all the terminals belonging to the line and constructs a linked list of SYSDB relative DIT pointers (DITS(DNEXT) contains SYSDB relative pointer to the first DITT, DITT(DNEXT) for the first terminal contains SYSDB relative pointer to the second DITT, etc.). MPMONCMD also rearranges DITS and all the DITT's in such a way that resulting DITS and DITT format corresponds to standard DIT format for SIO devices. Originally, word 5 of DITS and all DITT's contains logical device number of the line (instead of SYSDB relative pointer to ILT).

MPMONCMD then creates and activates MPMON process.

SHUT parameter results in terminating the MPMON process.

TRACE parameter (followed by ON or OFF and some other optional parameters) results in turning the trace facility on or off.

TABLE I. - MULTIPOINT TERMINAL I/O REQUESTS

Operation	Func Code	Parameters	Comments
Read	0	P1.(13:3) - EOF specificaiton	
		P1.(0:1) - No CR LF to be emitted at the end of a line	
		P2.(0:8) - If<> 0 then use as special read stop character	Read stop character is transmitted to the user buffer and is counted.
		P2.(11:2) - If 0 then ASCII else binary (Default - 0)	To read binary data the user must also enable the terminal by an appropriate escape sequence. No carriage control is applied after binary reads.

TABLE I. - MULTIPOINT TERMINAL I/O REQUESTS

Operation	Func Code	Parameters	Comments
Write	1	P1 - Vertical format specification	1 - Use first data character as format specification %53("+") - CR (no LF) %60("0") - CR LF LF %61("-") - FF (form feed) %200...%277 - CR and(n-%200) LF's %320 - No CR or LF All others result in CR LF
		P2.(15:1) - Prespace flag	If add, then do space operation before output of data. If even, then do space operation after output of data. If the request preceeding a post space request was a prespace request, CR LF is output before the post space request.
		P2.(11:2) - If 0 then ASCII else binary	No carriage control control is appended for binary writes.
File Open	2		No operation
File Close	3		Disables read timer and time-outs, clears all read flags, indicates indefined settings of D and J straps.
Device Close	4		Clears all mode and capability flags and sets terminal back to the initial state.
Set Read Timeout	5	P1 - If 0 then disable timeouts else timeout interval in seconds	

TABLE I. - MULTIPOINT TERMINAL I/O REQUESTS

Operation	Func Code	Parameters	Comments
Set Input Speed and Return Previous Speed	6	Current speed returned in count word.	Speed is not changed.
Set Output Speed and Return Previous Speed	7	Current speed returned in count word.	Speed is not changed.
Not Used-Enable Echo	8		No operation.
Not Used-Disable Echo	9		No operation.
Disable Break	10		
Enable Break	11		
Disable Subsystem Break	12		
Enable Subsystem Break	13		
Not Used-Disable P-tape Mode	14		Invalid request.
Not Used-Enable P-tape Mode	15		Invalid request.
Disable Read Timer	16		

TABLE I. - MULTIPOINT TERMINAL I/O REQUESTS

Operation	Func Code	Parameters	Comments
Enable Read Timer	17		
Return Read Timer	18	Count word contains read time in 1/100 seconds (a logical quantity)	
Not Used-Disable Parity Checking	19		No operation.
Not Used-Enable Parity Checking	20		No operation.
Logged-on	21	P1 - 0=JOB, 1=SESSION 2=DATA	Indicates that the user has successfully logged on and that the log on timeout is to be aborted. System break is also enabled.
Not Used	22		Invalid request.
Set Terminal Type	23	P1 - Terminal type. Requests for terminal types allowed: 14 and 0, otherwise invalid request.	Terminal type will not be changed from 14 to 0.
Allocate Terminal	24	P1 - Terminal Type. Requests for terminal types allowed; 14 and 0, otherwise invalid request.	Terminal type will not be changed from 14 to 0. This function allows the terminal to be set up without going through the log on procedure.
Clear Flush Flag and Write	25	Parameters same as for Write (function code 1)	System Break Flushing Flag is cleared before the write request is initiated.
Not Used-Disable CNTRL X Echo	26		No operation.
No Used-Enable CNTRL X Echo	27		No operation.

TABLE I. - MULTIPOINT TERMINAL I/O REQUESTS

Operation	Func Code	Parameters	Comments
Not Used	28		No operation.
Not Used-P-tape Read	29		Invalid request.
Set Break Mode	30	P1 - Odd - Set break mode Even - Clear break mode	
Not Used-Set Console Mode	31		Invalid request.
Not Used-Set Parity	32		No operation.
Allocate Terminal	33	Same as for function code 24	Same as for function code 24.
Set Terminal Type	34	Same as for function code 23	Same as for function code 23
Return Terminal Type	35	Terminal type returned in count word.	Terminal type returned will always be 14.
Return Output	36	Speed in characters per second is returned in cound word.	
Set Unedited Mode	37	P1.(0:8) - Attention character P1.(8:8) - End-of-record character	If P1.(8:8)=%137 (Underline), certain MTS/3000 functions are disabled as described in MTS/3000 Reference Manual.
Not Used-Set Console Interrupt	38		Invalid request.

TABLE II. - MULTIPOINT TERMINAL STATUS RETURNS

General (13:3)	Qualifying (8:5)	Overall (8:8)
0 - Pending	0 - Not started	0
	1 - Waiting for completion	%10
1 - Successful	0 - Normal completion	1
	1 - Completed on special read stop character	%11
2 - End of file	X - See description of MPE-EOF handling	%X2
3 - Unusual condition	2 - Read timeout	%23
	3 - I/O aborted externally	%33
	4 - Read Data Lost	%43
	5 - Terminal not on line or not ready	%53
	6 - Power fail abort	%63
	%15 - Enable subsystem break requested and no CY pin	%153
	%16 - Read time overflow	%163
	%17 - Read stopped when a break detected	%173
4 - Irrecoverable error	0 - Invalid request, function or parameter	4
	%12 - System error	%124

LINE MONITORING STATE

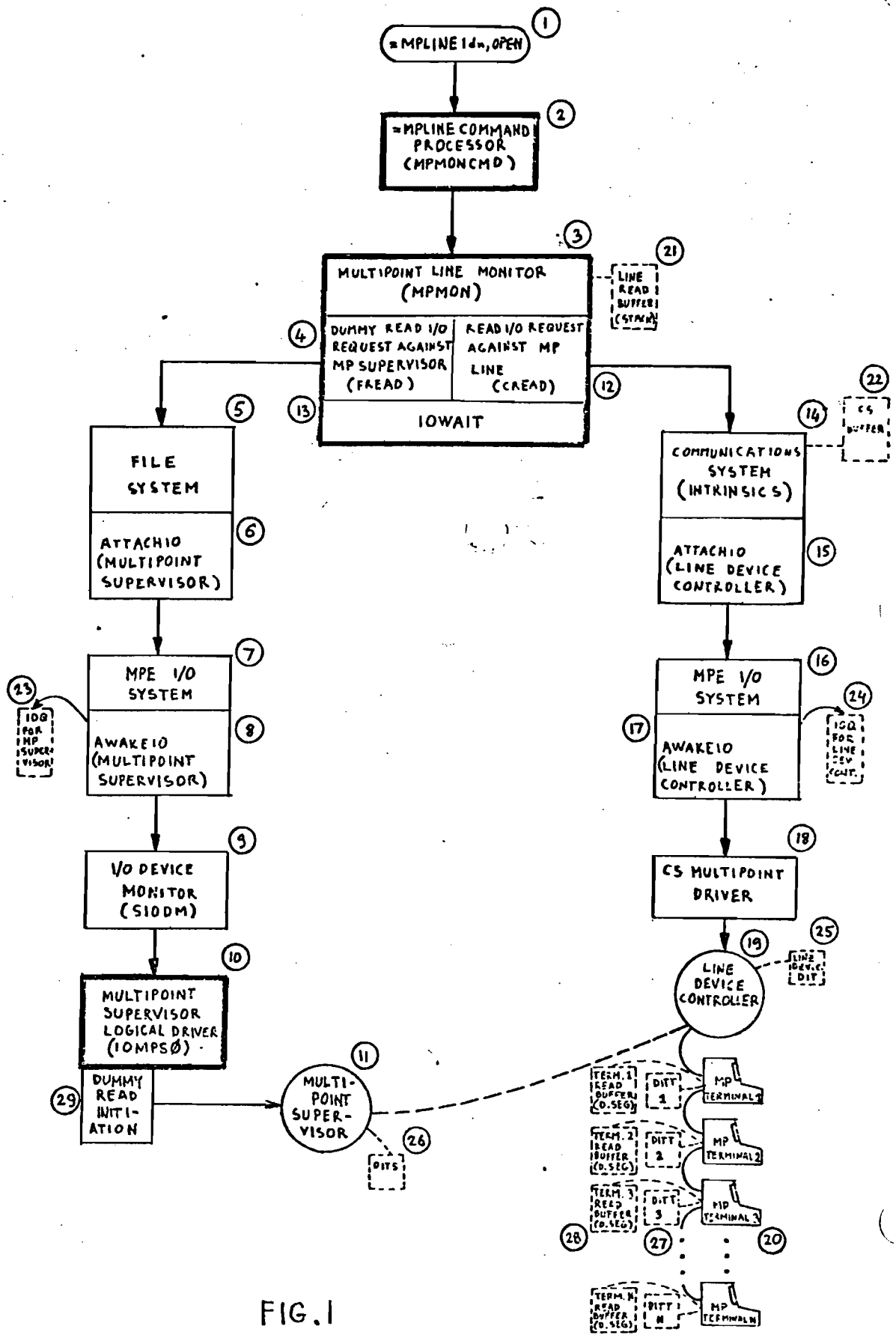


FIG. 1

READ REQUEST PROCESSING

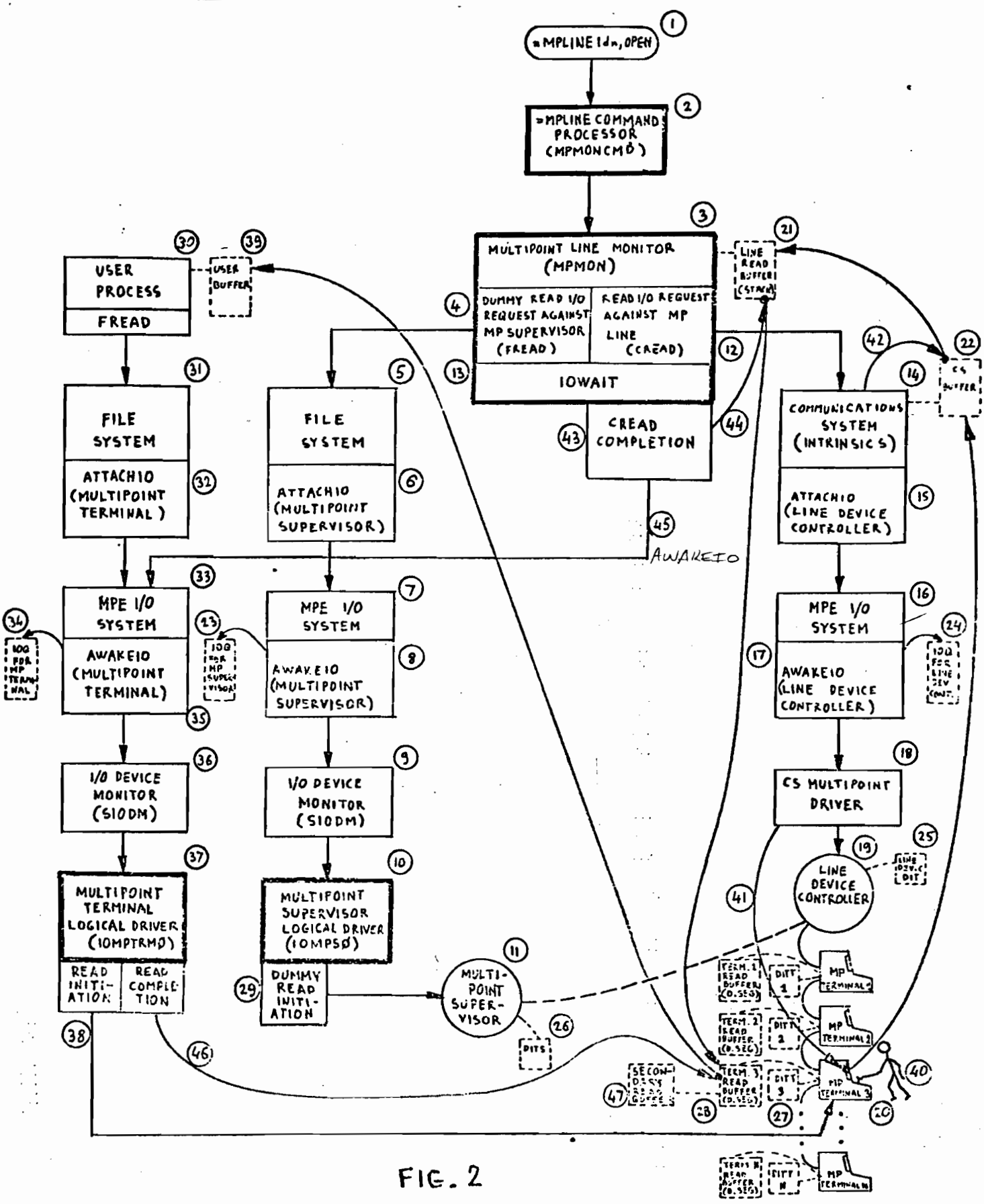


FIG. 2

WRITE REQUEST PROCESSING

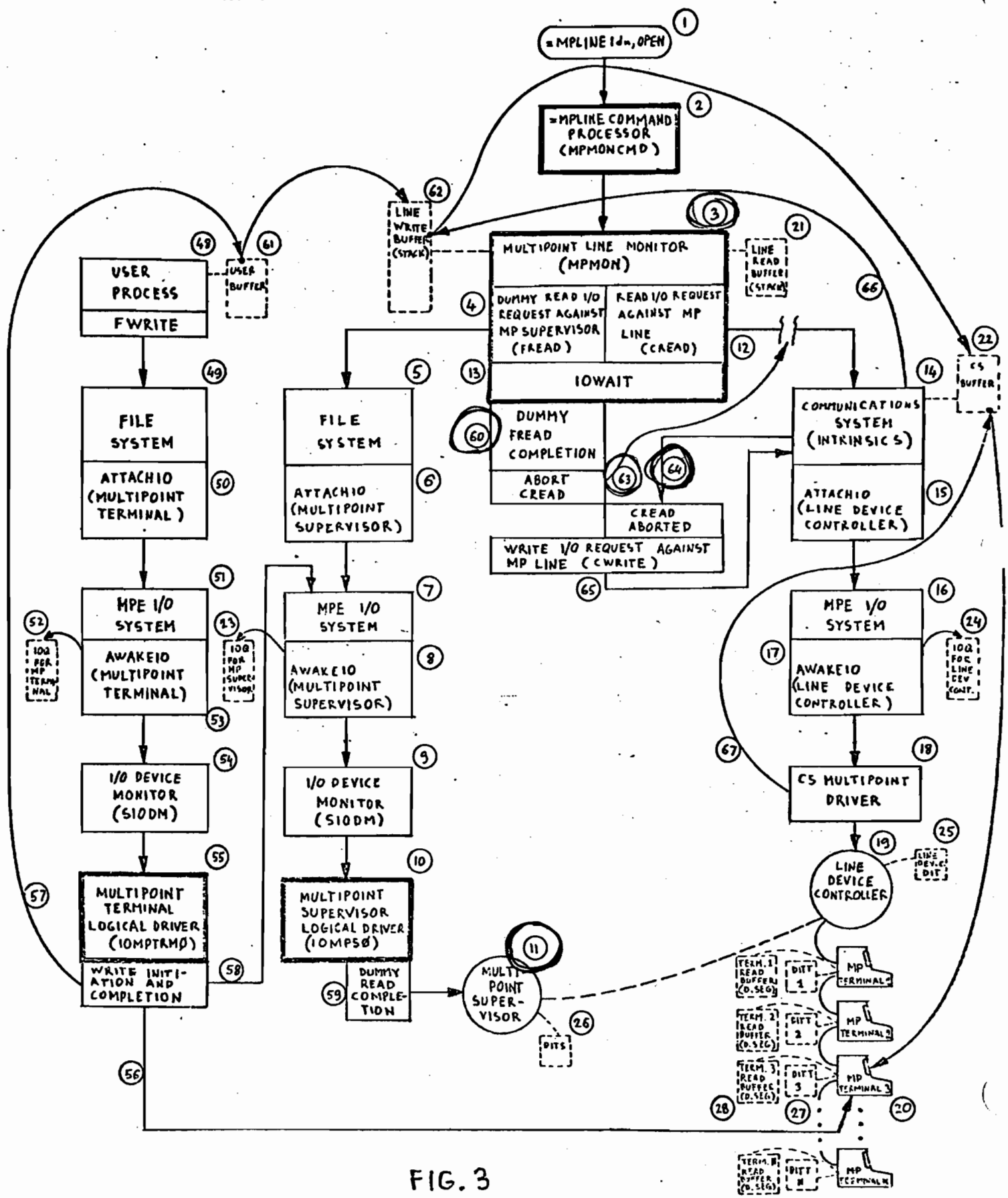


FIG. 3

HEADER FORMAT FOR DATA ENTRY IN LINE WRITE BUFFER

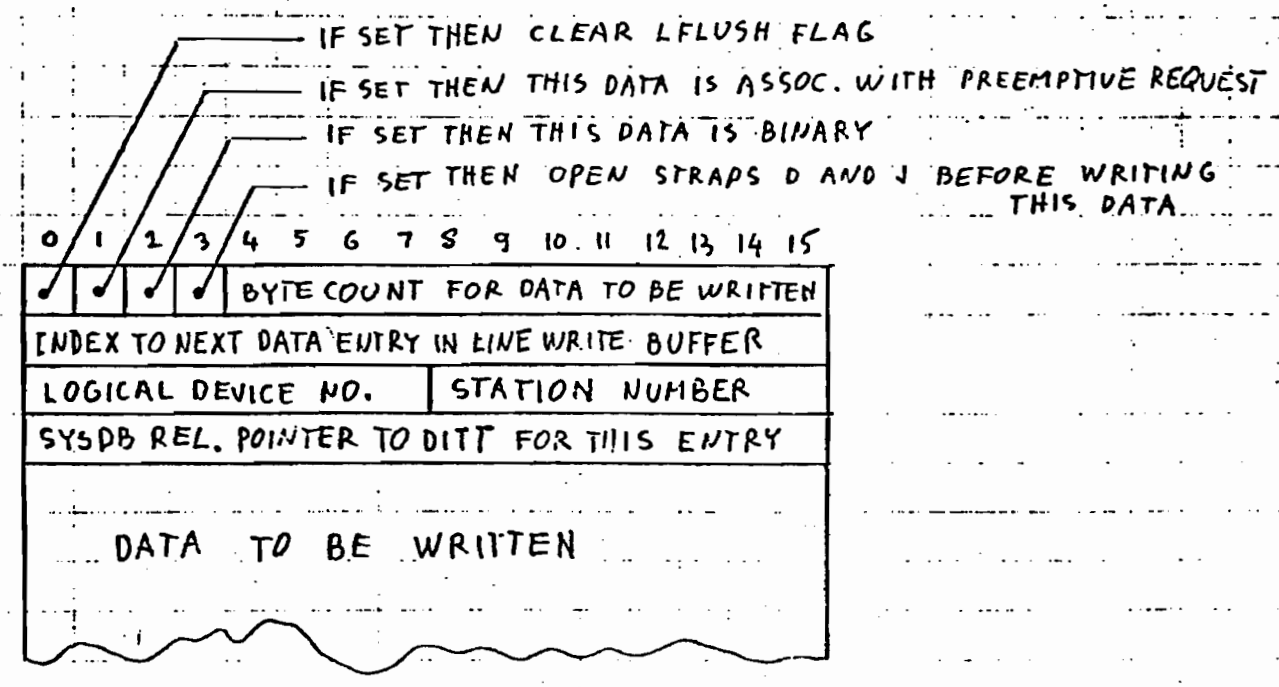


FIG. 4

OUTBUF CONTAINING SEVERAL DATA ENTRIES

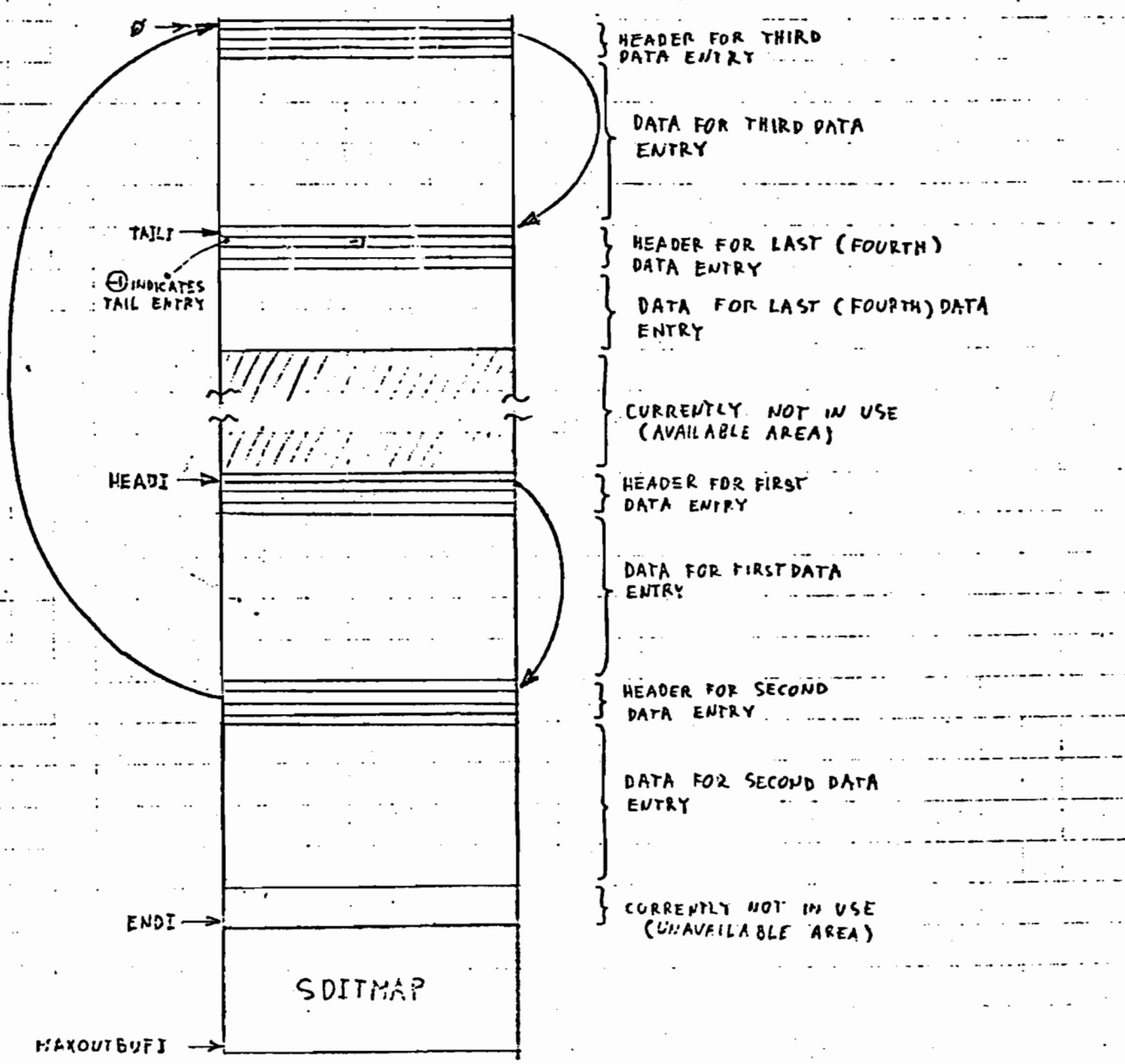


FIG. 5

MULTIPOINT TERMINAL DEVICE INFORMATION TABLE (DITT)

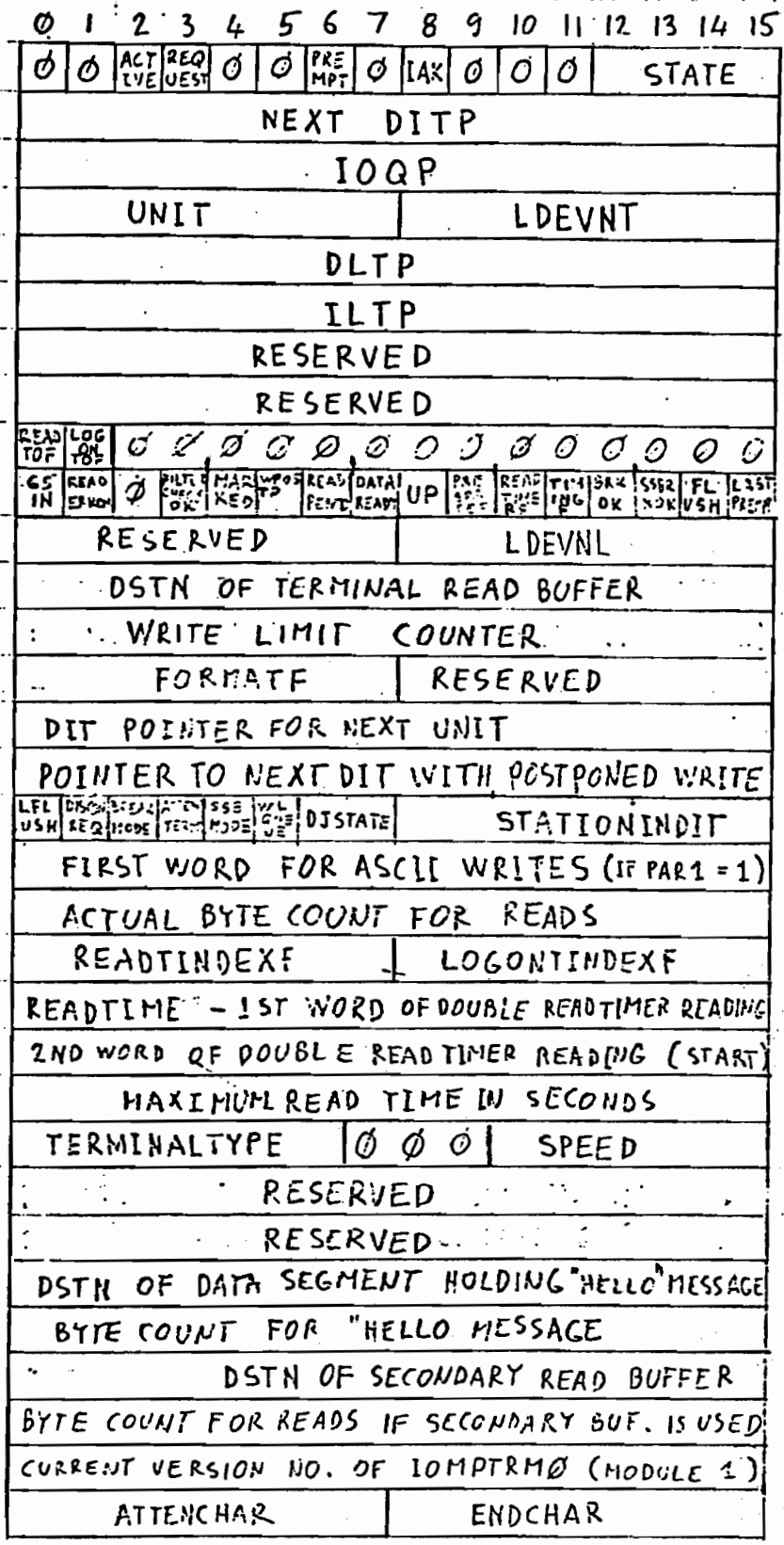


FIG. 6

MULTIPOINT SUPERVISOR DEVICE INFORMATION TABLE (DITS)

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15		
0	0	ACTIVE	REQUEST	0	0	PREMPT	0	LAK	0	0	0	STATE				0	DFLAG
NEXT DLTP																1	DLINK
IOQP																2	DIOQP
UNIT								LDEVNS								3	DLDEVS
DLTP																4	DDLTP
ILTP																5	DILTP
RESERVED																6	
RESERVED																7	
RESERVED FOR TIMER FLAGS																8	DTIME
MP OK	0	DE BUG ON	TRACE ON	TRACE OFF REQ	TRACE ON REQ	SHUT NOW	SHUT REQ	BUSY HEAD	MP MON ALT	MP MON UP	GEN K POST	GEN DISCN	0	0	CONTL REQ	9	DMISCS
RESERVED								LDEVNL.								10	DLDEVL
DIT POINTER FOR MP SUPERVISOR																11	DDITSP
OFFSET TO TRACE BUFFER IN MPMON STACK																12	DTBUFOFFS
WRITE LIMIT CONSTANT																13	DWLCON
DIT POINTER FOR FIRST UNIT																14	DNEXT
POINTER TO FIRST DITT WITH POSTPONED WRITE																15	DNWRITE
CURRENT VERSION NO. OF IOMPS0 (MODULE 2)																16	DMOD2VER
ADDRESS OF LINE READ BUFFER IN MPMON STACK																17	DINBUFA
ADDRESS OF LINE WRITE BUFFER IN MPMON STACK																18	DOUTBUFA
OUTPUT SPEED																19	DOSPEED
INDEX OF HEAD ENTRY IN LINE WRITE BUFFER																20	DHEADI
INDEX OF TAIL ENTRY IN LINE WRITE BUFFER																21	DTAILI
INDEX OF LAST AVAILABLE WORD IN L.WR.BUF.																22	DENDI
TERMINALTYPE							SUPER	DITS OK	0	SPEED						23	DTYPE
CURRENT VERSION NO. OF MPMONCMD (MODULE 3)																24	DMOD3VER
DSTN OF MPMON STACK																25	DMONDSTN
LINE SPEED - 1ST WORD																26	DLSPEED (DLSPEEDP)
LINE SPEED - 2ND WORD																27	

FIG. 7