MEMORANDUM
RM-5216-PR
JANUARY 1967

JOSS: USER SCHEDULING AND RESOURCE ALLOCATION

G. E. Bryan

PREPARED FOR:
UNITED STATES AIR FORCE PROJECT RAND

The RAND Corporation
SANTA MONICA • CALIFORNIA

MEMORANDUM
RM-5216-PR
JANUARY 1967

JOSS: USER SCHEDULING AND RESOURCE ALLOCATION

G. E. Bryan

This research is supported by the United States Air Force under Project RAND—Contract No. F44620-67-C-0045—monitored by the Directorate of Operational Requirements and Development Plans, Deputy Chief of Staff, Research and Development, Hq USAF. Views or conclusions contained in this Memorandum should not be interpreted as representing the official opinion or policy of the United States Air Force.

The RAND Corporation
1700 MAIN ST. • SANTA MONICA • CALIFORNIA • 90406

Published by The RAND Corporation

PREFACE

JOSS† is a multi-user, single-server computing system that provides for the solution of numerical problems. The system consists of a central computer containing the JOSS program and a number of typewriter consoles connected to the computer via telephone lines. The central computer turns its attention rapidly from console to console, in such a way that individual users would appear to have exclusive use of the system. The supervisory unit that exercises overall control of the system's operation is the monitor, which acts as a scheduling, resource-allocating, and synchronizing device, deciding when, and ensuring that, all data and hardware necessary for a particular action are simultaneously available.

This memorandum and its companion piece, JOSS: Accounting and Performance Measurement, present a detailed description of the monitor. The present study covers the priority queue structure, the hardware components comprising the JOSS computer system and their function in the JOSS environment, and instructions for operation of the JOSS system. The study should be of interest to those engaged in the design and implementation of time-shared or real-time computing systems.

This work is a part of The RAND Corporation's continuing program of research in computer sciences under U.S. Air Force Project RAND.

---

†JOSS is the trademark and service mark of The RAND Corporation for its computer program and services using that program.

SUMMARY

The monitor is that part of the JOSS system program that acts as the supervisory unit of the JOSS machine. Through this unit overall control of the system's operation is exercised through a detailed handling of input and output requirements. These are translated into a series of priority queues that order the "scheduling" of the system on a millisecond-by-millisecond basis and thus achieve effective concurrent user activity.

This report begins with a description of the PDP-6 hardware on which JOSS is implemented and the broad organization of the software that transforms the computer into a JOSS machine. This transformation is an important one and quite complete: It is virtually impossible for the user at a JOSS console to determine the characteristics of the underlying machine.

Changes in the priority queue structure in response to input, output, and computation signals are described in detail, as well as the way these changes are used to schedule the operation of the system. The monitor assigns each user in the system to a unique state. This state changes from time to time depending on the user's action at the console and the activity specified by his JOSS program. The monitor maintains a queue associated with each state that orders the users in each state according to time of arrival. Thus, the monitor can provide first-come, first-served processing for each state. In addition, the state queues are grouped and ordered to establish priorities for selection of users for JOSS compute time, transfers to and from the drum, disc transfers, and other system scheduled functions.

The two processors which represent that part of the monitor controlling user log-on and log-off are next detailed by sequence. Also presented is a description of those portions of the monitor that control system initialization, graceful shutdown, and several procedures that are convenient for operational and debugging purposes. Finally, the report concludes with a review of the techniques, limitations, and priority disciplines that enable JOSS to achieve its high response rate.

ACKNOWLEDGMENTS

I am indebted to my co-workers C. L. Baker, I. D. Greenwald, and J. W. Smith for innumerable helpful suggestions and constructive criticisms during the preparation of the code. The project would have surely been at a great loss without the tireless help of Janet Dorrough, who patiently assisted in the execution of hundreds of details. Finally, thanks are due Eleanor Harris for her work with the manuscript.

CONTENTS

FIGURES

TABLES

I. INTRODUCTION

JOSS[†] is a multi-user, single-server computing system consisting of a central computer containing the JOSS program and a number of typewriter consoles connected to the computer via telephone lines. The central computer turns its attention rapidly from console to console, in such a way that individual users appear to have exclusive use of the system. The system is purposely designed to be useful primarily to scientists and engineers with complex numerical problems.

Throughout this report, the term user is employed in several different ways. The real user is that person at the console engaged in the solution of a problem, but we will speak of manipulations of the user when what is really meant is the manipulation of machine records about the user. Thus, when we speak of transferring a user to the drum, we mean that the user's data are being transferred to the drum and not that some bodily harm is being done to his person.

[†]JOSS is the trademark and service mark of The RAND Corporation for its computer program and services using that program.

II. THE JOSS SYSTEM

This section describes the PDP-6 hardware on which JOSS is implemented and the broad organization of the software that transforms the computer into a JOSS machine. This transformation is an important one and quite complete: It is virtually impossible for a user at a JOSS console to determine the characteristics of the underlying machine.

JOSS HARDWARE

The essential hardware components of the PDP-6 computer system used for JOSS are outlined in Fig. 1. A more detailed description of the hardware is included in App. B.

The arithmetic processor, a word-organized multi-accumulator-index register machine, is provided with 32,000 words of high-speed core memory in two independently accessible 16,000-word boxes. JOSS uses one of the boxes, the low-addressed one, for the JOSS software, and the other to hold the user programs and data.

The processor contains a relocation register whose contents are added to memory references if certain conditions are met. The RAND PDP-6 has been modified so that the appearance of bit 20 in the address satisfies the requirement. The contents of the relocation register are set to the base address of the locations in memory that contain the user's program. All user references, as signaled by bit 20, are modified by the hardware to refer to the correct current user location. The contents of the relocation register represent, therefore, the context that determines the user of the moment. Since bit 20 corresponds to a real address of 32,768, both the size of the JOSS

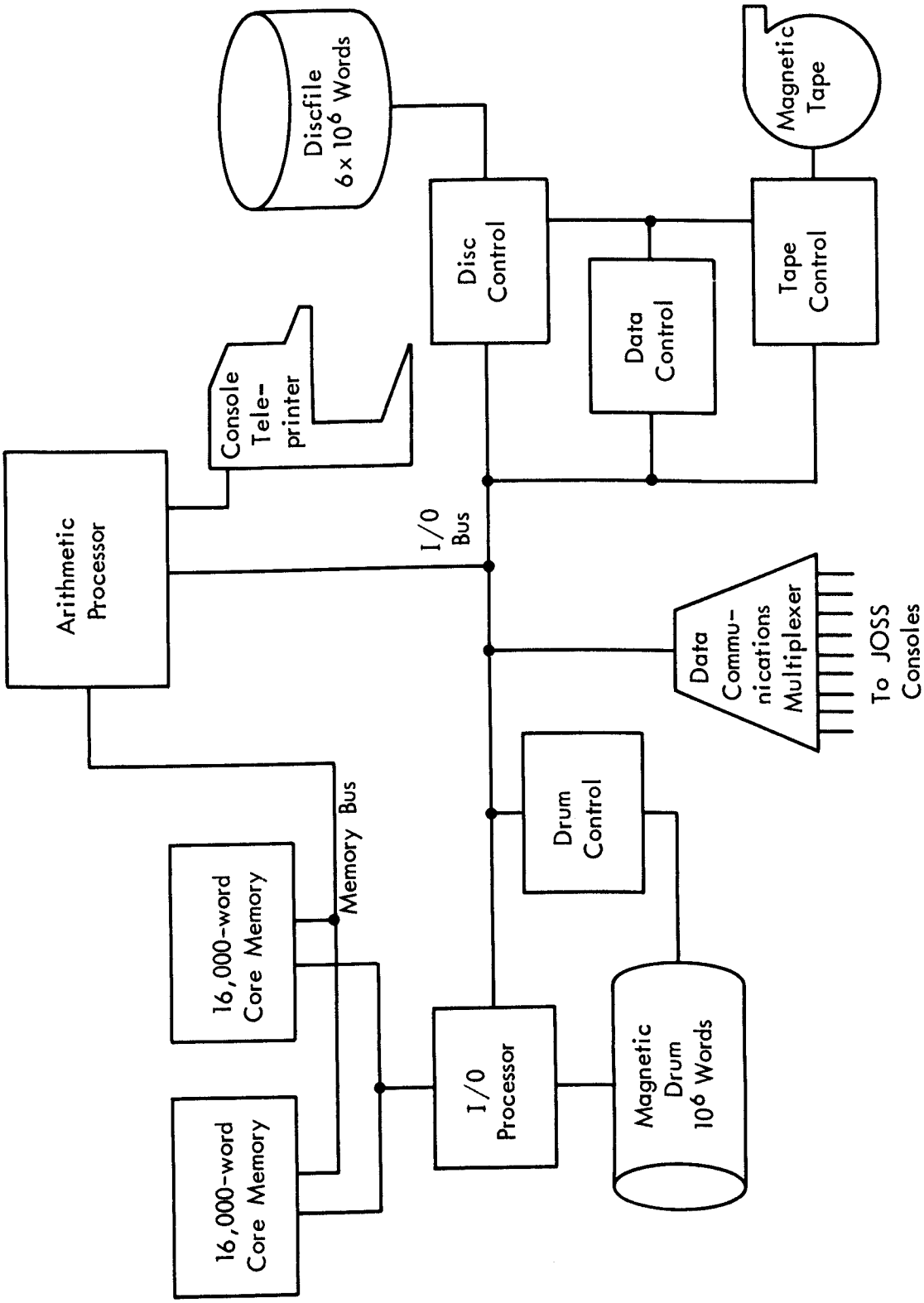

Fig. 1—PDP - 6 computer system hardware

system code and the size of individual users are limited to a maximum of 32,767 locations.

Because the capacity of core memory is not sufficient to hold data for all possible users simultaneously, the magnetic drum is used to store data for some of the users during those times when interpretation of their data is not required. Drum transfers are controlled by the input-output processor through independent ports to the memories. Thus, transfers of user data between the drum and core are accomplished independently of the arithmetic processor. The attention of the arithmetic processor is needed only to initialize the I/O processor for the transfer and to take action after its completion. Memory cycles are taken by the I/O processor as needed to service the drum. These cycles interleave with those taken by the arithmetic processor in the interpretation of JOSS users' programs.

Logging of information descriptive of the gross system operation is done on the console teleprinter. The information is printed each minute and includes the number of present users, the number of users computing, the amount of computing accomplished, the total lines transmitted to and received from users, and various error counts.

The data communications multiplexer scans lines connected to the JOSS consoles and reports via a machine interrupt to the arithmetic processor when a character has been received or the transmission of an output character has been completed. All communication with the consoles passes through the multiplexer and down the I/O bus to the arithmetic processor.

In addition to the local JOSS console lines, the multiplexer has timing and other special gear necessary to interface with dataphones, connected to remote JOSS terminals as well as with local TTY's and TTY's on the TWX's network.

The data control handles the transfer of information to both the discfile and the magnetic tape. Because it cannot transfer data to more than one device at a time, usage must be shared.

The discfile provides long-term storage for users' programs and data. Its capacity of 200 million bits is sufficient for many thousands of user programs. Access time to individual records on the disc is generally about 200 ms, but long programs, queued use of the disc by several users, and computing commitments may extend an individual file or recall action to several minutes. Normally, however, a disc action takes about a second.

The IBM-compatible tape unit is used to collect accounting records and statistical information, which are processed in the general RAND accounting system on another computer. The discfile is periodically dumped onto tape for backup purposes, using an off-line program not contained in the regular JOSS software.

JOSS SOFTWARE

The JOSS operating software is divided into five principal parts: the interpreter, its arithmetic and function subroutines, the monitor (which also contains the drum, tape, and TTY console I/O routines), the distributor (JOSS console and TTY I/O), and the disc routines. All of this code is permanently resident in lower memory. Its total size is about 16,000 memory locations. See App. C for a detailed storage breakdown.

These principal parts of the JOSS system represent categories dictated primarily by the division of the programming effort. Although in the discussion to follow we will speak

about the software in terms of these units, the logical divisions of the software should also be kept in mind. These are much the same as those commonly found in hardware organization. Because JOSS is, indeed, a computer of a special type that is simulated on the PDP-6 through the software, these logical divisions should be expected: (1) an input-output unit for communicating with the users' consoles, discfile, drum, tape, and other I/O devices; (2) a central processing unit for interpreting and executing commands directed to the system by the users; and (3) a supervisory unit for coordination of input, output, and processing, so as to provide smooth unbiased service to the users and thus maintain the illusion of a single-server system.

On the figures in this report the three logical divisions of software are indicated by the symbols IOU, CPU, and SU. (See Fig. 2.) Some of the physical divisions provide more than one logical function, and some of the logical functions are spread over several of the physical divisions.

The interpreter, along with the arithmetic subroutines, is the part of JOSS that examines users' commands (statements) and computes answers in response to them. Users' programs are analyzed character by character by the interpreter to produce the indicated results instead of compiling user programs into executable code. User commands and data are carried in a linked-list fashion within variable-size user blocks. The entire user block must be in core during interpretation, and the RAND relocation hardware requires that this block must always occupy a contiguous area of memory.

The distributor and disc routines are trap time I/O packages that control transmission of data to and from the user consoles and the discfile. Both communicate with the monitor

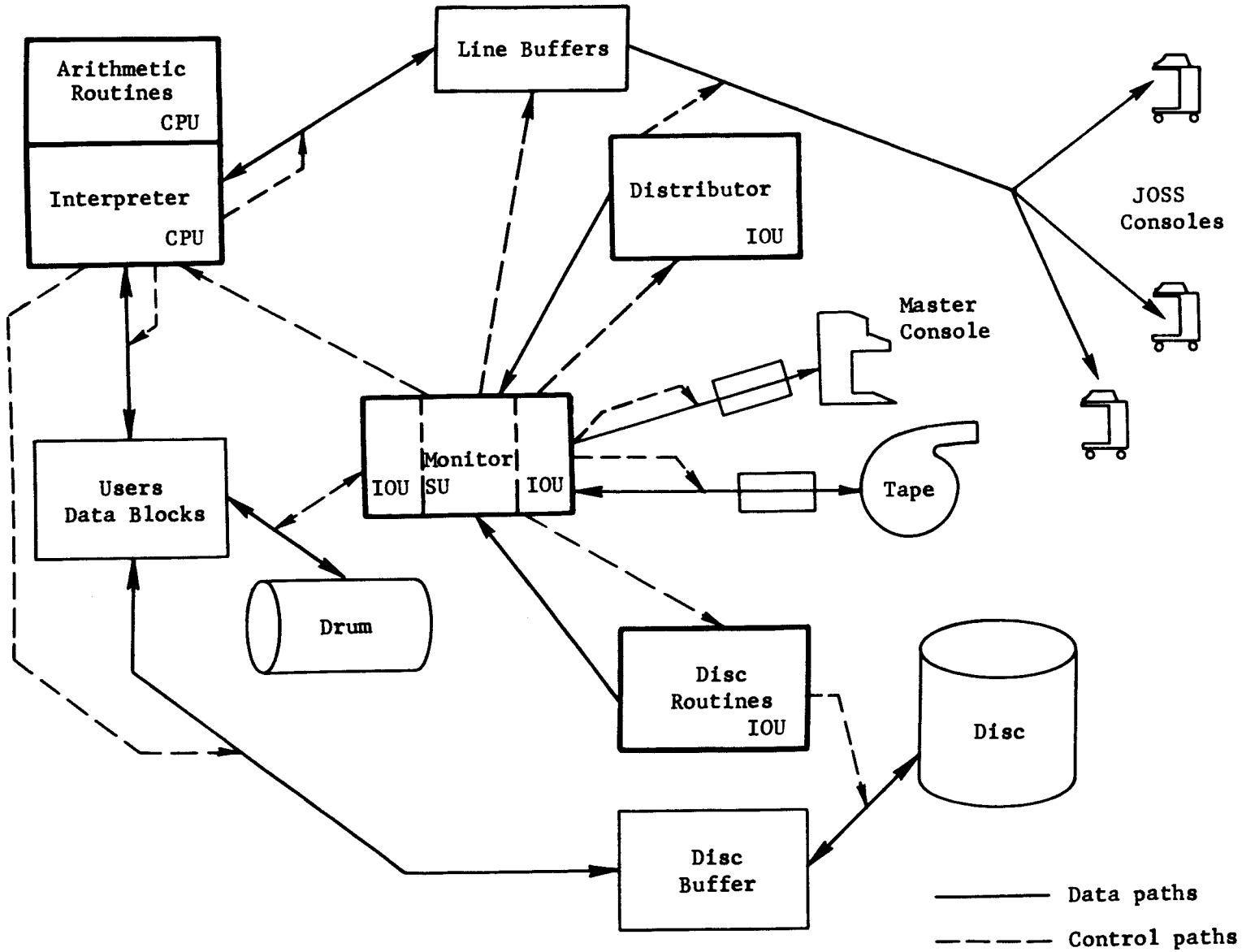

Fig. 2—Interrelations of JOSS software

about the I/O activity that they control through signals, which are software analogs of machine interrupts. These signals are referred to as "logical traps" or "logical interrupts." In addition to character I/O, the distributor carries out the backspace overstrike editing within individual input lines.

The monitor acts as a scheduling, resource-allocating, and synchronizing device, deciding when, and ensuring that, all data and hardware necessary for a particular action are simultaneously available. To carry out this process, the monitor maintains a series of queues of users in various activity states and of data for hardware devices. Signals from the other software components and a time-interrupt signal control both the changing of users from state to state and through these states the monitor's scheduling of the tasks of the system.

Also included in the monitor code are trap time routines to handle tape, drum, TTY console I/O, real-time clock interrupts; routines to gather and display performance statistics about overall JOSS operation; routines to provide accounting information to be input to the RAND computer time-charging procedure; and routines that act as processors on the level of the interpreter to supply the log-on procedures of receiving initials, project number, department name, and the final log-off process.

Figure 2 shows the interrelations of the JOSS software, including the data buffers that interface certain of the components. Control and data paths are shown, although it is sometimes difficult to classify a particular path as control or data.

The disc routines and the distributor, as well as the

portions of the monitor that deal with the drum, tape, and TTY console I/O, are trap-initiated routines; that is, they operate only in response to hardware interrupts from the I/O devices that they control. The routines are also entered to initialize an I/O action. These trap time routines generally signal completion of activity by setting switches or flags that are interpreted by the monitor as logical traps in its main processing loop.

Top-level control of the machine is shared between the monitor and the interpreter, with the bulk of time spent in the interpreter when users are requesting computation. The monitor regains control at least every 200 ms through a trap routine that sets the signal COMEBACK, the logical interrupt signal for the interpreter. During its control periods, the monitor adjusts user states according to the current logical signals, initializes I/O actions as may be appropriate, and determines, from the user states, which user should receive the attention of the interpreter next.

Some examples will clarify the overall system operation. Although these are the most common cycles, it should be realized that many other cycles and combinations of cycles occur.

Console Input Cycle

During the time a user is typing commands to JOSS, his console is in the green state and it is assigned a single line buffer in core. As each typed character is received by the machine, a hardware trap occurs, sending control to the distributor. The character is analyzed by the distributor, converted to internal form, and placed in the line buffer. If the character is "carrier return," a logical trap signal

is prepared and placed on a list of other such signals for the monitor. If enough of these signals have occurred to recall the interpreter, the COMEBACK signal is set.

When the monitor regains control, the logical signals are processed, which results in a change of the user's state, in this case from the low-priority green state to the high-priority carrier-return state. The monitor, in response to the high-priority state, either initiates action to bring the user's block of information in from the drum, or, if his block is already resident, changes context via the relocation register to that user and enters the interpreter. The interpreter analyzes the input line and as a result stores away an input step, performs an indicated short computation, or initiates a long computation continuing either to completion or until the logical interrupt COMEBACK occurs.

In the case of a short computation, the interpreter requests a buffer for the results and, after preparing the answer line, requests that it be transmitted and the station be switched to green. When the output to the console is complete, as signaled by the distributor at the time it transmits the last character of the line to the console, the console is provided with a buffer for input and is switched back to green, thereby completing the cycle.

Output Limited Cycle

An output limited cycle occurs when interpretation of the user's commands produces output faster than the typewriter can print it. To limit the number of output lines waiting to be printed at a console, the monitor defers buffer requests from the interpreter when the number of output lines reaches a limit called the choke number. As the typewriter catches up

and the number of waiting output lines is reduced to another limit called the unchoke number, the next buffer for output is delivered to the interpreter. The interpreter is unaware of whether the requested buffer is delivered immediately or at some later time. Speed of the console, interpretation rate, typical user operation, and desired signal response combine to determine the correct setting of the choke and unchoke numbers. Currently, they are two and one, respectively.

Compute Limited Cycle

Users who are computing (console in the red state) but are not output limited are termed compute limited. These users are considered together in a single compute queue, which, in absence of the need to service higher priority queues (for example, users who have just transmitted a carrier return), is serviced in a round-robin fashion, giving each user 200 ms of compute time before turning to the next in line. The 200-ms period is measured off by trap time routines that set the logical trap COMEBACK each time 200 ms have passed. The round-robin cycle is completed by returning the user who has just finished to the bottom of the compute queue and selecting the next user from the top of the queue.

Discfile Cycle

Information passing to and from the discfile moves through a single disc buffer either being filled from the disc under control of the disc routines and emptied by the interpreter into the user core area, or being filled by the interpreter and emptied by the disc routines. Only one user has access to the disc at a time; other users requesting disc activity are queued.

The monitor synchronizes control of disc activity to ensure that the user's data are in core when the interpreter is active and that the data control hardware is available when disc routine activity is desired. The user's data may be transferred to the drum while the disc routines are active in filling or emptying the buffer. Similarly, the data control may be used by the tape routines during periods when the interpreter is active with the disc buffer.

The logical control signals that pass between the various software components of JOSS are summarized in Fig. 3. Most of the signals are indicated by the setting of certain communication cells. Some signals, however, particularly the ones that start I/O action, are routine calls.

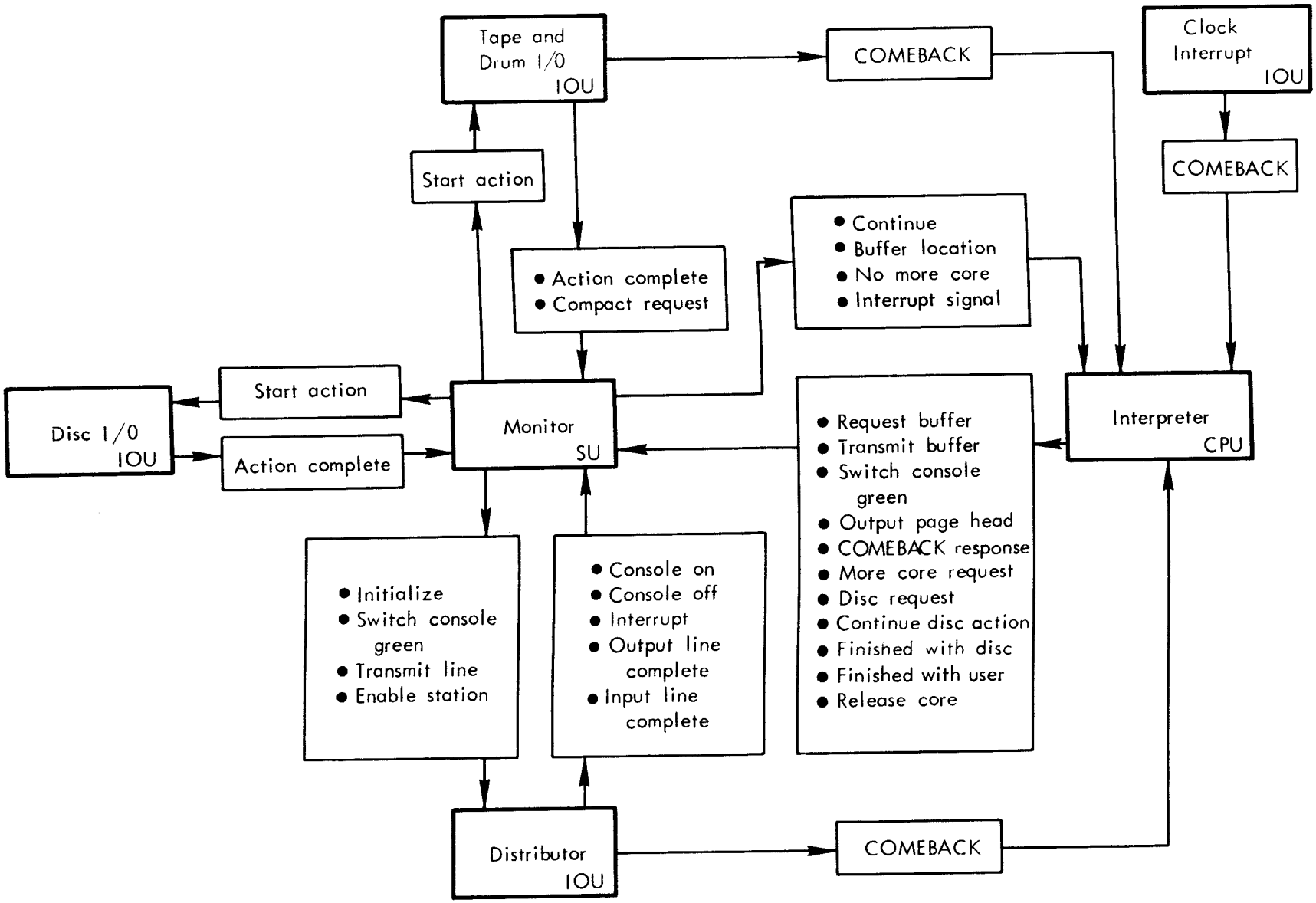

Fig. 3—Summary of logical control signals

III. SCHEDULING AND RESOURCE ALLOCATION

USER STATES

The monitor assigns each user in the system to a unique state. This state changes from time to time depending on the user's action at the console and the activity specified by his JOSS program.

The monitor maintains a queue associated with each state that orders the users in each state according to time of arrival. Thus, the monitor can provide first-come, first-served processing for each state. In addition, the state queues are grouped and ordered to establish priorities for selection of users for JOSS compute time, transfers to and from the drum, disc transfers, and other system scheduled functions.

The states are divided into a high-priority group composed of those users who are, in some way, requesting JOSS compute service, and a low-priority group composed of users who have no current need for computing. For example, in the high-priority group are users who have just released a line of input to JOSS by depressing the carrier-return button and those who have a JOSS program running currently, while in the low-priority group are those currently typing JOSS commands and those with output limited programs.

Exceptional states are those that list users waiting in the queue for JOSS service and users with consoles turned off. For bookkeeping purposes, a state is provided for the user currently being examined by the JOSS interpreter. The user remains in this state only until the monitor regains control.

Table 1 presents the user states in order of priority, the highest being listed first. The symbols given are those

Table 1

JOSS USER STATES

STATE QUEUE NAME	FUNCTION (Waiting for)
High-priority Group	
TOF	Accounting processing following turn off
ON	Core space and user log-on processing
RC	JOSS processing of a new input line
RI	JOSS processing following interrupt signal
RIB	JOSS processing following interrupt when a buffer is required
UC	JOSS processing when an output buffer is required
QC	JOSS processing following certain requests for more core
COM	JOSS processing of a continuing computation
CU	User currently receiving JOSS processing
Low-priority Group	
DCT	Synchronization of the data control device
DIP	Transfer of data to or from the disc
CK	Printing at the console
DQ	Access to the disc
GR	Input to be typed by the user
DSU	The final output line(s) before switch to green console
ABG	A buffer for switch to green console
QDM	Transfer to the drum because of need for core size increase
QM	Access to JOSS (active user limit exceeded)
OF	Console to turn on

used in the monitor code. The process for which users are waiting, according to their state, is described under the heading "Function." Descending priority represents a descending requirement for core space in the machine, and the monitor makes adjustments by swaps to and from the drum to replace low-priority users in core with high-priority users from the drum.

A number of features of the state list that contributes to the select-for-compute and select-for-swap algorithms should be noted:

1. The states of the high-priority group are the only ones that have a need for computation. External events bring low-priority group users into the high-priority group as they require compute service.
2. The states QM and OF are such that they never have core assigned, nor are they on the drum.
3. States of higher priority than COM are transient. As soon as they are given compute time, they revert to one of the lower priority states. Therefore, these states are never considered for transfer to the drum. Because they are so transient (meantime ~ 50 ms), the priority ordering is not particularly important.
4. The low-priority group is ordered from the top down by increasing expected time in the state. For example, meantime in-state is expected to be:

DCT	1.5 ms
DIP	100.0 ms
CK	2.0 sec
DQ	3.0 sec
GR	30.0 sec

DSU and ABG have times similar to GR with a small increment, while QDM is lowest because a transfer to

the drum is required. Thus, in selecting a user for transfer to the drum, the search of the state lists is from the bottom up.

5. Data transferring to and from the disc move through a dedicated core buffer; therefore, during these transfers (DIP state) the presence of the user's data in core is not required.

USER STATUS DATA

The monitor maintains a cell that contains status information about all possible JOSS users. The information includes the user's state, the number of core blocks currently assigned to the user, his base core location, and the occurrence of certain special situations. The exact contents of the status cell are as follows:

S.STA	The user's current state
S.TM	Current time within compute quanta
S.COR	User's core location (zero if not in core)
S.BLOCK	Number of blocks of core assigned to user
S.OFR	Flags receipt of an OFF signal during a disc action
S.INR	Flags receipt of an interrupt signal
S.DU	Set when using disc

CORE MAP

Sixteen cells map the sixteen or fewer blocks of core assigned to user data. Examination of these cells gives the current status of core with regard to user assignment and drum use. Each core map cell contains information such as the following:

S.IU	Flags the core block in use by a user
S.ID	Flags the block in use by the drum
S.UR	Gives the user number assigned to the block. The user number is carried only in the first block of a multiblock group.

BUFFERS AND BUFFER QUEUES

Lines of text transmitted to and from JOSS consoles pass through buffers that are always resident in core. The format of the buffers is:

Word 0	Buffer header containing a pointer to the next associated buffer
1	Pointer word used by the distributor
2	Count word used by the distributor
3-18	7-bit ASCII text

All buffers not in use are chained through pointers in the buffer header to an available buffer list. Buffers in use are attached to headers corresponding to the user who is typing into or receiving output from JOSS. During input a single buffer is attached to a user but on output more than one may be attached, the top one on the list being the one in active output.

These same buffers are used to transmit output to tape and to the console TTY. A transmission is initiated by the routine that attaches the buffer to its proper header, as determined by a context number present on entry to the routine. This context number is equal to the console number for JOSS stations and is -1 for output to the console TTY and -2 for output to tape. Different I/O start routines are required for different context numbers. Start routines operate only if the

header is empty on buffer presentation. Otherwise, it is assumed that a start has already taken place and the end signal from the transmission will provide subsequent starts.

USER STATE TRANSITIONS

During JOSS operation, users are sequenced through the monitor states in response to signals from the consoles, signals from the interpreter (in response to the executing program), and a time signal generated by the hardware.

A diagram of the transitions between states is presented in Fig. 4. Although the figure gives an excellent impression of the overall operation of the monitor, it is by no means complete. Reference to the actual code must be made if the exact details of state transitions are required.

Several subcycles are identifiable in the figure. At the top left of the diagram are the states that relate to console turn on and turn off; at the top center, the states applying to program input and short computations; at the lower left, the typewriter output limited sequence; at the lower center, the compute limited sequence; at the lower right, the disc activity sequence; and at the upper right, the drum transfer sequence, used in certain cases when more core blocks are needed by a user.

The states are broadly divided into two priority groups: a high-priority group, shown with heavy shading, and a low-priority group. Users whose states are in the high group require interpreter service, while those in the low group do not. Signals from the consoles or from the user's program cause changes from one state to another.

The ON and OFF sequence provides mechanisms for calling the separate software processors that provide the user's initial

core block and that handle the initial salutation, including receipt and checking of the user's initials, project number, and department, and the writing of the accounting record at turn-off time. At turn on, the user is placed in QM state if the system limit on the number of users has been reached. Under most operating conditions, this limit is set higher than the number of consoles so that the limit is never reached. Before the installation of the drum, the limit was lowered to a value that would keep all active users in core memory. When the user turns his console off, his state is changed to TOF, except when a disc action is in progress. In the latter case, the OFF signal is flagged in the user's status word, and his state remains unchanged through completion of the disc action. At that time, the change to TOF state, production of final accounting records, and the reenabling of the console occurs.

During periods when a user is typing program steps and doing short computations via direct commands, he is cycled through the states in the Input and Short Computation Sequence. During typing the console is green and the user's state is GR. As soon as the carrier return is pressed, his state changes to RC and subsequently to CU for interpretation of the line just typed. If the line is accepted without comment, the state returns directly to GR while error messages or response lines return to GR via DSU during the time that the typewriter is printing the output line. Attachment of a buffer to each green user is required for receiving his input. In the event that none are free, the user's state is changed to ABG until a buffer becomes available.

The Output Limited Sequence controls those users whose programs produce output faster than can be printed at the

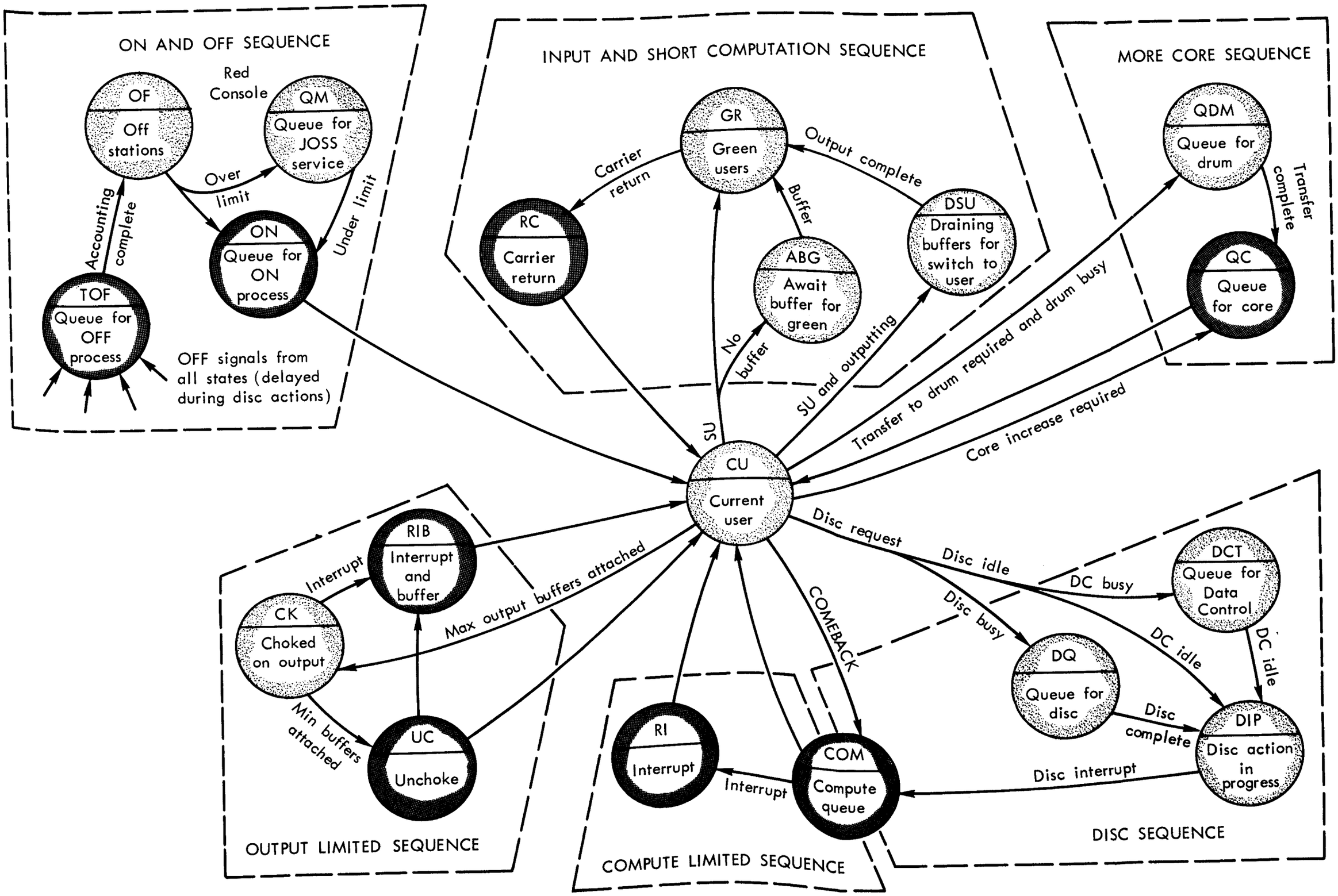

Fig.4—JOSS user state transitions

console. One of the system parameters is called the choke number. When the number of lines of output ready for printing at the console equals this number, the user's state is changed to CK and further computation ceases. When the number of output lines is reduced to the value of the unchoke number, the user's state is changed to UC and computation is resumed. If, during this sequence, the user presses the interrupt button, his state is changed to RIB and computation is resumed for response to the interrupt request.

In the Compute Limited Sequence, those users in the COM queue share the computer in 200-ms time slices or "quanta," as controlled by the setting of the signal COMEBACK through the hardware clock interrupt. During these sequences, the user's state is changed to RI if he presses the interrupt button.

The purpose of the Disc Sequence is to queue users for access to the disc, allowing only one user on at a time, and to synchronize the use of the data control device, which is also used for transfer of information to tape. The major cycle is CU-DIP-COM with actual disc activity occurring only during DIP. During COM the disking user waits his turn for the attention of the interpreter, and, then, in CU the disc buffer is either filled or emptied depending on the command in process. Other users requesting disc service during this time wait in DQ, while the disc routines wait in DCT if the tape unit is using the data control.

The More Core Sequence is employed when a user requests an additional block of core storage (implicitly through the interpreter) and no idle blocks are available in core. In this case, the user is transferred to the drum and his state is changed to QC, a high-priority state that forces the swap

algorithm to make core space available and restore the user to the main memory. If the drum is busy when the request is made, the user's state is changed to QDM where he waits for transfer to the drum.

DYNAMIC SYSTEM PARAMETERS

There are a number of parameters that set the dynamic operation of the system. Adjustment of these parameters provides, in some measure, a "fine tuning" of the system's operation to meet desired objectives of response time and efficiency. The coding that examines many of these parameters is such that it trys to bring the system's operations to that level specified by the parameters' current value. Some of the more important parameters are listed below:

N.SON Maximum number of stations allowed on before giving a "queue for service" message.

N.CK The "choke number." Maximum number of output buffers allowed a station at one time. Computation is suspended when this number is reached.

N.UC The "unchoke number." When the number of output buffers attached to a station is reduced to this number, computation is restarted.

T.MAX The standard compute quanta. JOSS time-shares among computing users, giving each user this amount of time (currently 200 ms) before going on to the next.

SG.LIM When the distributor has placed this many signals on the distributor-monitor signal list, it sets the COMEBACK signal to recall

the interpreter. Currently the limit is reached on every CR or on a total of five signals of the other kinds.

N.CB The maximum number of 1024-word core blocks to be assigned to any one user.

N.C Maximum number of core blocks that may be used by the system. The low-numbered blocks are used. This parameter is of value primarily for system debugging.

S.OK The monitor ignores signals from all stations numbered higher than the number contained in this cell. The parameter is used only for system debugging.

SCK When set causes all drum transfers to be checksummed for detection of transmission errors.

MAIN PROCESSING LOOP

The JOSS monitor main processing loop (MPL) provides for action on a number of possible asynchronous events. These events or signals are either such that no machine interrupt is available to flag the event or are such that processing at the time of occurrence would be impossible, improper, or inconvenient. The deferral of these actions to the main processing loop ensures that certain routines need not be trap-protected or coded as pure (reentrant) procedures.

Figure 5 outlines the flow of control for the major functions acted on in the main processing loop. There are two main paths, JOSS active (processing a user's request) and JOSS idle. The idle path has a minor branch not shown on the figure that distinguishes idle passes when the drum

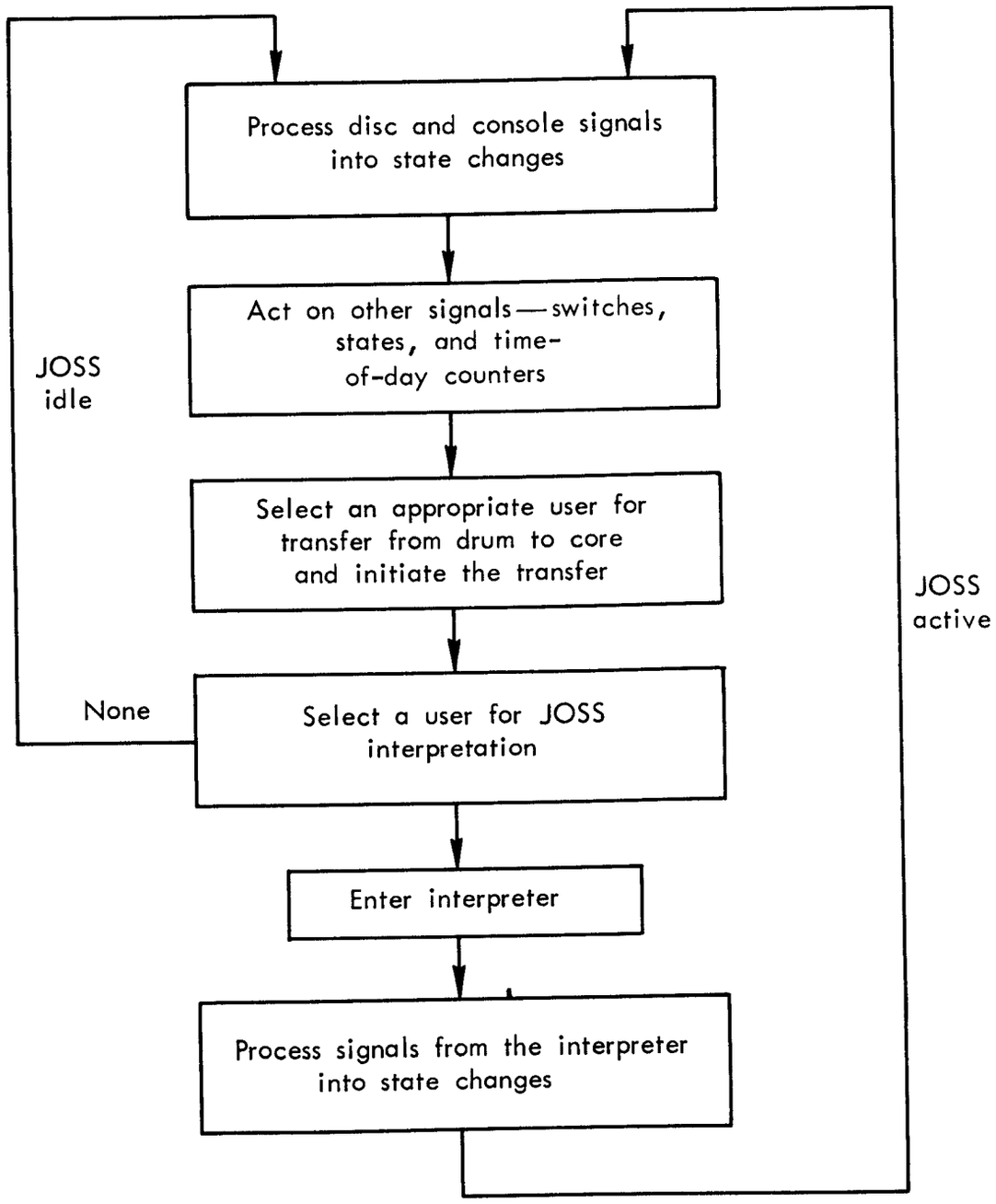

Fig.5—JOSS monitor main processing loop

is busy. This indicates that a user currently on the drum has requested service and no other activity could be performed in overlap with the drum transfer. Usually this happens when there are many users but a very light computing load.

The interpreter releases control to the monitor whenever any of the monitor-provided facilities are desired and also when the signal COMEBACK is set. COMEBACK is set by the occurrence of monitor-distributor signals and every 200 ms by a real-time clock interrupt. Thus, when the interpreter is processing long computations, the monitor regains control in order to service signals from other consoles and to time-share the use of the machine among compute-bound programs.

Console signals are produced for the monitor by the distributor. The signals and the resulting processing by the monitor are described in the next section on distributor-monitor signals.

The disc interrupt signal, DISC.S, is set by the on-line disc routines when a disc buffer is either filled during an input action or emptied during the filing of user information on the disc. The state of the user with disc action in progress is changed to compute (COM) so that the interpreter can proceed with the filling or draining of the next buffer load of information. This ensures that the user's block will be in core during the necessary points in the disc transfer without requiring it during the entire transfer.

Ordinary disc interrupts set the signal DISC.S to one. When a discard is in progress, it is set to two, and the monitor prepares a tape record that records the charges for disc use.

A final mode of processing is provided by a monitor flag that indicates the monthly production of disc accounting records. This essentially disc-to-tape operation is synchronized through the logical trap DISC.S and a similar one, SKT, produced by the tape routines.

The miscellaneous functions performed in the main processing loop can be divided into four broad categories: switch examination, dormant queue service, core compaction, and periodic functions.

Two special switches are monitored: one to enable the system shutdown procedure and the other to forcefully shut down the system by simulating OFF signals from all current users. The two procedures are described fully in Sec. V.

Certain user states are entered to await the occurrence of particular events. The main processing loop examines the queues corresponding to these states and, if a user is waiting, checks further to determine if the appropriate condition has been obtained. If it has, action is taken. Queues serviced in this way and the resulting actions are as follows:

DCT Disc user waiting for use of the data control. When the data control becomes idle, the user state is changed to DIP and the disc routines are entered.

ABG Users awaiting a buffer for switch to green.

QDM Users waiting for the drum to idle in order to be transferred there.

QM Users in the JOSS service queue. Users in this queue are given service whenever the number of active users falls below the allowable number.

During some kinds of drum swaps, more than one small-size user may be transferred to the drum in order to make room for one large user. The users to be transferred are selected on the basis of their priority. Thus, there is no guarantee that the core space freed is contiguous, which it must be to read in the large user. Compaction of core is performed at the main processing loop level to bring all available core blocks adjacent to one another at a time when they are not in use by the interpreter. The waiting transfer from the drum to core is then initiated.

Major time-interval incrementing--the counting of minutes, hours, days, months, and years--is also done from the main processing loop. Functions performed in this part of the MPL include initiating log reports each minute, accumulating statistics on the minute and hour, and initiating accounting for the disc each month.

Next in line in the MPL is the selection of an appropriate user to be transferred from the drum to core. This action is only performed if a transfer is not already in progress. Transfers are made only for high-priority users as described in detail below.

An in-core user is then selected and the interpreter entered. The selection for "execution" is described below.

The main processing loop is completed when the interpreter returns to the monitor. The actions performed on receipt of each interpreter signal are described in the section on monitor-interpreter signals.

DISTRIBUTOR-MONITOR SIGNALS

As we have seen, signals or logical traps are produced by the distributor in response to certain user console actions.

These signals are placed in a communication table by the distributor and processed from this table by the monitor in its main processing loop.

Several reasons contribute to the decision to use logical traps of this form. Primary among these is the reduction in the amount of processing that must be done at hardware trap time. This reduces the amount of context that must be saved and restored by trap supervision routines and minimizes the number of routines and data that must be protected against recursive use or simultaneous change or reference. In addition, the scheme provides a sharp programming interface that defines responsibility and eases the debugging problem. Finally, the method is sufficient to provide the desired real-time response. Items that must be performed quickly, such as character transmission within a line, are done at trap time, while interline process with less demanding response requirements is left to logical trap processing.

The distributor provides five logical event signals for the monitor as follows:

- TO Signals the transmission of the final character of an output line to a console.
- IN Signals the receipt of an interrupt signal from a console.
- ON Signals the receipt of the power-on signal from a console.
- OFF Signals the receipt of the power-off signal from a console.
- CR Signals the receipt of a carrier-return (end of input line) signal from a console.

Each signal is placed on the signal list by the distributor together with the number corresponding to the station (user) from which it was received. The monitor removes signals from the list as they are processed.

Seventeen different actions are performed by the monitor as a result of the combination of an input signal and the user's current state. These range from an error halt and no action to a change of state that will precipitate appropriate action.

Table 2 is copied directly from the monitor code. For each combination of signal and state, it gives the number

Table 2

SIGNAL-STATE TRANSITIONS

	Signal				
State	TO	IN	ON	OFF	CR
TOF	1	1	1	1	1
ON	1	1	1	9	0
RC	0	6	1	9	0
RI	13	1	1	9	0
RIB	13	1	1	9	0
UC	13	5	1	9	0
QC	13	6	1	15	0
COM	13	4	1	15	0
CU	0	0	0	0	0
DCT	13	6	1	14	0
DIP	13	6	1	14	0
CK	12	5	1	9	0
DQ	13	16	1	14	0
GR	0	1	1	9	3
DSU	11	1	1	9	0
ABG	13	1	1	9	0
QDM	13	6	1	15	0
QM	13	7	1	8	0
OF	1	1	10	1	1

of the appropriate response routine. Description of each response routine is given below.

The tabular method in which the coding is done contributes to the ease of debugging the program, because it tends to ensure that all of the possible combinations are examined. Without the table it is easy to overlook rarely occurring combinations. It is difficult, for instance, to think of the conditions under which a user who is currently actively transferring information to or from the disc (DIP) might produce an output line termination signal (TO). The condition occurs when a direct discfile command is given on the last line of the user page. The TO from the page heading line may occur during the disc transfer.

The signal-state action routines are listed below:

0 Machine hardware or programming precludes the occurrence of this combination. Thus, if all programs are bug free, a machine error has occurred. To date halts here have only detected program bugs.

1 The signal is ignored. Either it is a recurrence of one already recorded, or the requested action is already being performed, or the system cannot perform the required action, and furthermore it cannot even report to the user that it is unable to perform the action.

2 Unused. A remnant of a bygone day.

3 Respond to carrier return from a green station. Some statistics (e.g., interarrival time of carrier returns: the user interaction rate) are gathered and the user's state is changed to RC.

4 Interrupt from a computing user. Change state to RI.

5 Interrupt from a user whose program has an active request for a buffer. Change to RIB state.

6 Interrupt from certain other states. The signal is recorded but no state change is made.

7 Interrupt from a user who is in the queue for JOSS initial service. The queue message is sent to the user if a buffer is available. Note that the distributor inhibits more than one interrupt signal in this state.

8 OFF signal from a user in the queue for JOSS initial service. The user's state is changed to OF, his console is reenabled unless the system is shutting down, any output buffers that he may have are returned to the buffer pool, and messages are sent to all others in the queue to reflect their change of position on the queue.

9 OFF from all others except those using the disc. State is changed to TOF, the number of active users is decremented, buffers are returned to the pool, and the queue for JOSS initial service is examined for the possibility of giving service to the top user in that queue.

10 ON from an off station. The user's state is changed to ON unless the current number of users is at or exceeds the allowable maximum, in which case the user's state is changed to QM and, if buffers are available, a queue position message is sent to him.

11 End of output line signal (TO) from a user in DSU state. Statistics about the line just trans-

mitted are gathered, the buffer is returned to the available pool, and if more output buffers are attached their transmission is initiated via the distributor routine C27. If all output buffers have been transmitted, the final buffer is blanked in preparation for input, the user's state is changed to GR, and the console is switched to green via the distributor routine C28.

12 End of output from a station whose output buffer limit has been reached (CK or choke state). Statistics are gathered, the buffer is returned to the available pool, transmission of the next buffer is started, and if the number of buffers now attached to the user is less than or equal to the unchoke number his state is changed to UC, thus requesting further processing by the interpreter.

13 End of output from most other states. Statistics are gathered, the buffer is returned to the available pool, and any remaining buffers of output are started. State is not changed.

14 OFF signal from a user currently filing on, discarding, or recalling from the disc. The signal is recorded in the user's status word. State is not changed. We always allow completion of the disc action before responding to the OFF signal.

15 OFF signal from a user requesting more core space. If the user is using the disc, routine 14 is performed. Otherwise routine 9 is followed.

16 Interrupt from the disc queue (DQ). The disc usage flag is reset and state is changed to RI.

Some notes on the signal-state table and the associated routines are given below:

- Since CU state is a bookkeeping device used by the monitor only when the interpreter has control, no user should be in this state when the distributor signals are processed. Thus, the error halts for all signals in this state.
- ON signals may occasionally occur from stations currently in processing of some kind, particularly from stations operating over dataphones where the signals may be due to hits on the line, or from stations whose power plug has been pulled loose from the wall. The distributor reenables these stations, if necessary, and the monitor ignores the signals.
- In TOF state the user is waiting for processing of final accounting records. Because the station is disabled, no signals of any kind should occur, but if they do, no harm can result in ignoring them.

MONITOR-INTERPRETER SIGNALS

The interpreter's requests for service to the monitor fall into three broad categories. The first category consists of requests for resources and includes requests for output buffers, more core space, and disc action. The second category reports user status; for example, still computing, finished with the disc, and ready for a new input line from the user. The final group includes special actions; for example, send a page heading line to the user and release all of user core (the final result of an OFF signal).

The interpreter is reentered at once if the request can

be satisfied immediately. Otherwise, a change of state reflects the request or status report and the monitor enters its main processing loop. Thus, some requests (perhaps, for another buffer) may not be honored until a later time.

The monitor requests control from the interpreter via a single signal called COMEBACK. This signal is set by the clock trap routine every 200 ms, and by the distributor if enough (as defined by a system parameter) signals from the consoles have been received. The interpreter checks COMEBACK at frequent intervals in the course of processing, but always at points where all context regarding the user is in the user's data area. Thus, a switch to a different user or a transfer of a user to a new core location or to the drum may be accomplished without hazard whenever the monitor has control.

The entries from interpreter to monitor are listed below, together with a description of the monitor's actions. Control is returned to the monitor main processing loop (MPL) unless otherwise specified.

0. Switch console control to the user. If buffers are attached to the user, his state is changed to DSU and the main processing loop is entered. If no buffers are attached, one is obtained from the available buffer list, and it is blanked and attached to the user; the user's state is changed to GR; and the console is switched to green via distributor routine C28. If no buffers are available, the user's state is changed to ABG and the main processing loop is entered.
1. Request a console output buffer. If the number of buffers currently attached to the user has

reached a system limit called the choke number, his state is changed to CK and the main processing loop is entered. If not, and a buffer can be obtained, it is delivered to the interpreter. If no buffers are available, the user's state is changed to UC and MPL is entered.

2. Return a buffer. The buffer is attached to the available buffer list, and the interpreter is reentered.
3. Transmit a buffer (line of output) to user. The buffer is attached to the user and if transmission is not already in progress, it is started via distributor routine C27. Return is to the interpreter.
4. Transmit a buffer and switch to user. A combination of entry 3 followed by entry 0.
5. Continue computing when appropriate. The interpreter returns to this entry when processing is interrupted because COMEBACK was set. If the processing has continued for at least the "shot time" (200 ms), the user is placed at the end of the COM state queue and MPL is entered. If he has used less than the "shot time" (because COMEBACK was set as a result of an excess of console signals), he is placed at the top of the COM queue and MPL is entered.
6. Request disc action. If the disc is busy, the user's state is changed to DQ. If the disc is not busy, the interpreter is reentered for disc initialization.
7. Continue disc action. This entry is used when

the next disc buffer load of information has been prepared. If the data control device, which is used for transmission of data to the disc, is not busy, the user's state is changed to DIP and the disc processor routine DISC.C is entered. If the data control device is busy, the user's state is changed to DCT.

8. Disc action complete. The queue for disc (DQ) is examined and if there are users in that queue, the top user is changed to COM state. The interpreter is reentered unless an OFF signal has occurred during the disc transfer, in which case the user's state is changed to TOF.

9. Request another block of core. As users run out of space, core blocks are provided in increments of 1024 machine words until the limit contained in the cell N.CB is reached. The algorithm for providing more core blocks is described below.

10. Imperative request for more core. This entry is used in the process of typing user disc dictionaries. An additional block is provided unless the user's total size exceeds that of available core.

11. Page heading. A page heading line is made up in a buffer provided by the interpreter and transmitted to the user. Time, date, channel number, initials, project number, page number, and the system administrative message, if any, are placed in the heading line and spaced ap-

propriately for JOSS console or TTY output. Return is to the interpreter.

12. Release user core. This is entered by the final accounting processor to release user's core following an OFF signal. State is changed to OF. The completion of the OFF action is logged on the console TTY if the appropriate switch is set.
13. Return a block of core. The high block of core assigned to the user is made available.

EXECUTION SELECTION

Given the priority ordering of users indicated by the state queues, selection of the next user for interpretation is simple: We scan the queues of the high group from the top down, selecting the highest one in core but not in the process of transfer from the drum. Thus, we serve on's and off's before carrier returns, carrier returns before output limited users, and output limited users before compute bound ones. A few details of the operation need to be noted.

Users who have just turned on are a special case, because they need to be provided with a free block of core rather than have a current block swapped from the drum. This may be supplied by the drum swap routine, clearing one user to the drum without the usual read in. On as well as off are also special cases, because a different interpreter is used for each of these processes. The execution selection routine recognizes these special cases and sends control to the proper processor.

Other actions that are provided at this time, as indicated by the state, are the fetching of a buffer for RIB and UC states and setting of the interpreter's interrupt signal when the state is RI or RIB. The interrupt signal is also set in cases

in which the receipt of an interrupt signal does not change the user's state. An example is when the interrupt signal is received after a carrier return but before the interpreter has been recalled in response to the carrier return.

INCREASE IN USER'S CORE SIZE

Each JOSS user is assigned 1024 words of memory when he turns on his console. As his stored program and data increase beyond the 1024-word bound, additional blocks are assigned in 1024-word increments until the system-set limit is reached. This increase in size is accomplished through a monitor-interpreter dialogue without the involvement or knowledge of the user. Although the system is coded for a maximum of sixteen 1024-word blocks, the normal limit is set at four blocks, corresponding to a JOSS "size" of 1906 cells for users. It is believed necessary to hold the limit at the four level in order to maintain a sufficiently fast response for a large number of users (say, 100), but further experimentation should be considered.

The algorithm for increasing the number of blocks assigned to a user consists of several steps in which attempt is made to satisfy the request, first in core, and then by a drum swap. Of course, a test is first made to see if the limit has been reached; if so, the request is denied. The steps in the more core procedure are:

1. Test for a free block immediately above the user requesting additional core.
2. Test for a free block of the total required size anywhere in core. If found, move the user there.
3. Test for a free block in core below the user. If found, move the user and all users between

him and the free block down one block, putting the free area immediately above the user.

4. Test for a free block above and if found move users up to bring the free block adjacent to the user requesting core area.
5. If all of the procedures given above fail, size cannot be increased in core. The user's size record in his status word is incremented and his core area is transferred to the drum. His state is changed to QC, a high-priority state that forces the drum swap routines to find a proper transfer that will bring him in. If the drum is busy when the request is made, a special queue, QDM, is used to indicate that transfer to the drum is required.

DYNAMIC MEMORY ASSIGNMENT

Some of the tables described above contain the data necessary for the monitor to control memory assignment and to make the decisions necessary when that assignment must be changed to facilitate response to a user's command. Since no prescan of the line is performed by the monitor, every line of input must be presented to the interpreter. Furthermore, every entrance to the interpreter must be accompanied by the presence in core of the user's block of data. In fact, the interpreter is entered through a cell in the user's block.

At first one might think that some lines of input, say, blank lines and comment lines, might not have to go through this process and might be ignored without calling on the interpreter and swapping user's data to core. The maintenance

of the $ counter (count of lines on user's page) is the counter example; each input line must at least increment this counter in the user's block.

The two tables that carry the memory assignment records are (1) the core memory map, CORE, which carries a user busy bit, a drum busy bit, and the user number; and (2) the user status table, S.S, which carries the core location and the number of core blocks required by the user. Figure 6 shows the two tables and how they might cross-link with each other at a particular moment in time.

Each cell in the core map corresponds to a 1024-word core block in user core memory. The first user shown is currently occupying the first three blocks of user memory, the second user is on the drum as indicated by the zero in

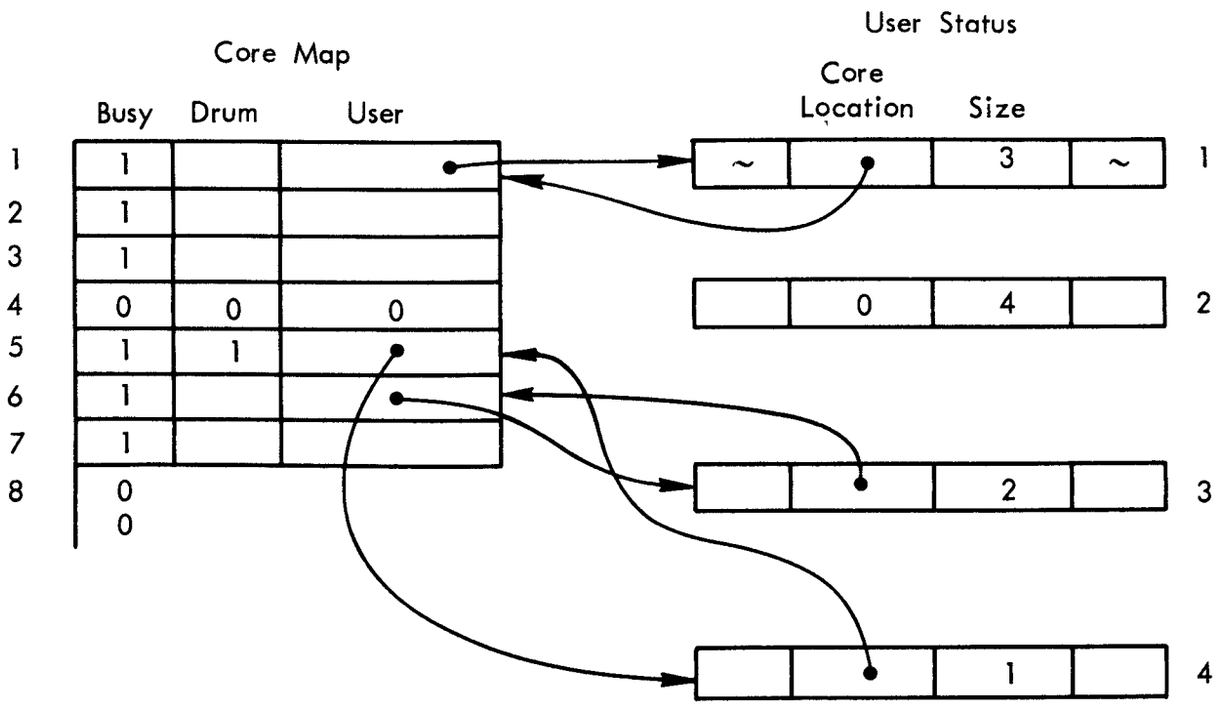

Fig.6—Dynamic memory assignment

the location part of his status word, the third user occupies blocks six and seven, and the fourth user occupies the fifth block. From the core map, it is apparent that no users occupy block four or blocks eight and beyond. Block five carries a drum busy bit that indicates that the fourth user is being transferred to the drum. (If the transfer was reversed, the location part of the status word would still be zero.) The double pointer from core map to user status and back again is, of course, redundant, but the information is easy to carry and avoids a search. When rearrangements of core are made (relocating user blocks from one part of core to another), updating of the location part of the user status word need not be done until all such movements are complete.

One ground rule needs to be reiterated: For any user action, a user's block of information must be in core and, if he requires more than one block, all of his blocks must be located in contiguous core space. Consideration of the linked-list internal structure of the user's block, together with the value of some degree of execution speed, dictates this requirement.

SELECTION FOR DRUM TRANSFER

The algorithm that controls transfer of a user block to and from the drum is outlined in Fig. 7. The flow chart should be examined in conjunction with the table of priority state queues (Table 1, p. 15). Each drum transfer is treated as a complete, uninterrupted sequence of events; that is, we decide who to bring in and who to send out in order to provide the core space. Even if events during the transfer

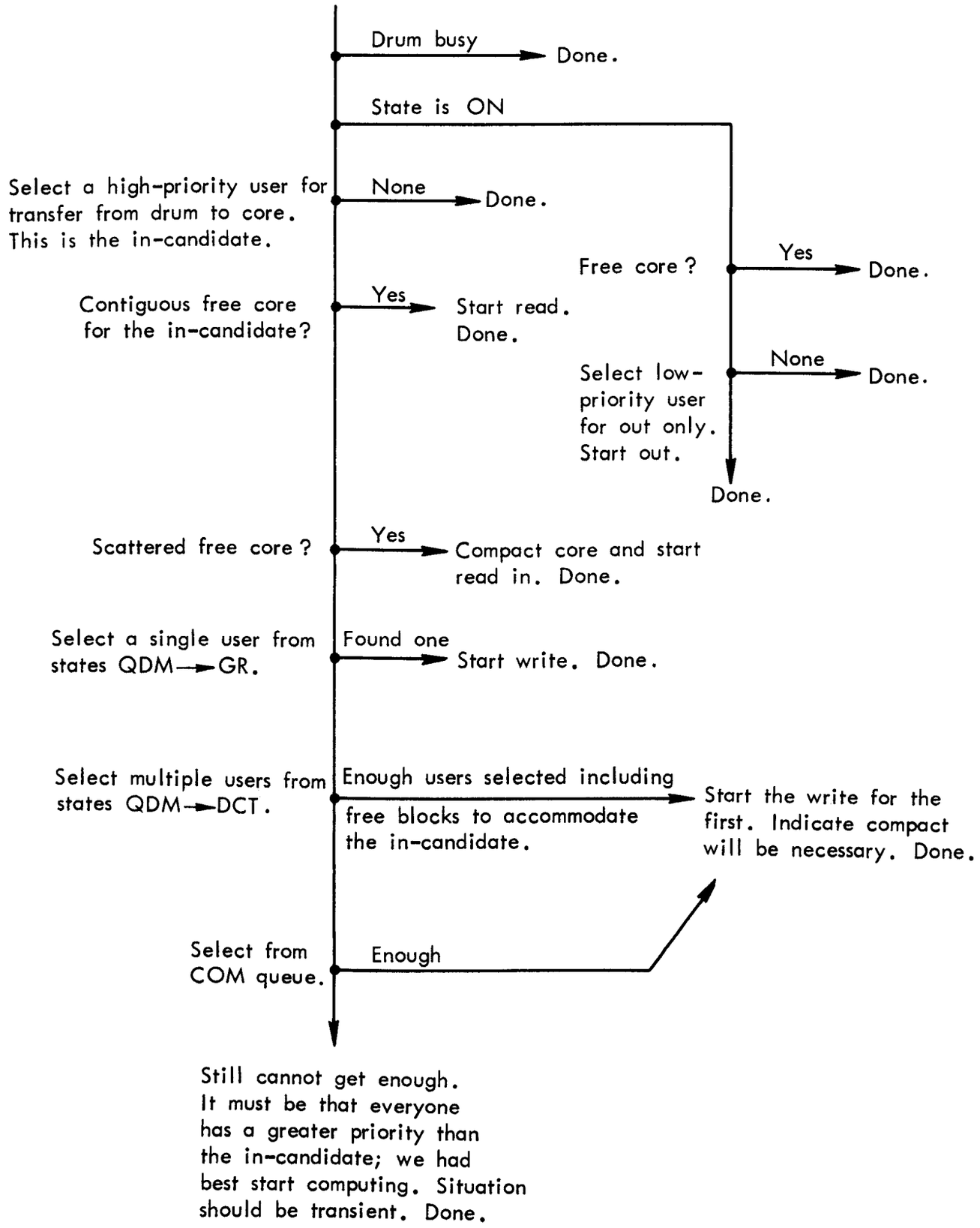

Fig. 7—Drum swap selection

make an alternate user a more appropriate candidate to bring in, it is not done; the new event will generate a new transfer later on.

The rather low average rate at which users change state from low to high, thereby generating a need for swap, makes our choice a reasonable one. The average time for a complete swap (writing one or more users on the drum and reading in the selected user) is about 60 ms. Because we expect an interrupt requiring interpreter response only about every 500 ms (from 30 users), it is clear that very little can be lost by not reselecting in mid-swap.

Accordingly, the selection algorithm is not entered if the drum is busy, as indicated by a flag set by the selection routine once a transfer has started. Two other flags, not shown on the flow chart, control entrance to the routine: First, it is always entered if there are users who have just turned on. In this event, a full swap is not mandatory, but clearing a free area of core is required as shown in the first major branch of the flow diagram. Second, a count of the number of users on the drum is examined. If none are on the drum, no swap or swap selection is needed.

The high-priority users--those in states TOF through COM--are examined; if one is found to be on the drum, he is the "in-candidate." Once the in-candidate is selected, a series of tests is made to determine the most economical way to perform the necessary swaps. As shown on the flow chart, we first try to find free core blocks, even if they are scattered, thus requiring only a drum read. Next, we try to find a single user in the lower priority states QDM through GR who has at least as many blocks as the in-

candidate. There is some question as to whether the queues should be searched from bottom up or from top down. The critical queues are GR and DSU, which may be considered together since DSU is printing the last output line before a switch to green. Examination of the interarrival times of carrier returns from individual stations shows that they are distributed approximately exponentially with a mean of about 30 sec. In other words, the expected time until a user hits carrier return, given that his state is green, is 30 sec and is independent of how long he has already been green. Thus, inasmuch as the exponential approximation is valid, it makes no difference in which direction the search is made; in fact the queues are searched from top down.

On failure to find a single user to transfer to the drum, we try to make up a list of low-priority users, which, together with any free core that may be available, will free enough space for the in-candidate. All queues in the low group are searched. Most often this process will free scattered core blocks and will require that core be compacted between the out and in portions of the swap in order to make contiguous core blocks available to the in-candidate.

The final possibility of setting up a swap lies in the compute queue (COM). Here we must take more care to account for the case in which all users are computing and there are more users than can fit in core at once--for example, five users computing, each requiring 4000 words of core, and 16,000 words available. We wish to serve these users in round-robin fashion, giving each a 200-ms quanta of compute time before proceeding to the next.

In this case, both the in-candidate and the out-candi-

date(s) come from the same queue and its detail as a priority list is important. We search for out-candidates from the bottom up of the compute queue, but only as far as the in-candidate. If we get as far as the in-candidate without finding a user to transfer, there is no user on the drum who can replace a user in core. Also, there must be plenty of work to do on users already in core. As the users in states of higher priority than compute are serviced, their states change to some lower state and as each compute user gets his 200-ms quanta, his priority is lowered by placing him at the end of the compute queue. Any of these actions may make a swap possible at a later time.

TIMING FACTORS FOR SYSTEM RESPONSE

One of the prime goals of JOSS is to provide very high speed response. The perceived delay at the user's console between request for a computation and the typed result should be as small as possible--unnoticed if possible. To accomplish this goal, careful control over the details of the I/O activity must be achieved. Figures 8 and 9 display the detailed timing for two cases of response. In each figure the sequence of events moves in time from left to right across the chart. The actions of the various equipment and the signals between computer and console are shown for each device as they key each other to action.

Each chart begins on the left as the user strikes the carrier-return key. Figure 8 shows the actions and signals that result when the input line is blank or when the line's contents are properly responded to by switching the console to the green state. An example is provided by the input line "x=3.1415."

Following certain mechanical actions at the typewriter, the console transmits the carrier-return signal to the computer. The distributor portion of the JOSS code responds to the receipt of this signal by transmitting to the console a "request status" character. At the same time, it sets the logical trap signal COMEBACK to request that the interpreter return from the processing of the current task in order that the monitor may schedule processing time in response to the just received input line. Typically, the interpreter responds to COMEBACK in about a millisecond. If data for the user who has just presented the new line are in core, the monitor reenters the interpreter with the proper user context. If not in core, the monitor initiates the necessary drum swap to bring in the user's data. Between 40 and 100 ms are available for the use of the interpreter to decide on the switch-to-green response without increasing the delay already imposed by the mechanical typewriter actions and the transmission of signals between computer and console. Typically, the interpreter's time is less than 3 ms. Crucial among the signals for continuing action is the receipt at the computer of the ready signal from the console, indicating that the carrier has indeed returned to the left margin. Receipt of the ready signal activates the transmission of the switch-to-green signal, which when received at the console causes the green light to turn on and the keyboard to unlock. The entire cycle is completed in one-third of a second, a limit set by the mechanical action of the typewriter and the signaling time between console and computer.

The three status requests required and the corresponding status responses occur because the speed of the operation

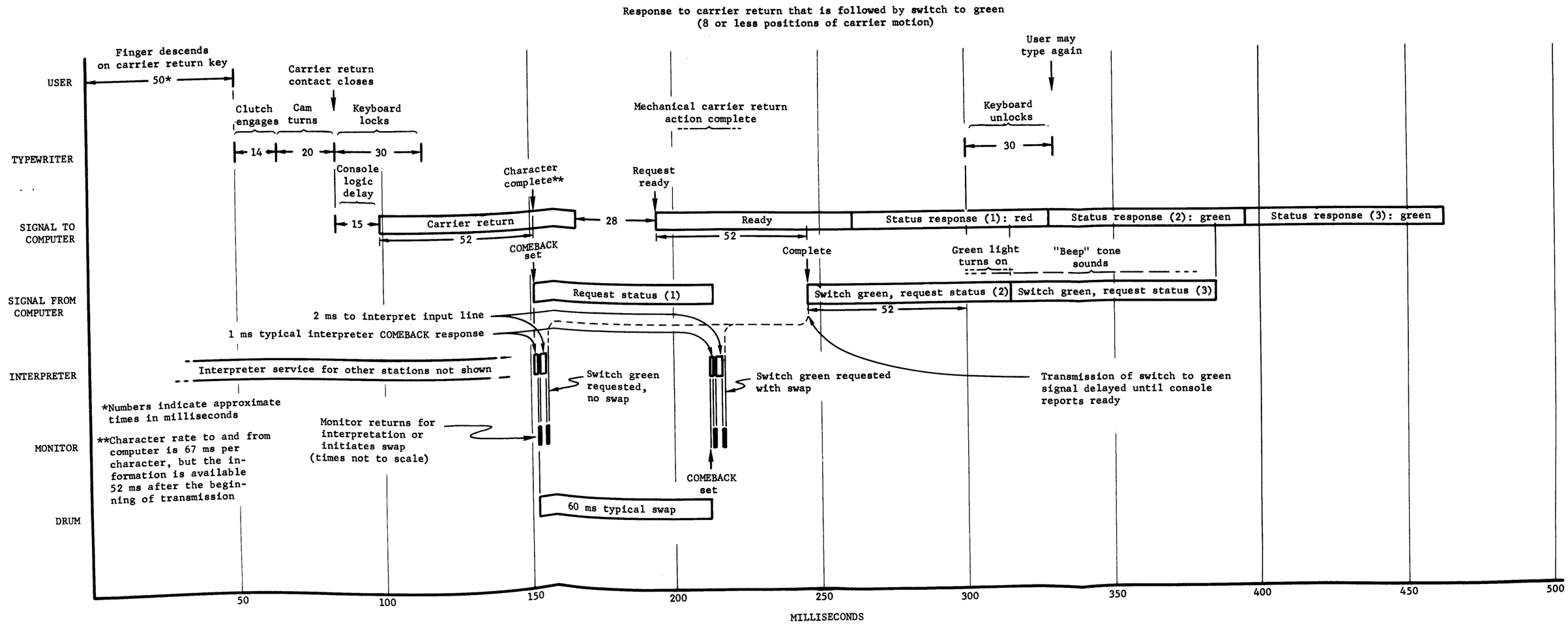

Fig.8—JOSS response to user request I

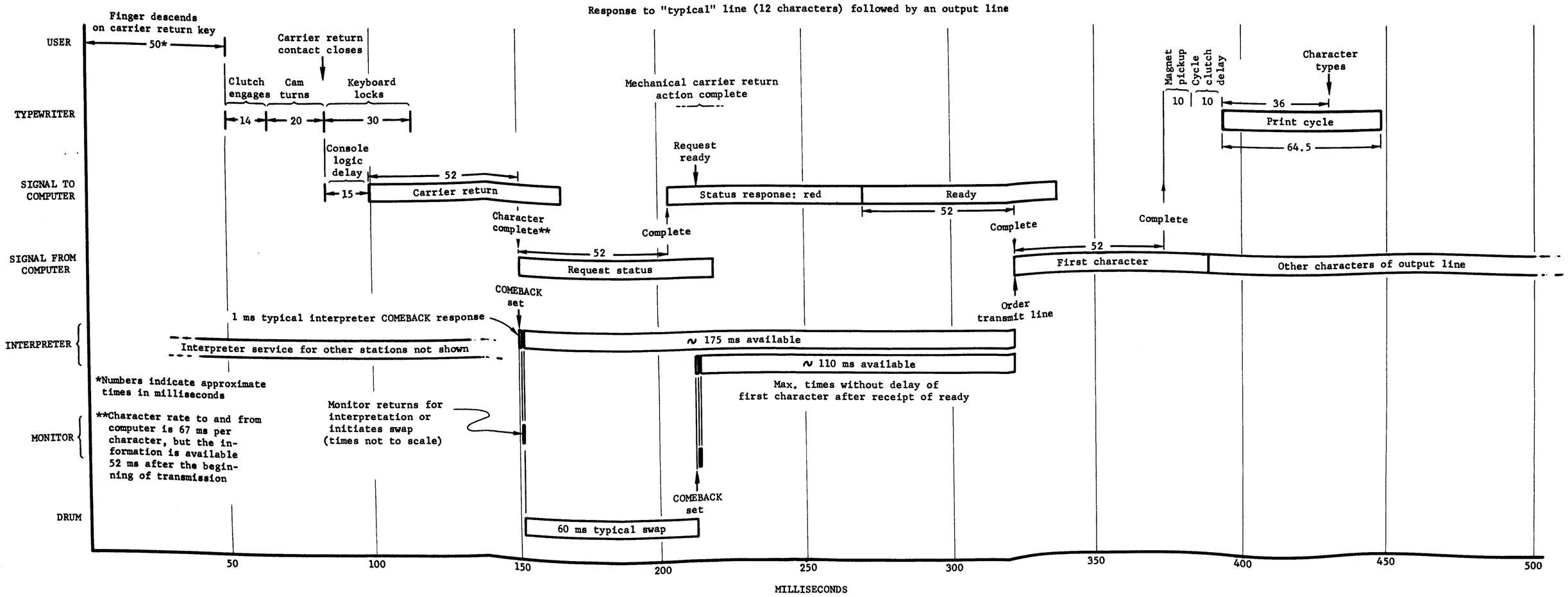

Fig.9—JOSS response to user request II

and timing of the signals cause the actual state of the console and the correct state as perceived by the computer to be temporarily out of sync. The orders to switch to green and respond with status continue until synchronization of state is achieved.

Figure 9 depicts a typical situation in which the proper response is a typed result, for example, to the input line "Type x/(1+x)." Again, the minimum time between pushing the carrier-return key and the printing of the first character of the typed response is determined by the mechanical action of the typewriter and the signaling time between computer and console. Between 110 and 175 ms of computer time can be used in preparing the response without increasing the delay. The delay of about 150 ms between mechanical completion of the carrier return and the first typed character of response is detectable to the careful observer.

To provide fast response, it is easy to see the importance of setting the logical interrupt COMEBACK both on the receipt of the carrier return and on the completion of the drum swap. If the actions were taken in regular round-robin fashion with several computing users each taking 200-ms quanta, the delays would be large, noticeable, and annoying.

Simultaneous occurrences of carrier-return signals from several consoles are stacked and serviced in order of arrival. This can delay response from a few milliseconds to a drum swap's worth but the probability of occurrence is extremely small, as can be seen from the average rate of such signals of 2 per user per minute.

IV. USER INITIALIZATION: LOGGING ON AND OFF

The two processors that handle user log-on and log-off are similar to the interpreter in the way they communicate with the monitor.

The log-on processor is called in response to a power-on signal from the console. This signal is passed to the monitor by the distributor and causes a state change to ON, an implied request for the log-on processor. The log-off processor is called in response to an OFF signal from the console. The OFF signal is delayed if a disc action is in progress to ensure completion of disc processing.

LOG-ON PROCESSOR

Once in control, the log-on processor is indistinguishable from the interpreter or any other processor: It requests buffers, it asks for transmission to the user, and it calls for switching of station control to the user.

Figure 10 outlines the flow of the log-on processor. Because this processor, like the interpreter, must be prepared to handle simultaneous log-ons for many stations, a sequence counter is stored in each user's block to control proper sequencing at each new entry. This sequence number is referred to on the flow chart as SEQ. At each entry the log-on processor picks SEQ up from the user block and enters a point of code, which corresponds to the flow block numbered with that sequence. On the flow chart, returns to the monitor are indicated by the flow break mark. Where not otherwise indicated, those returns request the transmission of a line and change of console control to the user.

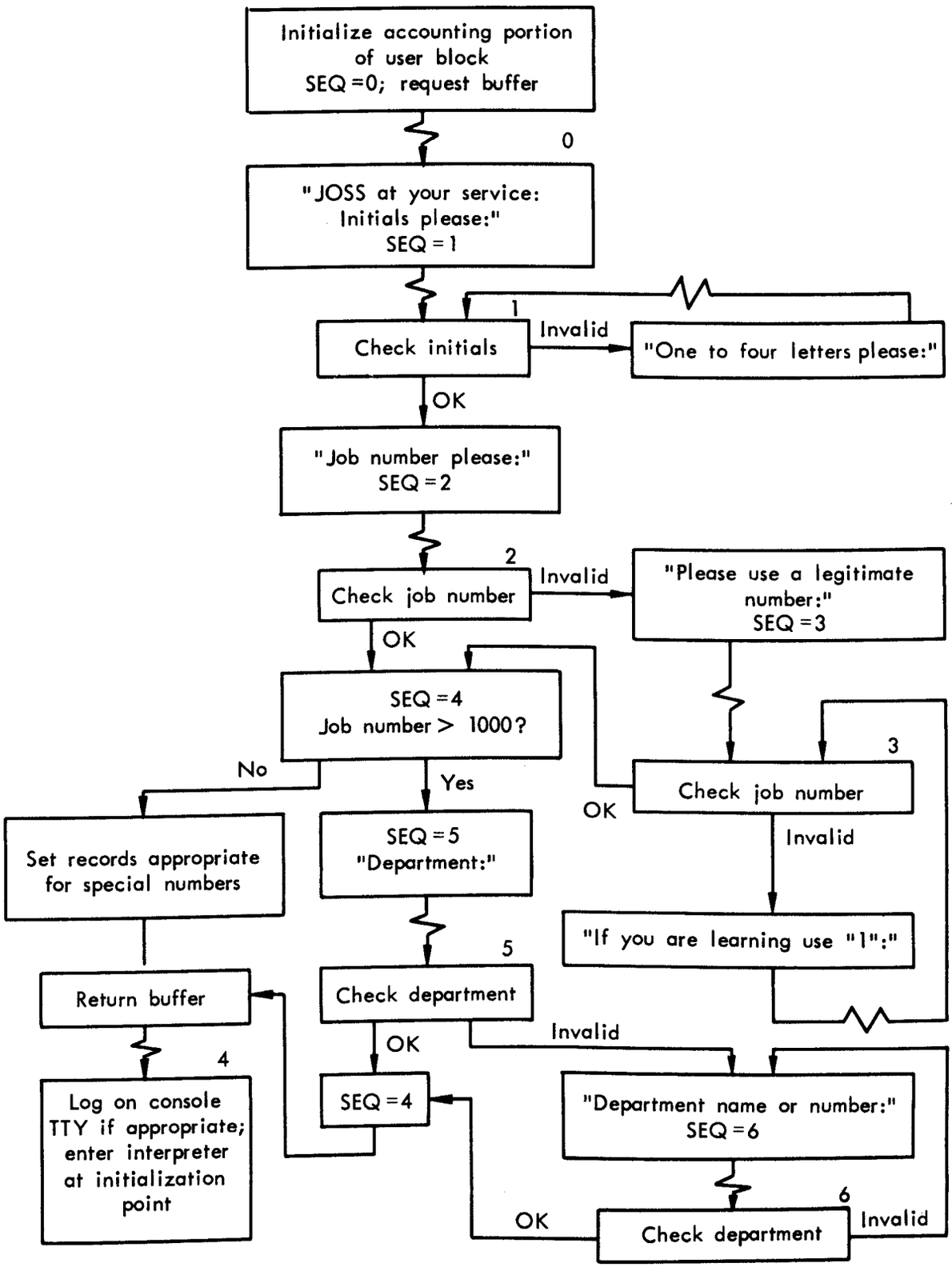

Fig. 10—Sequencing of the log-on processor

At sequence zero, JOSS salutation and request for initials are sent to the station. When the response is returned, a simple scanner analyzes the input string for a proper set of initials. No context about the input string is developed. Letters are counted, and periods, blanks, and hyphens are ignored. Numbers and special characters abort the scan and return an error message. The cycle at sequence 1 is continued until valid initials are typed.

Next, the job number is requested. Again, the scan is a simple counting of digits received, ignoring periods and blanks, and aborting if other characters are typed. If four or fewer digits are received, the number is subjected to a range check. This check eliminates over 90 percent of the numbers that are not valid project numbers. Although this validity check is not exhaustive, it is good enough to reduce usage of invalid numbers to an acceptable minimum.

Use of invalid numbers results in a two-level error message. The second and all subsequent invalid project numbers result in the same suggestive message at sequence 3.

Acceptable project numbers less than 1000 do not require a department name. Special department numbers are assigned automatically for accounting purposes depending on project number range for the various remote locations. Project numbers 1 to 4 are reserved, respectively, for users learning JOSS, demonstrating JOSS, debugging JOSS, and checking JOSS consoles.

The check for valid department at sequence 6 examines the first two digits for valid department number or the first three letters for the beginning of a valid department name or abbreviation. Invalid responses result in the re-request loop at sequence 6.

Finally, having satisfied all necessary, log-on information, the buffer is returned to the monitor, and the interpreter is entered at its initialization point from sequence 4. No return from the interpreter is expected or possible.

LOG-OFF PROCESSOR

The log-off processor is entered whenever a user is placed in TOF state by an OFF signal. It requests a buffer from the monitor, gathers appropriate accounting information from the user's core block, and attaches the buffer full of accounting information to the tape I/O queue. Final exit to the monitor is to the point that releases the user's core block.

V. SYSTEM OPERATION

In this section we shall discuss portions of the monitor that control system initialization, graceful shutdown, and several procedures that are convenient for operational and debugging purposes.

INITIALIZATION

For system initialization, the monitor requests date and time from the console TTY and then initializes all of the volatile memory records of the system including I/O headers, user states and status, and user buffers. Appropriate I/O channels and interrupts are enabled, and through a distributor routine all JOSS consoles currently with power on are sent signals to turn on their JOSS system light. ON signals are also sent to the monitor for all of these stations. The monitor main processing loop is then entered and regular JOSS services begin.

SHUTDOWN

The JOSS system shutdown procedure is a two-stage process controlled by the setting of two bits in the system switch register (see App. A for a complete description of the switch register). The first bit enables a gentle shutdown in which the following actions take place:

- o The queue-for-service message is replaced with a message indicating that the system is shutting down.
- o New active users are disallowed by setting the system parameter N.SON to zero. Because new active users are allowed only if the current number of active users

is less than the contents of N.SON, stations turning on will be placed in QM state and sent the now replaced queue-for-service message.

- o The administrative message of the system is changed to "Prepare for JOSS shutdown.", and the system switch that causes this message to be placed in all page heading lines is set.
- o All consoles are sent "beep" signals for the first five seconds of each minute.
- o When all users have finally turned off, summary statistics are printed on the console TTY, the message SYSTEM HAS SHUTDOWN is printed, and an end-of-file mark is written on the log tape and backed over.

The second stage of shutdown enables the operator to force users from the system by simulating OFF signals for each active console. Care is taken not to remove users currently reading or writing on the disc.

SPECIAL ENTRIES AND FUNCTIONS

Although JOSS is designed to operate continuously without manual intervention and does so almost completely, some controls can be set to vary outputs of the system. System initialization and shutdown have been discussed above, as have the control functions exercised by the modal parameters listed in Sec. III (pp. 24-25). It should be clear that the modals are primarily useful during nonstandard operation, particularly in system debugging.

As mentioned in the shutdown procedure, switches are used to control certain actions in such areas as statistical counts and console records. These switches are enabled by

setting bits in the memory cell SWITCH, which has been assigned memory location 100_8. The contents of this cell can be set during operation of JOSS through the use of the address and data switches on the console and the deposit key. Detailed system reaction to the setting of the switches is discussed in App. A, "JOSS Operating Instructions." In brief, the setting of switches enables the system to print the user's initials at the console, to log stations going on and off, to print a statistical summary, to disable the use of magnetic tape, and to send the administrative message to users.

This last, the administrative message, is input to the system through the keyboard of the console TTY (the only function for which the console TTY keyboard is used).

Fixed entry routines use cells 101_8 through 105_8. They function as follows:

101 Entry to a routine that produces a printable image or dump of core on magnetic tape. This routine is useful, of course, for system debugging.

102 Pseudo-signal cell. The contents are treated by the monitor as if they were signals from the distributor.

103 Recovery entrance. Reinitializes I/O and enters the monitor main processing loop. The failure situation that this routine will actually recover in unusual.

104 Routine to read a fresh copy of the JOSS system from the drum.

105 Routine to write the contents of core on the drum. This and the above "read" routine can act as residences for an absolute copy of the JOSS system. A single fixed drum location is used.

In addition, the debugging program DDT is customarily loaded with the system. It is entered at location 140_8. Dictionaries for DDT reside in upper core, which is overlain by user programs during JOSS operation. Therefore, use of DDT following JOSS is limited to octal storage references and standard conversion routines; that is, no symbolic translations can be made.

Detailed information regarding the use of these special routines is included in App. A.

VI. CONCLUDING REMARKS

The designer of an operating system for computers faces two kinds of problems: on the one hand is the desire to use a high-speed computer efficiently (unfortunately, efficiency is usually measured by calculating the percent of time the computer is doing something, without regard to the usefulness of that effort); on the other hand is the desire to provide rapid response to users' requests. It is with this second goal that we have been largely concerned throughout the design of the JOSS system and, indeed, the structure and operation of the monitor are aimed at providing a very fast response at a detailed level. Efficiency has not been sought for its own sake, but, of course, cannot be ignored, especially as it affects the prime goal of response. In fact, JOSS uses its machine today at about the same efficiency level as batch-operated machines (about 60 percent), with the marked difference that the remaining unused capacity is available instantly to whoever requests it. The machine overhead imposed by the monitor for scheduling and resource allocations is about one-half percent of real time.

In JOSS we are concerned with a specific kind of response--the computer response to typewriter requests. We begin a response cycle when the user's carrier-return signal is received at the machine and end the cycle when the requested result is prepared. In cases where a typed result is part of the request, the good response-poor response measure is clear: If the carrier rests at the left margin for a noticeable period of time before beginning to type the result, the response is poor; otherwise it is good. Of course, the amount of computing requested will limit response. The timing dia-

grams in Sec. III (Figs. 8 and 9) show that typical monitor activity, including drum swaps and the response time of the software, allows about 100 ms of computing before a noticeable delay occurs--enough for 60 to 200 JOSS arithmetic operations. Other modes, such as lack of required drum swap or coincidences of requests, can cause this time to be increased or decreased.

As in all computing systems, and in time-sharing systems in particular, the problem is to bring together into high-speed storage all the relevant data and processing programs that are needed to satisfy the request. To provide both response time and efficiency of operation, the data and program brought together must be all that is needed for a computation time of up to 200 ms. Furthermore, the contents of high-speed memory at any instant must be sufficient to support computation for periods of time that are very long compared to a drum swap time.

The central design features of JOSS that contribute to its ability to provide fast response are:

1. Resident software. All operating software is permanently resident in core memory. Virtually every portion of the JOSS code is used at a rate comparable to or higher than the rate of drum swaps. Serious response time degradation would result if any of the operating software had to be swapped.

2. Limited reentrant design of software. The operating programs must operate on more than one JOSS program "simultaneously." Although pure procedural design is not impossible, it generally requires more code to ensure that all context is in the user data area at every instant. The logical trap scheme of JOSS allows the interpreter routines time to reach

a convenient stopping place--a point of minimum context--before a switch of user occurs. In general, this inserts only a millisecond or two of delay--an insignificant amount of time compared to console signaling and drum swap times for the programming convenience obtained.

3. Relocation hardware. The hardware relocation scheme is such that no modification of user programs or data is required as data are moved from place to place in core. References to user data are hardware detected, and the current value of the relocation register is applied to give the correct memory reference.

4. Multi-port memory. Separate entries to the memories are provided for the drum I/O processor and the arithmetic processor so that swaps between drum and core may occur concurrently with computing.

5. Limited size for users. By limiting the total core available to an individual user, we ensure service to a computing user, while swaps into other core areas, coupled with short computing breaks, provide fast response to the interactive user.

6. Dynamic storage allocation. Allocation of user storage is provided on two levels: Thousand-word blocks of core memory are supplied to the interpreter by the monitor up to the system limit, and within that area the user data is organized in a linked-list fashion. Thus storage is supplied only as it is required by the computation, the scheme taking advantage of the great preponderance of small-size programs to achieve a high ratio of core size to average user size.

7. I/O controlled priority. The priority structure of JOSS first gives attention to signals from the consoles--

particularly carrier return. This provides a minimum of delay in response to simple requests without noticeably affecting compute-bound problems.

These combined techniques provide response with undetectable delay in the vast majority of cases at the present operating level of 30 consoles. We believe that such rapid response can be maintained to a level of more than 100 active consoles.

Appendix A

JOSS OPERATING INSTRUCTIONS

In normal operation JOSS needs no manual intervention, barring machine failure or the appearance of program bugs. However, JOSS does have both a loading and startup and a shutdown procedure.

Copies of JOSS are kept on the drum and on Dectape. The drum copy may be loaded only by routines contained in JOSS itself, and so it is not useful if other programs have replaced JOSS in the machine. The Dectape copy is loaded via a loader called 32K DECDUMP, which in turn is loaded by the RIM loader. DECDUMP is usually destroyed by JOSS operation and by many maintenance programs. The RIM loader, which begins in location 20_8, is rarely clobbered by anything. The discussion below outlines a detailed loading procedure.

LOADING

If 32K DECDUMP is in core, hit START with the ADDRESS switches at location 77600_8 and type "u,fLG" on the console TTY, where u is the Dectape unit number (8=0) and f is the file number (0 or 1) of the latest version of JOSS as determined from the tape label and the Hebraic maintenance log. The tape should spin and the G should finally print on the TTY. If you are successful so far, go on to the initialization paragraph. If not, you probably need to load DECDUMP.

Load the paper tape marked 32K DECDUMP in the reader with the full punched column toward you. Be sure the tape is well back and square in its guide slots. Hit STOP,

I/Ø-RESET, and TAPE-READER-ON. Put 20_8 in the ADDRESS switches and hit START. If the tape reads in, you may proceed from the middle of sentence 1 of the first paragraph under loading. Otherwise, you need to key in the loader.

Using the DEPOSIT key, the ADDRESS switches, and the DATA switches load locations 20_8 to 27_8 as in the table below; then proceed as directed in the previous paragraph.

ADDRESS	DATA		
20	71060	00000	60
21	71074	00000	10
22	25400	00000	21
23	71044	00000	26
24	71074	00000	10
25	25400	00000	24
26	-----	-----	--
27	25400	00000	21

If this doesn't work, and you have power on, etc., you need help beyond the scope of this discourse.

INITIALIZATION

At the end of a normal load, control will be in DDT where parameters may be set or patches may be made. See the section on system cells (pp. 24-25) for the effect of parameter changes at initialization time on subsequent operations.

JOSS is started by typing the sequence BEGIN$G.

JOSS will type its current date and time (as assembled or last-patched) and request a new date and time at the

console TTY. These should be typed in the same form that JOSS has used for typeout. If illegal characters or ridiculous numbers are given, the request will be made again. At the end of the date time sequence, JOSS immediately begins operation.

DATA SWITCHES

JOSS operation is controlled by pseudo switches (bits) in cell 100_8. Given below is a list of the system reaction to the setting of bits in location 100_8.

Bit	System Action
1-17	None
18	Beep all consoles
19	Enable shutdown procedure
20	Force off users
21	Disable the discfile
22-29	None
30	Administrative message has priority
31	Print statistical summary if fp(min+2)/5=0
32	Log ON and OFF signals at the console
33	Disable mag tape
34	Print users' initials and station number at console each minute
35	Send administrative message to all JOSS consoles

SHUTDOWN PROCEDURE

To ensure the proper collection of accounting information and to avoid clobbering programs in certain critical states (particularly those related to disc transfers), an orderly shutdown procedure is provided through the setting of bits 19 and 20 of the switch register in location 100_8.

When bit 19 is set, the following actions occur:

1. New user log-ons are prevented.
2. The queue for service message is replaced with a "shutting down" message.
3. The message "Prepare for JOSS shutdown." is placed on all page heading lines.
4. All consoles are "beeped" for the first 5 seconds of each minute.
5. When all users have turned off, the accounting tape is end-filed, final statistics are printed on the on-line log, and the message SYSTEM HAS SHUTDOWN is printed on-line.

When bit 20 is set, OFF signals are generated for all users as they enter noncritical states. Disc actions are allowed to complete before the "off" is forced.

A proper shutdown procedure, therefore, is to set bit 19 of location 100_8 about 10 minutes prior to the desired shutdown time, to give users warning and time to clean up. If all users have not departed voluntarily, bit 20 will force them off when actual shutdown is desired.

ADMINISTRATIVE MESSAGE

A message may be broadcast to all stations on their page heading lines by typing the message in at the console TTY.

The message must be less than 30 characters in length and terminated by a carrier return. It is not broadcast unless pseudo-switch bit 35 is set.

Two other messages may be generated by the system and may replace the administrative message. In priority order, they are (1) the shutdown message produced during the shutdown procedure, and (2) a message indicating trouble with the discfile. The administrative message may be given top priority by setting pseudo-switch bit 30.

CONSOLE TTY CHARACTER SET

The table below gives the correspondences between the TTY keyboard and JOSS typewriter characters for special characters. Alphabetics, numerics, and special characters with similar graphics are the same on both machines except that lowercase alphabetics cannot be typed in on the TTY.

	TTY	JOSS	
	↑	_	Underscore
Shift L		≤	
	@	≥	
	!	|	Absolute value
	%	≠	
	&	·	Multiplication dot
Shift K	[	[	
Shift M	]	]	
Control N		Backspace	
	Bell	Beep	

SPECIAL SYSTEM FUNCTIONS

In the event of system trouble, some or all of the functions listed below may be useful.

Tape Dump. Stop, I/O Reset, and Start at location 101_8 give control to a routine that dumps the contents of memory, including the accumulators, on tape in BCD print image form. On the resulting printed output, each line gives the octal memory location and the octal conversions of eight memory cells.

Drum Read and Write. Octal locations 104 and 105 are the entries to the routines that read or write, respectively, 32K words on the drum. If the drum code is intact, the system may be recovered from the drum. The routines may, of course, be entered from DDT by 104$G or 105$G, or by RJD$G or WJD$G if the symbol table is intact. These routines return to DDT when the action is complete.

Console JOSS Lights. The JOSS system lights on users' consoles may be turned out by JSR SHUT$X typed to DDT. In the absence of the symbol table for DDT, as is usual after JOSS operation, the $X must be preceded by the octal equivalent of JSR SHUT.

Simulated Console Signals. The signals sent to the monitor by the distributor in response to the console functions of on, off, interrupt, carrier return, and output line termination may be simulated by the use of cell 102_8. A word containing bit 0 set, the station number in the right half, and the signal code number in the left half will be picked up by the monitor if deposited in cell 102. Cell 102 is set to zero when the signal is processed.

The signal is processed at the same time as other signals

are interpreted and only if other signals occur. Thus, it may be necessary to generate a signal from another console to ensure that the simulated signal is processed. The signal numbers are

TO	0
IN	1
ON	2
OFF	3
CR	4

Thus, to simulate an OFF signal from console 33, deposit 400 003 000 033 in location 102.

SYSTEM CELLS

Table A-1 presents a short list of system cells that may be interesting to examine during JOSS operation. Actual octal locations should be taken from the current dictionary. (See JOSS maintenance log, Hebraic version.) For certain types of limited operation for debugging and testing, it may be desirable to set some of the parameter cells at initialization time. Of particular interest in this regard are S.OK, N.C, N.CB, and N.SON.

HALTS

Tables A-2 and A-3 give a complete list of machine halts in the JOSS code. The halts divide naturally into two classes: (1) detection of machine errors through the priority interrupt system, and (2) logical consistency checks that detect anomalous data. The halts have signaled both machine and programming

Table A-1

SYSTEM CELLS

Symbol	Contents
T75	Distributor SCR's are at T75 + station #
SG.L	Number of distributor → monitor logical traps
CTYR	Entry to channel 6 interrupt routine (tape and TTY)
APRR	Entry to channel 7 interrupt routine (processor)
DRMR	Entry to channel 2 interrupt routine (drum)
PROP	Entry to UUO interpreter
CUI	Station number of user currently being interpreted; zero when in idle loop
YEAR, MONTH, DAY, HR,MIN,SEC	Current date time (integers)
N.SON	Limit on number of active users
S.OK	Ignore all signals from stations numbered higher than this
S.S	Monitor user status cell; individual users are at S.S + station #
CORE	Status cells for 16 users' core blocks
DT.BUF	Head of the tape buffer list
N.BUF	Number of available user buffers
N.C	Number of blocks of core to be used by JOSS for all users
N.CB	Number of core blocks allowed per user
TOF	First of a table of 19 headers of monitor's user state queues
CT48A	Display of number of active users assigned 1,2,3,4,5,6 core blocks; six bits for each; count from the left
SHUT	Entry to the JOSS console system light turn-off routine
CKF	If non-zero, all drum transfers are checksummed for verification of the transfer

Table A-2

MONITOR HALTS

(All halts with 7770XX in PC, where XX is the octal halt number. Location of halt +1 in MA.)

Symbol	Explanation
H1	Impossible user number for interpretation
H2	Memory parity error or PDL overflow
H3	Unknown UUO (undefined user op code)
H4	Clock overflow
H5	Unknown monitor entry from interpreter
H6	Unknown logical trap from distributor
H7	TO reported but no buffer
H10	Impossible logical interrupt from disc routines
H11	Zero buffer address
H12	Zero buffer address
H13	Attempt to return available buffer
H14	Drum inexplicably busy
H15	Impossible drum interrupt
H16	Missed data, NONEX MEM, or parity error during drum transfer
H17	Unknown interrupt on channel 2
H20	Impossible single-state combination
H21	Unknown entry from disc routines
H22	No buffers available during shutdown
H23	Interpreter has attempted to reduce user to < 1 block
H24	User state inconsistent with state queue
H25	Illegal I/O context
H26	No buffers available
H27	Nonexistent memory reference from noninterpreter code
H30	CAIN or ILDB instructions fail in idle loop
H31	CPA ILL OP--memory protect violation from noninterpreter code
H32	Drum fails to write after five tries

Table A-3

HALTS OUTSIDE THE MONITOR
(PC is zero for these halts)

DISTRIBUTOR	Explanation
C25.50+3	Signal list overflow--too many logical traps
$C27+17_{10}$	First character in output buffer is a terminal
DISC	
F1.4+4	Improper disc initialization

errors many times in the past, but it is impossible to assign the cause of any halt without detailed analysis of the contents of memory at the time of the failure; even then the results may be inconclusive.

The best and surest recovery from a halt is to reload the JOSS system and start over again.

LOW BUFFER ALARM

The monitor periodically checks the number of available buffers and writes an alarm message on the on-line console if the number falls below a certain threshold. Although occasional heavy-output demand may cause the alarm to go off, a typical place for buffers to pile up is waiting for the tape unit. The alarm message suggests that the tape unit be checked for ready condition.

GRONKS

A number of the system-detected errors are responded to by clearing the user's data from core--in effect, turning him

off. This is done only when the nature of the error guarantees that the trouble is confined to the user's data. The user is said to GRONK,† and a message is printed on the console log indicating the cause of the GRONK. The message consists of an octal word containing the GRONK type code, the station number followed by the word "GRONK!" and two more octal words containing additional data related to the GRONK type.

The interpreter has several exits to the GRONK routine, which it enters using a PUSHJ instruction. The type word contains the contents of the push register.

The four monitor-generated GRONKs are type-coded as in the table below.

Type	Cause	First Data Word	Second Data Word
71	Drum checksum failure	Correct checksum	Computed checksum
72	Nonexistent memory reference	--	User's initials
73	Monitor and user data block initials fail to agree	Data block initials	Monitor initials
74	Reference to user memory outside the user's bounds	--	User's initials
75	Execution of unknown UUO	Checksum	User's initials

CONSOLE LOG

The console TTY logs the contents of certain counters each minute. In order to pack as much as possible on one line,

---

†The word GRONK is derived from the Johnny Hart comic strip "BC," in which a brontosaur shatters earth and sky by uttering the word GRONK. When the machine or program error causes the entire system to fail, we call it a system GRONK.

the counters are first scaled by dividing by a predetermined factor and then unscaled, or scaled by additional factors of ten if the fit into the allotted columns is poor. The number of columns allotted to each counter is one more than the number of characters in the column heading, the extra character being on the left. Printed output is followed by an * if the standard scale factor was not used. It is followed by an ! if an additional scaling by 10 was required (! may be read as x100), or by ", #, $, %, &, ', (,), for scaling by 3, 4, 5, 6, 7, 8, 9, and 10 powers of 10. Meaning of the headers and standard scale factors are given in Table A-4.

The log content is divided into six broad groups: user status, computation activity, console I/O, errors, monitor performance, and bulk I/O rates. All are cumulative event counts for the preceding minute, except the first group and OD which are instantaneous counts.

Table A-4

JOSS CONSOLE LOG

	Log Heading	Scale Factor	Contents
User status	TM	----	Current minute
	UR	1	Active users
	GQ	1	Green users
	C	1	Computing users
	B	1	Output limited users
Computation Activity	COM	6	User compute time (1 count = 100 ms)
	STA	10	Statements interpreted
	A	100	Unused
Console I/Ø	I	1	Core compacts
	TL	1	Lines transmitted to users
	CR	1	Lines received from users
	CI	10	Input characters
	CO	100	Output characters
Errors	T	1	Tape errors
	K	1	Disc errors
	T	1	Last station receiving a parity error
	#	1	Console parity errors
	R	1	Last station sending a parity error
	#	1	Scanner parity errors
	D	1	Drum errors (cumulative) plus number of GRONKs
Monitor Performance	T%	100	% of time computing for users
	RP	2000	Idle loop count
	U	1	Drum unoverlapped with compute
	OD	1	Number of users on drum
Bulk I/Ø	SW	1	Drum transfers
	S	1	Disc saves
	L	1	Disc loads
	D	1	Disc discards

Appendix B

JOSS HARDWARE DESCRIPTION

This appendix presents a short summary of the characteristics of the hardware comprising the JOSS system, together with the functional use of each equipment in the system. Equipment marked DEC was manufactured by the Digital Equipment Corporation, Maynard, Massachusetts. A detailed diagram of the components of the PDP-6 system is given in Fig. 11.

Arithmetic Processor (DEC 166)

A word-organized, binary, single-address machine with 16 accumulator/index registers (one of each can be referenced from every instruction), indirect addressing, and a versatile and complete instruction set including floating and fixed-point arithmetic, Boolean, bit, and variable-size byte operations. Also included is a 7-channel interrupt system with variable priority assignment. Memory protection and relocation registers have been modified in the RAND system to include an automatic data relocation mode similar to a second indexing. Instruction execution is asynchronous, with times typically around 4 μsec.

Fast Accumulator/Index Registers (DEC 162)

Sixteen flip-flop registers for the accumulator/index registers. Access time is approximately 400 nanoseconds.

Core Memory (DEC 163)

A total of 32,768 words of 36 bits each in two 16,000-word independently accessible boxes. Independent ports to the memory are provided for the arithmetic processor and the drum I/O processor. Full memory cycle is approximately

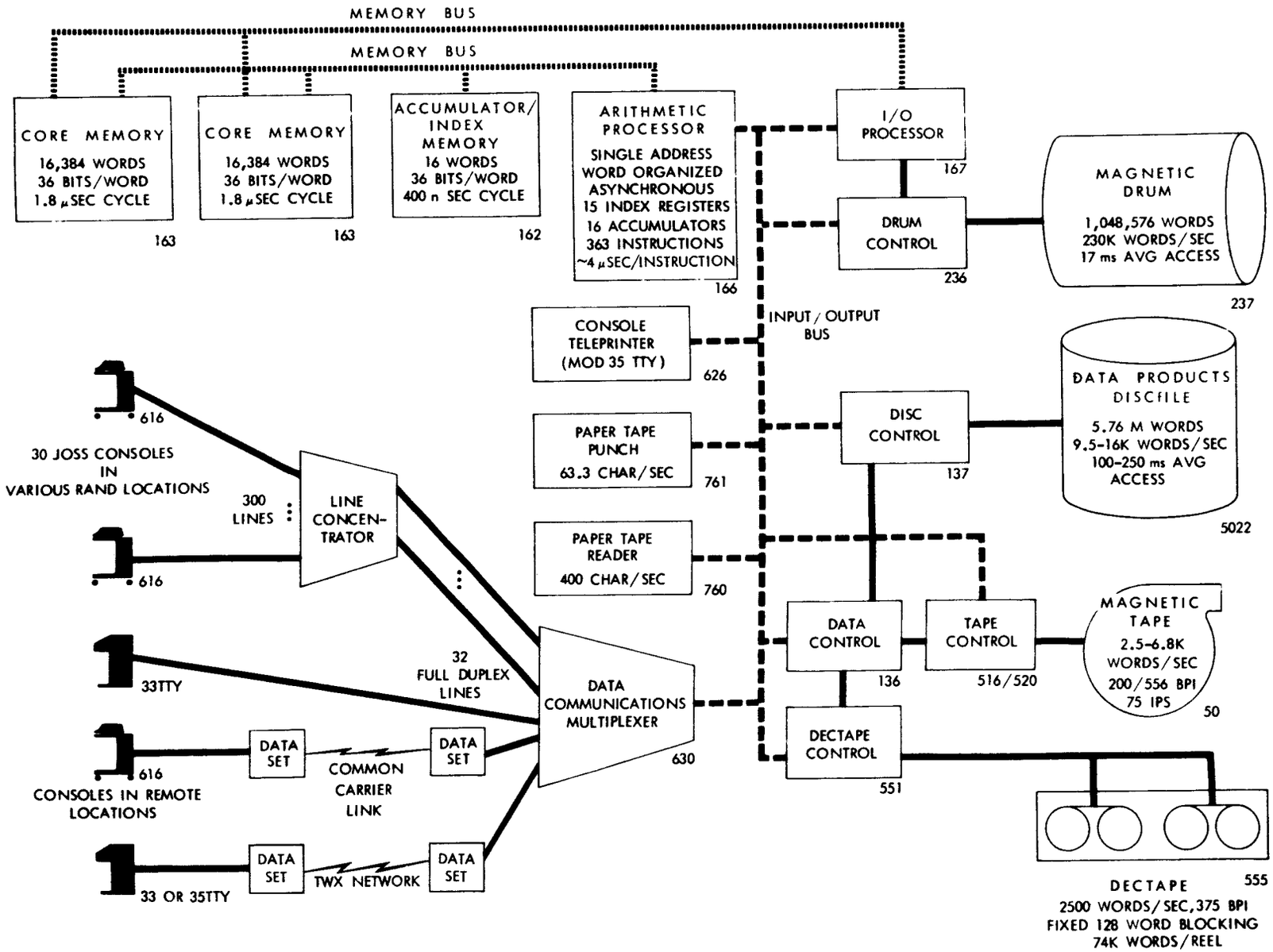

Fig. 11—JOSS PDP-6 system

1.8 μsec. A 37th-bit parity checks each word. The processor is interrupted on parity check failure. The JOSS system resides permanently in one box and users share the other box.

Model 35 TTY Console (DEC 626)

A teleprinter that acts as an auxiliary machine console. The printer and keyboard are completely independent devices. JOSS uses the printer to log usage statistics each minute and the keyboard to input administrative messages.

Drum I/O Channel (DEC 167)

A unit providing for the transfer of data between the drum and core memory. After word count and location initialization, the transfer is independent of the arithmetic processor and operates through memory ports separate from those used by the arithmetic processor.

Drum Control (DEC 237)

A device that provides logical control and addressing of the magnetic drum.

Magnetic Drum (DEC 237)

A drum supplying storage for JOSS user data when the total storage for all active users exceeds the size of allotted core memory. The drum can store 1,048,576 36-bit words. Maximum transfer rate between the drum and core is 4 μsec per word. In a single revolution of 35.2 ms, 8192 words may be transferred.

Data Control (DEC 136)

A device that controls transfer of data for the discfile, the tape unit, and Dectapes, converting words to bytes

as appropriate for each device. It may be used for only one such device at a time and, operating over the I/O bus, interrupts the arithmetic processor for each data word transferred.

Disc Control (DEC 270)

Logical control and addressing device for the discfile. More than one file can be handled by this unit.

Disc (Data Products DISCFILE 5022)

A file holding 5.7 million 36-bit words that may be transferred to core at a rate of 61 or 105 μsec per word depending on disc zone. Revolution time is 52 ms and positioning of one of sixteen independent read/write arms takes between 80 and 225 ms. JOSS uses the disc for long-term storage of users' programs and data.

Tape Control (DEC 516-520)

Logical control and addressing for the tape unit. Up to 10 tape units may be controlled by this device.

Magnetic Tape (DEC 50)

A unit that records on ½-in. IBM-compatible magnetic tape. It reads and writes forward or backward at 15,000 cps. Tape speed is 75 ips; recording density is either 200 or 556 bpi. JOSS records accounting and statistical information on this unit. Dumps of the disc are taken on tape for backup purposes.

Dectape Control (DEC 551)

Control and addressing device for up to 16 Dectape drives.

Dectape Dual Transport (DEC 555)

Four Dectape drives are included in the JOSS system for use in system support. Operating binaries for the DEC-

supplied time-sharing system and its subprocessors (assembler, editor, file manipulator) used in JOSS development are contained on Dectapes, as well as symbolic, relocatable, and absolute copies of the JOSS software. The tape is contained on small reels, each having a capacity of 74,000 machine words arranged in fixed-address 128 word blocks. Start-stop time is long, but, because of the fixed addresses, any tape block may be written without disturbing adjacent blocks. Tape travels 80 ips, packs 375 bpi, and transfers a maximum of 2500 words per sec to core.

Paper Tape Reader (DEC 760)

A device to read standard 8-channel tape at 400 cps. This is a primary system input for loaders and maintenance programs.

Paper Tape Punch (DEC 761)

A machine that punches standard 8-channel paper tape at 63.3 cps. It is used only for copying of paper tape programs.

Data Communications Multiplexer (DEC 630)

A unit that provides input and output character buffers for each console line. It scans up to 64 JOSS and TTY lines, providing an interrupt to the arithmetic processor when a complete character is received or transmitted. JOSS stations and TTY's are handled in full duplex mode either directly over wire lines or through dataphones.

JOSS Console (DEC 616)

RAND-designed user console built around an IBM Selectric I/O writer with a RAND-designed paging mechanism. Console control logic and transmission equipment for either wire or dataphone lines are included.

Model 33 TTY (Teletype Corp.)

Two full duplex model 33 teleprinters to provide access to the time-sharing system that supports JOSS system development and also to act as JOSS consoles (with character set and speed limitations) during JOSS operation.

Line Concentrator (J. Robins Electronics)

A machine built around Strowger stepping switches that detects the connection of a JOSS console to one of up to 400 wire lines and provides a connection to one of the inputs to the data multiplexer (DEC 630).

Appendix C

JOSS STORAGE BREAKDOWN

In order to give the reader an indication of the amount of code required for the monitor, the following breakdown by function is given. The numbers are decimal and approximate.

System initialization and shutdown	200
JOSS user log-on and log-off	400
Scheduling and signal interpretation	1400
Drum, tape, console, processor interrupt routines	700
Accounting and statistics (storage and code)	700
Special service routines (e.g., tape dump)	100
I/O buffers	800
Total	4300

The total storage breakdown (decimal numbers) of the software components of the system is given below in the usual order of loading. Appearance of the debugging routine DDT is not required for JOSS operation, but is included because it is a decided convenience when trouble occurs.

Arithmetic	1,600
JOSS console I/O	1,200
Disc I/O	700
Monitor	4,300
Interpreter	6,700
DDT debugger	1,700
Total	16,200

BIBLIOGRAPHY

PUBLICATIONS OF HISTORICAL INTEREST

Baker, C. L., JOSS: Scenario of a Filmed Report, The RAND Corporation, RM-4162-PR, June 1964 (AD 602074).

"The JOSS System: Time-Sharing at RAND," Datamation, Vol. 10, No. 11, November 1964, pp. 32-36. (This article is based on RM-4162-PR above.)

Shaw, J. C., JOSS: A Designer's View of an Experimental On-Line Computing System, The RAND Corporation, P-2922, August 1964 (AD 603972); also published in AFIPS Conference Proceedings (1964 FJCC), Vol. 26, Spartan Books, Baltimore, Maryland, 1964, pp. 455-464.

-----, JOSS: Conversations with the Johnniac Open-Shop System, The RAND Corporation, P-3146, May 1965 (AD 615604).

-----, JOSS: Examples of the Use of an Experimental On-Line Computing Service, The RAND Corporation, P-3131, April 1965 (AD 614992).

-----, JOSS: Experience with an Experimental Computing Service for Users at Remote Typewriter Consoles, The RAND Corporation, P-3149, May 1965 (AD 615943).

PUBLICATIONS OF CURRENT INTEREST

Baker, C. L., JOSS: Introduction to a Helpful Assistant, The RAND Corporation, RM-5058-PR, July 1966 (AD 636993).

Bryan, G. E., JOSS: Introduction to the System Implementation, The RAND Corporation, P-3486, December 1966; also published by The Digital Equipment Computer Users Society, DECUS Proceedings, Fall 1966.

Greenwald, I. D., JOSS: Arithmetic and Function Evaluation Routines, The RAND Corporation, RM-5028-PR, September 1966.

-----, JOSS: Console Service Routines (The Distributor), The RAND Corporation, RM-5044-PR, September 1966.