INITIALIZATION AND RECOVERY
EXTENDED OPERATING SYSTEM
3A PROCESSOR

CONTENTS PAGE

CONTENTS PAGE

Figures

Tables

Printed in U.S.A.

CONTENTS PAGE

1. GENERAL

1.01 This section describes the initialization function as applied to both the 3A Central Control (3A CC) and to the Extended Operating System (EOS). Initialization is a means of providing the system with a known good operating configuration of memory by applying a 3-step initialization sequence of Central Control (CC), the EOS and associated tasks, and the application. An initialization is any hardware, software, or manually initiated execution of the Central Control (CC) Initialization Program (CINIT). The CINIT may be initiated by a signal or command from the Program Timer (PT), error register, a signal from the System Status Panel (SSP) (Fig. 1), or a command from a terminal, or a software request via the INITCC macro execution. An initialization signal or command indicates that the integrity of the system data base or programs is questionable, and action is initiated to recover the integrity of the system.

1.02 This section is reissued to include features available with the G2B issue of the generic and to include revised and expanded postmortem data. The postmortem dump is now formatted so that a portion of the dump is common to all 3A CC-based applications, and a portion is unique to the application. Due to extensive revisions, change arrows have not been used.

1.03 The following sections and documents contain descriptions and data related to the functions described in this section.

SECTION	TITLE
254-000-000	Numerical Index, Stored Program Control
254-300-100	3A Processor, Description, Common Systems
254-300-120	3A Central Control, Theory of Operation, 3A Processor

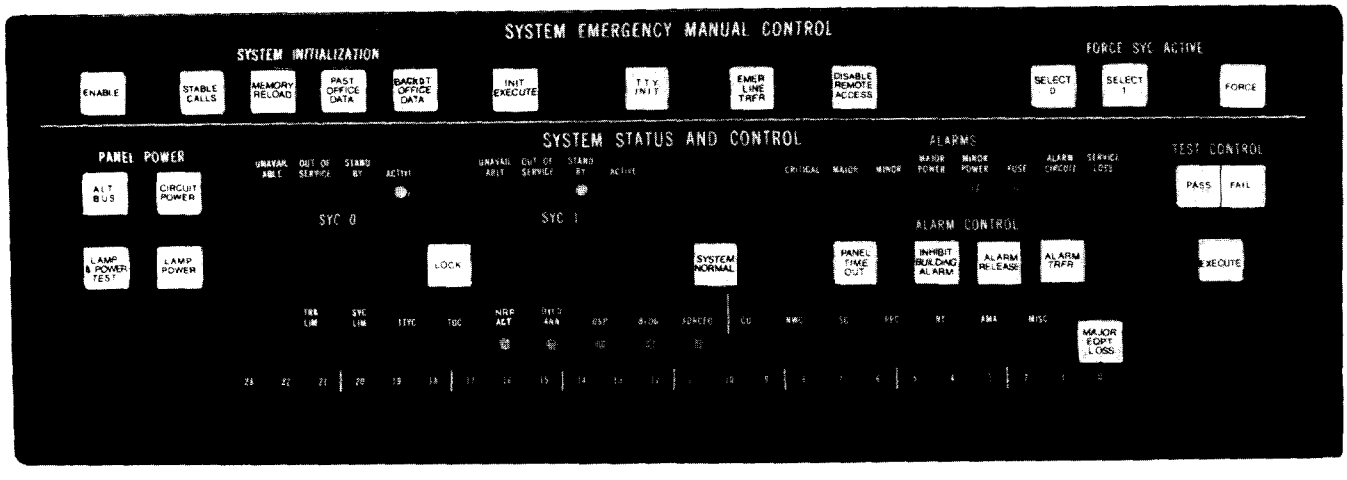

Fig. 1—System Status Panel (SSD)

254-300-180	System Status Panel, System Status Panel Controller, and System Status Panel Relay Unit, Description and Theory of Operation, 3A Processor
254-340-030	Processor/Process Management, Creation, Event And Communication Control, Software Subsystem Description, Extended Operating System, 3A Processor
254-340-080	Maintenance Overview, Software Subsystem Description, Extended Operating System, 3A Processor
254-340-082	System Utilities, Software Subsystem Description, Extended Operating System, 3A Processor
254-340-084	Resident Maintenance, Software Subsystem Description, Extended Operating System, 3A Processor
254-340-102	Basic and Extended 3A Processor Instruction Set, Software Subsystem Description, Extended Operating System, 3A Processor
254-340-104	Program Listing Organization and Usage, Software Subsystem Description, Extended Operating System, 3A Processor
254-340-106	Macros and Glossary, Software Subsystem Description, Extended Operating System, 3A Processor
IM-4C001-01	Input Message Manual for the Extended Operating System
OM-4C001-01	Output Message Manual for the Extended Operating System
TLM-1C900-01	Trouble Locating Manual, 3A Processor, Common Systems
TLM-4C702	Trouble Locating Manual, 3A Processor, Peripheral Channels (DMA)
TLM-4C704	Trouble Locating Manual, 3A Processor, CDI TTL Interface (CTI)
TLM-4C705	Trouble Locating Manual, 3A Processor, Parallel Channel (PCH)
TLM-4C706	Trouble Locating Manual, 3A Processor, Tape Data Cartridge (CTAPM)
TLM-4C707	Trouble Locating Manual, 3A Processor, Programmed Magnetic Tape System (PROMATS)
TLM-4C708	Trouble Locating Manual, 3A Processor, RS232-C Serial Interface
TLM-4C709	Trouble Locating Manual, 3A Processor, Teletypewriter Controller (DGNTC)

1.04 Table A provides a list of abbreviations and acronyms used in this section.

INITIALIZATION PROGRAMS

1.05 System initialization is accomplished by executing specific programs and subprograms. Initialization is performed under the control of the programs listed below:

(1) CC Initialization Program (CINIT) PR-4C618-01—provides primary control for any hardware, software, or manually initiated initialization, and initializes the CC.

(2) EOS Initialization Program (MAICCI) PR-4C605-01—provides a means for initializing the EOS and associated tasks.

(3) Initial Bootstrap Program Loader (CIPL) PR-4C616-01—provides a means of bootstrapping to load memory with programs and data so that processing may resume.

1.06 The 3A CC is initialized by any hardware, software, or manually initiated execution of CINIT. The stimulus for execution may be the failure of a software check which indicates that the integrity of the programming system and/or its data base is questionable. The stimulus for execution may also be a manual request for initialization. The CINIT initializes the CC and may consist of some or all of the following actions:

- Restoring the 3A CC memory to a known good state

- Restoring the periphery access to a known good state
- Aborting certain activities
- Zeroing or otherwise initializing temporary data memory locations
- Reloading resident generic programs and translations data from magnetic tape
- Saving previous system states for postmortem data and trouble analysis.

Not all of the above actions are performed for every initialization. The severity of the fault or problem encountered determines the actions performed. In general the system reaction becomes more drastic on successive recovery attempts. Progressively higher level recovery action is accompanied by more rigorous system reaction. Each successive recovery attempt raises the initialization level to the next higher level.

1.07 The EOS initialization program MAICCI initializes the EOS and its associated tasks. The MAICCI program performs the following actions:

- Initializes dynamic memory
- Performs kernel audit
- Sets up task descriptors
- Performs memory audit
- Switches Dual Bus System to on-line active 3A CC
- Initializes the input/output (I/O) channel
- Collects data for the postmortem dump.

Not all of the above actions are performed on every initialization. The value of the system initialization counter (INITLVL) determines the extent of the action taken during an EOS and application initialization. The values of INITLVL are summarized in System Initialization Table B.

RESULTS OF INITIALIZATION

1.08 A successful initialization consists of confirming that one or more of the following conditions exist:

- The 3A CC has a sane memory and is capable of performing normal system operations.
- The initialized 3A CC has a known good memory which was protected from outside interference during the initialization.
- Peripheral units selected to operate on-line are initialized.
- The duplicated or "standby" 3A CC designated to be placed on-line is capable of operating in a sane manner.

A successful initialization will provide a sane system capable of continuing normal processing and system operations.

2. CAUSES OF INITIALIZATION AND RECOVERY

2.01 If a hardware or software fault is detected, a system initialization will occur. A system initialization attempts to recover system integrity in the shortest possible time. The 3-step sequence of initialization involves initialization of (1) the central control (CC), (2) the EOS and associated tasks, and (3) the application. Briefly, the state of the system during an initialization is:

- Normal processing has been interrupted, hardware interrupts are blocked, and the EOS kernel is not executing.
- The updating of the off-line CC memory is disabled. If the initialization is manually requested via the system status panel (SSP), for a short interval of time both CCs are executing as independent entities. One CC or the other will be selected to become the active CC and will be placed on-line; the other CC will be disabled during the initialization.
- The severity of the action taken during an initialization is dependent on the value of the system initialization counter INITLVL. The value assumed by the counter is

dependent on inputs from the EOS, the application, and the SSP.

The action taken during a system initialization does not depend upon the source or the cause of the initialization, but upon the number of initializations that have occurred in the immediate past (within the initialization interval). A system initialization will establish a sane memory, but will not modify or change any application data base or send input/output messages to application peripheral devices. The causes of a system initialization are described in the following paragraphs.

A. Program Timer (PT)

2.02 The PT ascertains the time when the 3A CC begins executing a set of instructions and provides a maximum time (up to 1.6 seconds which may be set to a shorter period of time by the software) within which the execution must be completed. Upon completion the PT is reset (zero). If the PT times out before execution is complete, then the on-line or active 3A CC will stop and send a switch message via the maintenance channel (MCH) to the other (standby or inactive) 3A CC which will then initialize and go on-line. If the standby 3A CC PT times out, it will initialize again, restart, and check the status of the on-line 3A CC. If the on-line 3A CC has been manually forced on-line (locked on-line), a stop-and-switch does not occur and the on-line 3A CC will initialize.

B. Error Register

2.03 The detection of a hardware or a software fault within the CC sets an appropriate bit in the CC error register (ER). Three classes of faults or errors may be stored in the error register.

(1) Faults that cause a stop-and-switch and initialization of the other 3A CC.

(2) Faults that cause a hardware initialization of the 3A CC on which the fault was detected.

(3) Faults that cause an error interrupt (some of which may cause a stop-and-switch to occur).

The state of the error interrupt signals is stored in the error register as shown in Error Register Table C. Error signals that result in a stop-and-switch are represented by bits 0 through 9, and those that result in a hardware initialization are represented by bits 10 through 13. Bits 14 through PL and PH represent the error interrupts. The contents of the error register are outputted as part of the postmortem dump.

C. Software Initialization Request

2.04 If an application audit of memory detects a mutilation of some critical data base, a software request for a system initialization will be made via the INITCC macro. During normal system operation, control may be switched from one 3A CC to the other at intervals specified in the application programs.

D. Manual Initialization Request

2.05 Initialization may be initiated manually via the system status panel (SSP). A manual request is accomplished via the SYSTEM EMERGENCY MANUAL CONTROL - SYSTEM INITIALIZATION keys if automatic software initialization fails to restore the 3A CC to a proper operating condition. A manual initialization request would be necessary if it is determined that the system was functioning improperly and not recognizing fault conditions. Manual initialization must be accomplished in a specific manner and may vary somewhat for various applications; therefore, refer to appropriate documented procedures for each application for the proper manual initialization procedures.

2.06 The SSP SYSTEM INITIALIZATION keys (Fig. 1) are the ENABLE, STABLE CALLS, MEMORY RELOAD, PAST OFFICE DATA, BACKDT (backdate) OFFICE DATA, and INIT (initialization) EXECUTE. The functions of the keys are described as follows:

(1) Operation of the ENABLE - INIT EXECUTE keys will cause an initialization with INITLVL set to LCRIT, unless INITLVL was already greater than LCRIT. There will be no stop-and-switch of the 3A CCs unless there was a previous initialization within the initialization interval.

(2) Operation of the ENABLE - STABLE CALLS - INIT EXECUTE keys will cause an initialization INITLVL set to LMSG, unless INITLVL was already greater than LMSG. There will be no stop-and-switch of the 3A

CCs unless there was a previous initialization within the current initialization interval.

(3) Operation of the ENABLE - MEMORY RELOAD - INIT EXECUTE keys will cause an initialization with INITLVL set LBOOT unless INITLVL is already greater than LBOOT. This will cause a bootstrap of the system.

(4) Operation of the ENABLE - MEMORY RELOAD - BACKDT OFFICE DATA - INIT EXECUTE keys will cause an initialization with INITLVL set to LBOOT unless INITLVL was already greater than LBOOT. This will cause a bootstrap of the system plus the backdating of the translation data file from the backdate file.

The stop-and-switch function that occurs with certain initializations can be prevented by forcing the active 3A CC to be locked on-line. The 3A CC can be locked on-line by operating the appropriate FORCE SYC ACTIVE keys (SELECT 1 or SELECT 2) on the SSP.

3. INITIALIZATION OVERVIEW

3.01 ***General:*** A system initialization recovers (in a short period of time) the system integrity following the detection of a fault, and normal operation (processing) resumes. An initialization consists of initializing the CC hardware, the EOS and its associated tasks, and the application. An initialization can be as severe as a bootstrap or as simple as the initialization and termination of a single process. It is important that the action taken be matched to the severity of the fault to prevent unnecessary loss of service or availablity of the CC. If one initialization attempt is not successful, another initialization occurs immediately and the level of the initialization action is escalated. More severe recovery action is taken each time the level is escalated. An initialization may consist of one or more of the following actions:

- Aborting certain activities
- Restoring a control unit (CU) to a known good state
- Restoring peripheral devices to a known good state
- Zeroing or otherwise initializing temporary data locations in memory
- Reloading programs from the cartridge tape.

3.02 ***Initialization Levels:*** The initialization level is the value of the internal system initialization counter (INITLVL) whose value determines the severity of the action taken during a system initialization. The counter is set to zero at the end of the initialization interval (set by the application for a value between 0 and 1.6 seconds). Each time an initialization or recovery attempt fails during an initialization interval, the INITLVL is incremented and the initialization level is escalated. Each escalation results in more severe recovery action. There are three levels, LBOOT, LMSG, and LCRIT. The initialization level details are provided in System Initialization Table B. The values of INITLVL are set by the application and are summarized as follows:

- LCRIT - If the value of INITLVL is less than LCRIT, only the task in control is initialized plus any tasks specified by the application. If the value is equal to or greater than LCRIT, all tasks in the system are initialized.
- LMSG - If the value of INITLVL is equal to or greater than LMSG, all EOS dynamic memory and other data regions within EOS are completely initialized. This is the most severe EOS initialization short of a bootstrap initialization.
- LBOOT - The value of INITLVL is set to LBOOT if a bootstrap occurs and INITLVL is less than LBOOT. The EOS does not automatically cause a bootstrap. The application initiates a bootstrap via the INITCC macro.

The initialization levels are encoded in the Initialization Level-Time word as shown in Table D.

3.03 ***Bootstrapping:*** A bootstrap initialization results in the loading of selected blocks or all blocks of memory from the cartridge tape unit (partial or complete reloading respectively). A bootstrap will result if a system initialization occurs and both the initialization sanity check bits (ISC1 and ISC2) are set in the system Status Register (Table E). A bootstrap loads critical programs

and data from the cartridge tape and transfers control to the programs read in from the tape. Generally, the bootstrap initialization is as follows:

- A single block of program information is read into memory from the bootstrap file on the cartridge tape by a microcoded routine. Control is then transferred to the programs read in from the bootstrap file of the cartridge tape.

- The bootstrap program (CIPL) is read into memory along with associated programs and data responsible for initializing the central control (CC), and the CC is initialized.

- The generic, translations, and patch files are read into memory from the cartridge tape. Control is then transferred to the CC initialization program (CINIT).

3.04 The EOS does not cause a bootstrap. A bootstrap is the responsibility of the application via the INITCC macro. Certain hardware conditions, such as program timer (PT) time-outs, will automatically cause a bootstrap to occur, but this is a 3A CC hardware action rather than an EOS action. The bootstrap must be completed prior to calling the application routines.

3.05 The INITCC macro options are used to control the bootstrap. The bootstrap may (option YES) use the checksums to bootstrap, or may not (option ALL) use the checksums to bootstrap. If checksums are used when bootstrapping, only those blocks of memory are reloaded whose checksums indicate that the particular block of memory is mutilated, and the bootstrapping is completed in a shorter period of time. Otherwise all of memory is reloaded (ALL option specified).

A. 3A CC Initialization Program (CINIT)

3.06 The first step in a system initialization is the initialization of the 3A CC itself (Fig. 2). If a maintenance reset function (MRF) has just occurred in either or both CCs, both CCs may be attempting to initialize. At this time both CCs are isolated from each other and interrupts and MRFs are blocked. The following sequence of actions occurs:

(1) If a bootstrap initialization occurs, critical programs and data (bootstrap file) are loaded into memory from the cartridge tape. Most initializations are not bootstrap initializations and start at the next event (2 below).

(2) Critical registers, such as the general registers, are initialized.

(3) Postmortem data is collected for the CC that is initializing. This data may be useful later for analyzing the cause of the initialization. The postmortem dump format is shown in Table F.

(4) A decision is made as to which CC is to be placed on-line (in the active state). If one CC has been forced on-line by a signal from the SSP, the decision is simple and the forced CC goes on-line. Otherwise the decision is more complex and is based upon criteria encoded into the software.

(5) The CC chosen to be active disables the other CC and places it off-line.

(6) If a stop-and-switch occurred, postmortem data is collected from the newly off-line CC.

(7) If this is a bootstrap initialization, generic, translation, and patch files are loaded from the cartridge tape.

(8) If a stop-and-switch occurred, a test is made to determine if the newly on-line CC memory is current (in date). If it is not current, CINIT will attempt to update the memory from the now off-line CU.

(9) The value of the counter INITLVL is incremented. No action up to this point has been dependent upon the value of INITLVL.

(10) Control is passed to MAICCI for the EOS initialization.

B. EOS Initialization Program (MAICCI)

3.07 The MAICCI (Fig. 3) intitalizes the EOS and associated tasks. Both CCs have been initialized. One CC has been selected as the on-line CC and the other CC has been placed off-line and has no access to the peripheral devices. Hardware checks (also called MRFs) and error interrupts have

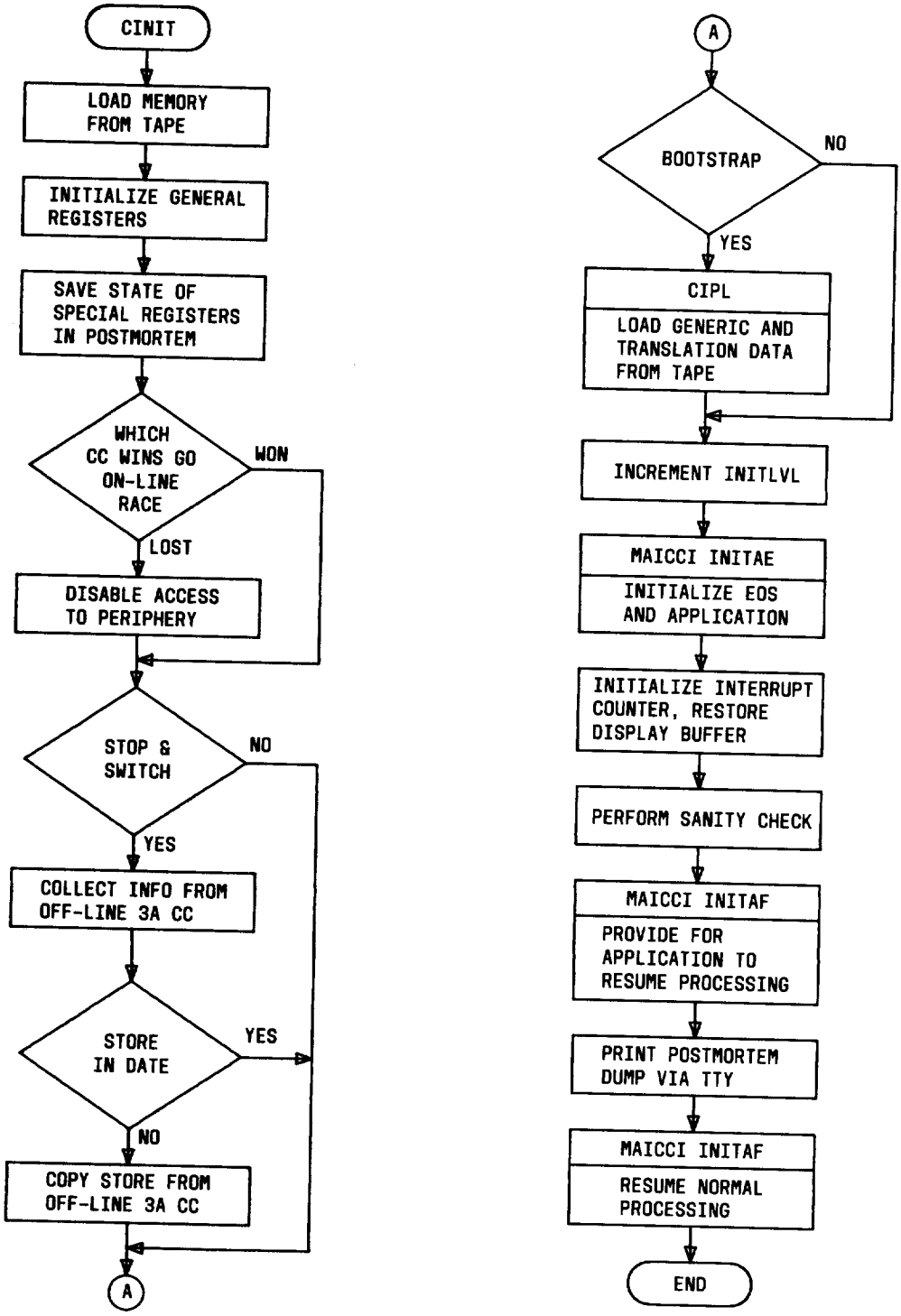

Fig. 2—3A CC Initialization Program (CINIT)—Flowchart

been enabled. The EOS and its associated tasks are initialized by the following sequence of actions:

(1) Set up initialization parameters in INITPARM in preparation of calling task initialization routines.

(2) Collect final postmortem data which consists of:

(a) Data pertaining to the EOS

(b) Testing to determine if this is the first initialization of a sequence of initializations

(c) Copying of the postmortem data by the off-line CC. A test is made to determine if this initialization was initiated via an application request for a bootstrap (if generic issue G2D is being used). If so, postmortem data has been saved in a memory scratch area. The MAICCI will now overwrite the current postmortem data with the contents of the scratch area, and this is denoted by writing the hex word CODE in the valid data indicator location in the postmortem dump (Table F).

(3) Perform a kernel audit if the value of INITLVL is greater than or equal to LCRIT or if INITLVL is less than or equal to LMSG.

(4) Initialize system dynamic memory if the value of INITLVL is equal to or greater than LMSG. Process descriptors and device control blocks are initialized from values specified in the prototype operating system tables (OSTABS).

(5) Reset the program timer to its maximum value of 1.6 seconds. Initialization of tasks that follow must be completed within the 1.6 seconds.

(6) Initialize input/output circuitry for which the EOS is responsible.

(7) Initialize the EOS and associated tasks as defined in OSTABS. This includes the following:

(a) Tasks are initialized in the order of entry of the task system names, not by any other priority. All file system input/output and all events in process are aborted, and all outstanding messages (queued but not retrieved) are flagged as old messages.

(b) If the value of INITLVL is equal to or greater than LCRIT, every process specified in OSTABS is initialized. If the value is less than LCRIT, only the process in control is initialized.

(c) If the pseudo initialization option (PSINT) was specified with the TASK macro for a particular task, the task (if initialized) will be restarted at the address and in the state specified in the TASK macro.

(d) Each task for which ready is the desired state is placed on the READY list. The READY list is built in the order of the task priorities.

(8) Initialize the input/output again because the OSTABS may have been changed by the application during the process of task initialization. Also, improper task initialization may have caused channel errors.

(9) Enable write protect capability according to the write protect tables in OSTABS as specified by the application.

(10) Clean up miscellaneous functions and transfer control to the dispatcher to restart EOS.

(11) Initialize the pseudo initialization routines specified by the PSINT=YES option of the TASK macro.

(12) Zero the value of the system initialization counter (INITLVL).

(13) The initialization complete message INIT RCOVRY COMPL is printed by CBLM at the end of the initialization interval.

C. Bootstrap Program (CIPL)

3.08 A bootstrap (Fig. 4) is the most drastic or severe initialization, resulting in a memory reload from the cartridge tape. The memory reload may be completely reloaded, or selected blocks of memory may be reloaded through the use of checksums. The use of checksums reloads selected

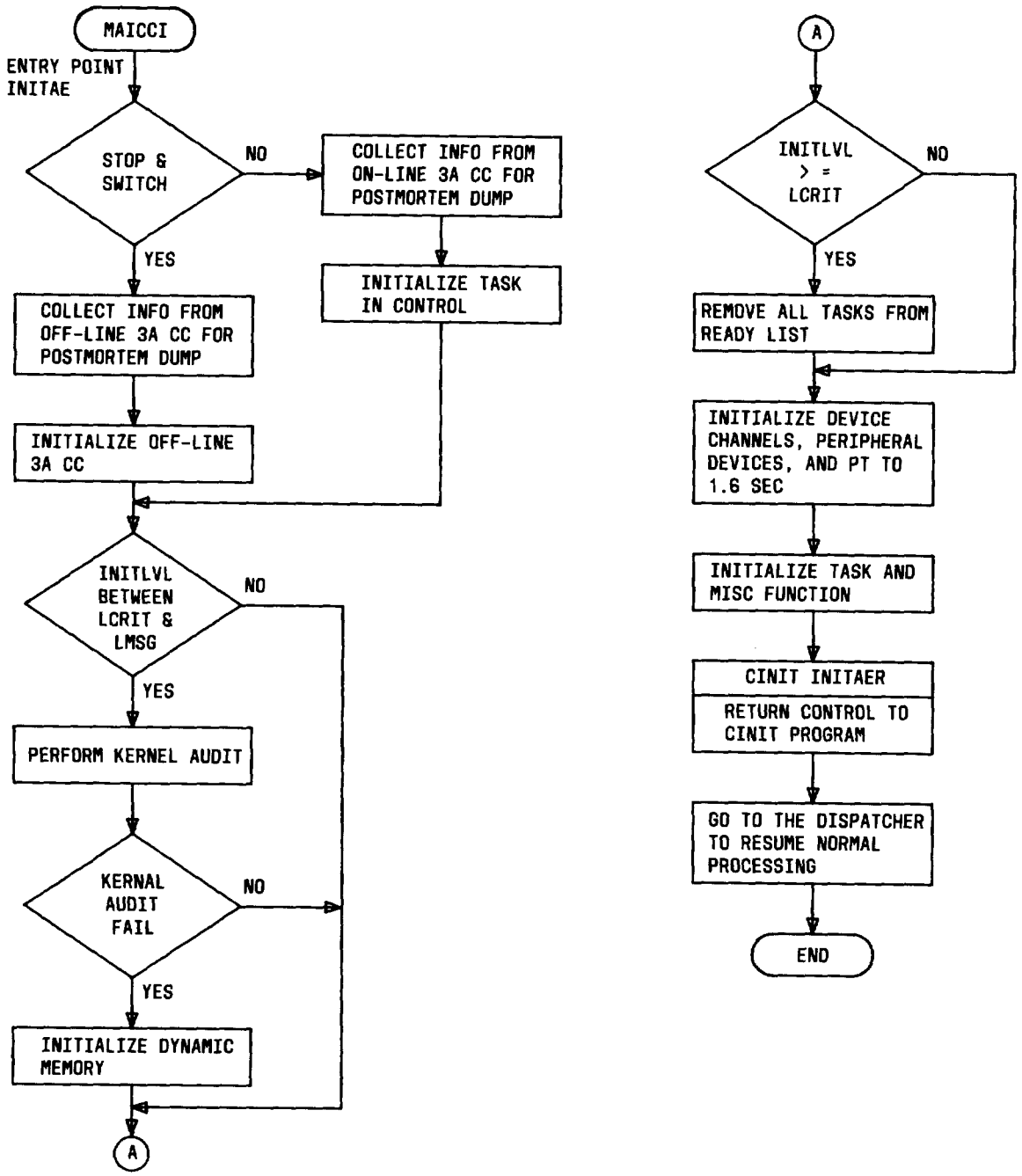

Fig. 3—EOS Initialization Program (MAICCI)—Flowchart

blocks of memory that have been determined as being mutilated. A memory reload using checksums is accomplished quicker than a complete memory reload and may be used when it is determined that severe mutilation memory has not occurred.

3.09 A complete memory reload may be initiated via the SYSTEM INITIALIZATION- MEMORY RELOAD key on the SSP, or via the software under certain conditions. Microcode passes control to the entry point START in CIPL and the bootstrap portion of the programs is loaded into memory from the cartridge tape. Control is then passed to entry point INITBOOT in CINIT. The CINIT performs some hardware initialization and testing and returns control to entry point BOOTLOAD in CIPL. This

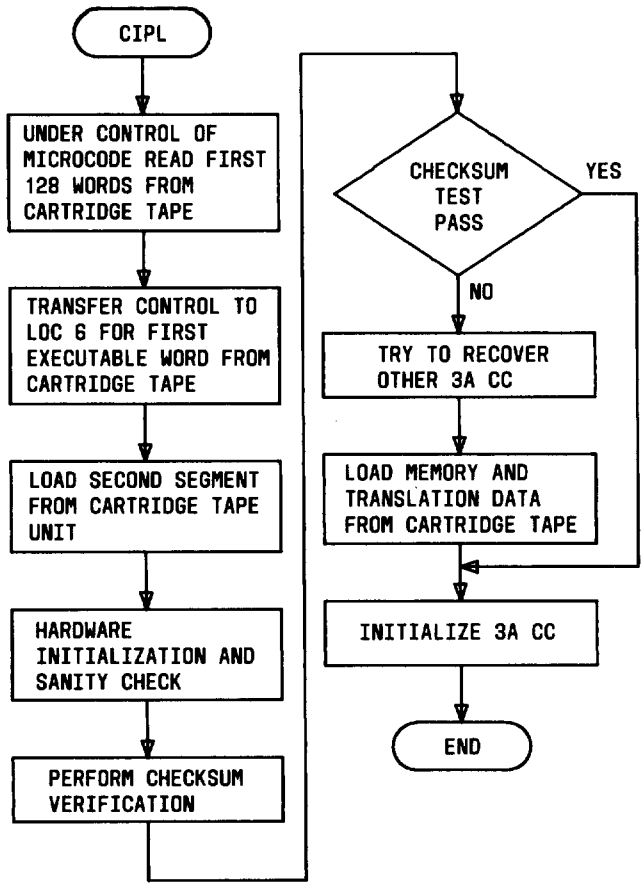

Fig. 4—Bootstrap Program (CIPL)—Flowchart

portion of CIPL generates checksums for program and data memory blocks and tests the checksums to determine which (if any) blocks of memory are mutilated, and reloads all blocks that are mutilated. However, if CINIT requests that all of memory be reloaded regardless of the checksum status, CIPL will ignore checksums and reload the complete memory.

4. INITIALIZATION AND RECOVERY DESCRIPTION

4.01 ***General:*** The need for a system initialization arises when a fault or error severe enough to prevent normal processing is detected, and system integrity is questionable. The system will attempt to recover by restoring the CU and peripheral equipment to a known good state. If the initializing CC cannot recover, then a switch to the other CC may be made to restore the system. An initialization may progress through several levels of initialization, with each level involving more severe actions. The CC Initialization Program (CINIT) is entered on all initializations and initializes the CC. The Initialization Program (MAICCI) is used to initialize the EOS and application tasks. The Bootstrap Program (CIPL) is used whenever it is necessary to reload memory from the cartridge tape. A memory reload may be either a complete reload of all memory or only a reload of selected blocks of memory that appear to be mutilated. A bootstrap may be initialized via the software or via a signal from the SSP (a manual request). A postmortem dump is printed on the terminal after an initialization and contains data useful in determining the cause of the initialization. The end of an initialization interval is signified by the output message INIT RCOVRY COMPL.

A. CC Initialization Program (CINIT)

4.02 The stimulus for execution of CINIT may be the failure of a hardware check or a software check, or a manual request for an initialization. The CINIT may perform a complete system initialization or may initialize only selected blocks of memory having mutilated data as determined by the checksum check. The main section of CINIT initializes the CC for a complete system initialization and the remaining sections contain routines or subroutines which are used primarily by the CC initialization section or by CC-oriented interrupts. The interrupt routines bear no direct relationship to the initialization routine although they do share some subroutines. The CINIT subroutines are described in Table G. After the CC is initialized, CINIT exits to MAICCI to perform some of the EOS and application initialization, and then control is returned to CINIT for completion of some clean-up tasks. Postmortem data is collected by CINIT for later printout as part of the postmortem dump. CINIT calls the CIPL for a bootstrap when a complete memory reload is necessary or when only selected blocks of memory require reloading (which is quicker).

4.03 Prior to entering CINIT certain registers are initialized by the hardware or microprogram. Other registers are used by CINIT execution and their contents are dependent upon the execution. Initialization of the critical registers established a known good state of the registers. It is also necessary to protect the initializing CC from interference sources from the other CC, ie, interference via the maintenance channel (MCH), interrupts, and the main store bus. These sources are blocked

by setting CC = 1 (disables fatal MCH commands such as a STOP), setting BIN = 1 (blocks all interrupts), and by setting ISO = 1 (isolates this main store from the other CC).

4.04 It is also necessary to protect the initializing CC from its internal self-check circuits which could detect bad latent data in the CC or an abnormal state of the CC. All check circuits are blocked. The isolation and blocking of certain activities is necessary because it is not known what state the system was in when the fault or error occurred that caused the initialization.

4.05 When an initialization occurs, the state (data from certain registers) of the CC is saved for the postmortem dump. If no switch (to the other CC) occurred, data is saved from the current CC. If a switch did occur, data is saved from the formerly on-line CC. The format of the postmortem dump is shown in Table F. Before the data is saved, however, it must have good parity. The parity of the saved data is checked, and if bad parity is detected it is corrected.

4.06 The general registers, the maintenance channel, and the special registers are initialized. Certain bits of the SS register (Table E) must be reset. The block timer check (BTC) bit is reset for use by the SANITY check, and the block hardware check (BHC) bit is reset during the SANITY check. The initialization SANITY check bits (ISC1 and ISC2) cannot be reset until the initialization level (INITLVL) count is incremented.

4.07 ***On-line/Off-line Decision:*** A decision is made as to which CC is to become active or on-line. This decision is made based upon certain priorities and rules as follows:

(1) If the CC reset circuit is active, place the CC off-line. The reset circuit is activated via the processor panel or via the power-up sequence. If the reset circuit is activated for the on-line CC, both CCs will go off-line and recovery will be made by a programmer timeout.

(2) If either the lock on-line (LON) or the lock off-line (LOF) bit is set, this bit is the basis of the decision. If the LON bit is set by a signal from the SSP, the 3A CC selected will be locked on-line. The LOF bit is set by a signal from the SSP LOCK key. When LOF is set, the active 3A CC is locked on-line, and the off-line 3A CC has its input/output channels disabled so that it cannot interfere with the on-line 3A CC.

(3) Checks are made to determine if any tests are being executed on the off-line CC, or if the on-line CC is trying to initialize the off-line CC to a known good state. (Diagnostics may be testing to determine the source of an initialization.) For either of these the CC will be held off-line. This check is performed only if INITLVL is zero.

(4) Checks are made for legitimate initialization switch sources internal to this CC. If an active and legitimate source is found, a switch to the on-line status is made for this CC. A legitimate source is either a programmer timeout or an initialization message via the MCH.

(5) If none of the above (1 through 4) apply, the on-line/off-line decision is based upon the old state of the CC flip-flop. If the CC was on-line prior to the initialization, its CC flip-flop is set and it will initialize and become the on-line CC. Faults detected in the on-line CC (which cause an initialization but no switch to the other CC) will cause the on-line CC to initialize itself. That there was no switch implies that this CC is the old on-line CC and it will initialize and remain on-line as the active CC.

(6) If none of the above (1 through 5) apply, the state of the other CC is interrogated after the decision to go off-line is made. If the other CC is stopped, the initialization will proceed with this CC and this CC will go on-line. Otherwise, under certain conditions (ISC1 bit set and an initialization request from the SSP) the system could become deadlocked with the formerly on-line CC stopped and the formerly off-line CC halted.

The decision to place the CC on-line has been made based upon the above priorities, but a SANITY check is performed on the CC prior to placing the CC on-line. The check determines if the CC is reasonably capable of executing some code. If

the CC fails the SANITY check, it is stopped and the other CC will have a chance to take control of the system. If the CC passed the SANITY check, the decision to place it on-line is final and the CC goes on-line and assumes control of the system. The 3A CC not selected to go on-line is disabled by the on-line CC by disabling its access to the peripheral units.

4.08 ***Memory Reload (Bootstrap) Request:*** When bootstrapping, differentiation is made between a manual and an auto (automatically via the INITCC macro) request for a reload. For a manual request, everything is copied from the cartridge tape or only a checksum reload may be requested as desired; whereas, for an auto request, only selected blocks of memory are reloaded. The auto request speeds up recovery by using checksums to determine which blocks of memory are mutilated and need to be reloaded and reloads only those blocks. Time is saved by not reloading the good blocks of memory. If the initialization was initiated from the SSP, a bootstrap will be performed. The ISC bits in the other CC SS register must be set to guarantee that both CCs will be bootstrapped. Bootstrapping is delayed until after one CC has been initialized and given control, ie, it is the on-line CC. This is necessary since the on-line CC must use the tape data unit normally associated with the other CC, and if the other CC is attempting to use the tape unit, interference is possible (and not desirable). The Initialization Control (INITCTL) word (Table H) is checked to determine if a checksum reload, a complete memory reload, or a backdate office data reload is required.

4.09 If there has been a stop-and-switch (the formerly off-line CC is now the on-line CC) and postmortem data must be collected and saved from the previously on-line CC. An attempt is made to retrieve data from the major registers. If an error is detected in the MCH while collecting the data, then the data probably is not reliable, and the Valid Data Indicator BOBO (Table I) is returned to flag the unreliable postmortem data. If no MCH errors are detected, the postmortem data collection is completed. The on-line CC store is marked as the up-to-date store and the off-line store is marked as out-of-date. Gathering of data from the CC has now been completed and it can be initialized and tested. The on-line CC is presumed to be good at this time and the hardware checks (self-checking) are turned on. The CINIT now exits to MAICCI (entry point INITAE) to perform some EOS and application initialization after which control is returned to CINIT (entry point INITAMR) to complete the initialization task.

4.10 ***Completion of Initialization:*** The system initialization has been completed and some clean-up tasks remain. The recovery output message RCOVRY CU INIT is initiated and printed out on the terminal to indicate that the system has recovered from an initialization due to an error. The contents of the error (ER) register and the initialization level number are also printed as part of the message. At the end of the initialization interval the message INIT RCOVRY COMPL is printed out to signify completion of the EOS recovery.

4.11 The initialization strategy is to initialize larger and larger portions of unprotected memory in order to recover (purge system of bad data) automatically. The protected portion of memory has not as yet been initialized and may contain bad data. The protected memory is audited by the use of checksums (from tape) and blocks of memory determined to be mutilated are rewritten. The CIPL is used to perform the checksum audit. Initiation of the execution of CIPL can be accomplished by setting both of the ISC bits which will cause a bootstrap and checksum audit. If the checksum audit succeeds in removing the bad data, the system recovers.

4.12 On other than the highest initialization level, ISC2 is cleared but not ISC1. The first initialization will not cause a switch to the standby CU, thus if the fault is such that the initialization can be successfully executed, the system does not cycle. Recovery will not occur if the ISC1 bit is zeroed during the initialization. Leaving ISC1 set will cause a switch and recovery upon detection of the next error. The ISC1 bit is cleared at the end of the initialization interval. The ISC bits in the off-line CC are always zeroed except when the initialization level recycles, in which case they are both set. This guarantees that the checksum audit will be executed regardless of whether or not the next initialization causes a switch. The ISC bits are zeroed so that intermediate initializations will not require a program timer timeout to recover.

4.13 ***Remove Other CC From Service:*** If a switch occurred, the now off-line CC experienced an error, and a decision must be made whether or not it should be automatically

removed from service (placed in OFF-LINE OOS/AUTO/FAULT state). The criteria used to make the decision is:

- CC fails SANITY check
- CC experiences excessive errors.

The first criteria determines if the CC is capable of executing some code but is not completely foolproof, hence the second criteria is used as a backup to the first to assure that the CC is not removed unnecessarily. Excessive errors are defined for this purpose as two initializations within a certain number of base level loops or approximately 30 minutes.

4.14 ***Error Interrupts:*** The 3A CC self-checking circuits may detect an error or fault which does not affect the execution of the software programs. Such errors include those affecting the other MAS, the input/output (I/O) channels, I/O bad parity, the MCH, and the other CC. The action taken is determined by the error interrupt that occurred. The possible errors and their corresponding actions are as follows:

(a) ***I/O Bad Parity:*** Purge the bad parity and record significant data for later analysis.

(b) ***Other Store Error:*** Test the other MAS and if the test fails, mark the other MAS as out-of-service. In addition, the off-line CU is removed from service (noted by a 3 in the variable field of the RMV CU message).

(c) ***Error from the Other CC:*** Test the other CC (SANITY check) and if it fails the test, flag the CC as out-of-service.

(d) ***I/O Channel Error:*** Stop and switch to the other CC.

B. EOS Initialization Program (MAICCI)

4.15 ***General:*** The MAICCI program initializes the EOS and associated tasks and is entered at entry point INITAE (Fig. 3) from CINIT. Both 3A CCs have been initialized, and one 3A CC has been selected as the active (on-line) CC, while the other has been selected as the standby CC, having no access to the periphery units. Hardware checks have been previously enabled, but interrupts are still blocked. The EOS is not cycling and active system macros cannot be used except for the INITCC macro. The EOS initialization proceeds in a step-by-step sequence with some activities depending on the particular value of INITLVL. For a major portion of the initialization, the EOS and application initialization is interleaved, since a number of EOS functions are initialized as tasks in exactly the same fashion as the application tasks. A brief description of the subroutines used by the MAICCI is given in Table J.

4.16 Various actions are taken dependent upon the value of INITLVL, (Table B) which is summarized below:

- If equal to or greater than LMSG, system dynamic memory is initialized.
- If equal to or greater than LMSG + 1, perform memory audit and correct any bad parity found.
- If equal to or greater than LCRIT, initialize all tasks (specified in OSTABS).
- If less than LCRIT, initialize the task in control.
- If value is between LMSG and LCRIT, perform a kernel audit.

4.17 The EOS initialization routine first sets up the initialization parameter vector (INITPARM). The INITPARM is used to call the task initialization routine and to pass critical parameters (information) concerning the initialization to the task being initialized. The entries in INITPARM will contain the following types of information:

- The value of INITLVL for this initialization
- The state of the SSP initialization keys if a manual request for initialization is made, otherwise key states will be zero
- The system name of the task in control when the initialization occurred (name equals zero if kernel was in control)
- The miscellaneous bits of the INITBITS word

- The restart address at which the task being called is to be restarted (address must be entered by the task and only if task is called prior to the system restart)

- The restart state in which the task being called is to be restarted (state must be entered by the task) after system is executing

- Interrupt bit K is set if the 3A CC interrupt K was active when the initialization occurred (otherwise interrupts will be zero signifying no active interrupts)

- A pointer (address) to a 16-word block of memory that contains the contents of the 16 general registers when the initialization occurred (if there was no switch the contents of registers R9, R10, and R11 are not meaningful).

EOS Initialization

4.18 The EOS initialization routine sets the program timer (PT) to a maximum interval (1.6 seconds) and all task initializations must be completed during this interval. All input/output circuitry for which the EOS is responsible is initialized. This includes the switching of all Duplex Bus Selectors (DBSs) in the direction of the on-line 3A CC. The application is permitted to modify the store which is normally write-protected.

4.19 The EOS and application tasks defined in the Operating System Prototype Table (OSTABS) are initialized.

(a) Tasks are initialized in the order of the system task name entry into the ready list rather than priority. Since EOS is not active at this point, priority has no meaning to the system. The EOS initialization performs the following functions regardless of the initialization level:

(1) All EOS file system input/output in progress is aborted, and all open files are closed and detached.

(2) All events in progress are aborted. The event mask is disabled (zeroed) and remains disabled until re-enabled by the proper task.

(3) All outstanding messages which are queued but not retrieved are flagged as old messages.

(b) When the value of INITLVL is less than LCRIT, the task in control at the time of the initialization is initialized, along with those which are selected to be initialized each time a maintenance reset function (MRF) occurs. When the level is greater than LCRIT, every task defined in OSTABS is initialized.

(c) When a pseudo-initialization routine is used for a task, then the task will be restarted at the restart address and in the restart state specified. When a pseudo-initialization is not specified, the task will be called by an event (0 or 1 depending upon the value of INITLVL) routine directly from the EOS initialization program before the EOS is restarted.

(d) Each task which is to be initialized in the "ready" state is placed on the ready list so that when initialization is complete, the ready list is structured in task priority order. The priority order is the order in which task names are entered into the ready list.

4.20 The EOS initialization routine will then reinitialize the input/output circuitry in case improper initialization sequences during task initialization may have caused channel errors.

4.21 Prior to transferring control to the EOS dispatcher, various miscellaneous functions and checks are performed to verify that certain conditions exist:

- Activate the write-protect function

- If system dynamic memory was initialized earlier, initialize the device control blocks, direct memory access, and channels

- Any DBS switches that are required are switched

- Depending upon the value of INITLVL, various regions of temporary memory are cleared (some regions are cleared on every initialization)

- Initialize the EOS program timer
- Zero the interrupt counter.

4.22 At this point the system is in control. The EOS out-of-service interrupts are initialized, and the MAICCI program returns control to the CINIT program at entry point INITAER.

4.23 Control of the initialization has been returned to the CINIT program and history checks are made to determine the severity of the initialization required by the EOS initialization routine. The following actions are performed, some of which are dependent on the results from the history checks:

- When the INITLVL equals LBOOT, the INITLVL is cycled and flagged if a lock-up occurs.
- The variable portion of an output message to be printed is prepared.
- The SSP keys are initialized.
- The interrupt counter is initialized.
- The dynamic display of branches in the display buffer is restored (it has been inhibited until now to allow the initialization counting display).
- All blocked interrupts are enabled.

4.24 The sanity of the off-line 3A CC is checked to determine if the 3A CC is reasonably capable of functioning (executing some code). A subroutine (OSANITY) of CINIT performs this check by exercising as much of the 3A CC as possible in a reasonable amount of time. The CINIT assumes that the self-checking circuits will detect errors. A maximum number of data manipulation logic functions are exercised while minimizing the number of microinterprets within the shortest possible execution time. The SANITY check is executed on the on-line 3A CC during initialization to determine if it is reasonably sane before allowing it to assume control. The SANITY check is executed on the off-line 3A CC to determine if it should be removed from service.

4.25 The CINIT program transfers control to the EOS initialization routine MAICCI via entry point INITAE to provide for the initialization of the EOS and application tasks to resume normal processing.

4.26 When all of the postmortem data has been collected the postmortem dump is printed on the terminal. This does not signify the end of the initialization interval. The end of the interval is signified by the output message INIT RCOVRY COMPL. This message marks the end of the initialization although normal processing may have been in progress for some time.

Kernel Audit

4.27 The kernel audit verifies the integrity of several doubly-linked lists containing process and task information used by the EOS kernel. The audit is executed during the processor initialization if the initialization level (INITLVL) value is between LCRIT and LMSG. The MAICCI program calls the Audit Controller (AUDCON) routine (prior to dynamic memory initialization) which invokes specific audits. A return code from the kernel audit is then used to determine whether or not the EOS kernel data base should be rebuilt. The kernel audit verifies the integrity of the EOS kernel data base by performing the following functions:

(a) If the initialization is a result of the system dynamic memory being full, the kernel audit frees all messages from any process which has queued more than 100 messages.

(b) Verifies all system file tables and that each device in an INITDVCB is valid.

(c) Verifies each device in the device equipment table and that its device control block (DVCB) is correct.

(d) Verifies that the system name in the file control block and tables is a valid system name linked off the process descriptor.

(e) Verifies that all DVCBs, except those which are neither interrupt driven nor polled, are in an INTDVCB.

(f) Verifies that every entry in the system name table points to the correct process descriptor.

(g) Verifies the linked list in the process descriptor.

(h) Verifies all system processes and tasks are in the Operating System Prototype Table (OSTABS) and then sets the name priority, time slice every MRF, INITROUT, and PSINIT.

(i) Audits system dynamic memory tables verifying that the header information is correct in each table.

(j) Verifies that the previous and next pointers in the headers for all blocks of memory in the list are valid.

(k) Verifies that the block of memory is within the allowed range of system memory.

(l) Verifies the consistency of the type of blocks within a list.

(m) Verifies that the chain-start points to the first and last memory blocks.

(n) Verifies that the timers are active and in ascending order (if order requested).

(o) Verifies the linked list continuity (forward and backward).

(p) Performs SANITY checks on timer entries.

4.28 The success of the kernel audit is flagged and control is returned to the MAICCI program.

Dynamic Memory Initialization

4.29 If the kernel audit fails or if the value of INITLVL is greater than LMSG, the system dynamic memory is initialized. In this case, the dynamic memory is cleared and all task descriptors and device control blocks are reentered into dynamic memory along with data specified by the application user in OSTABS. When the value of INITLVL exceeds the set value of LMSG by one, a memory audit is performed to identify any memory locations containing bad parity. The bad parity is corrected for any location having bad parity. The following actions occur during the dynamic memory initialization:

(a) The program timer is set to provide enough time for the memory audit.

(b) If bad parity is detected, the data is read and written back with the parity corrected.

(c) All of the system dynamic memory is put on the free list.

(d) Process descriptors are initialized for critical processes.

(e) All events are aborted.

C. Bootstrap Program (CIPL)

4.30 ***Introduction:*** Bootstrapping is the most severe initialization (Fig. 4) and results in either a complete memory reload or a selective partial memory reload so that the EOS and its application may resume normal processing. Bootstrapping can occur as the result of certain hardware conditions, eg, repeated time-outs by the program timer or by an application task utilizing options of the INITCC macro. The EOS does not automatically cause a bootstrap. A full bootstrap may occur, in which the system is entirely loaded. A partial bootstrap is a selective memory reload in which only mutilated memory (over 4k word or block boundaries) is identified through the use of checksums and reloaded from the tape cartridge. The nonmutilated memory is not loaded.

4.31 Bootstrapping is a process which reloads all or selected 4K blocks of memory from the data cartridge in the cartridge tape unit. The files on the data cartridge essential to the system reloading are the generic, translation, and patch files (named GENERIC, TRNSLN, and PATCH, respectively). In the past the generic file was required to reside on track 0 of the tape cartridge and was not permitted to continue onto the next track. Translation data could be located anywhere after the cartridge directory file on track 2. Thus the bootstrap program was able to load memory from the first three tracks of the cartridge. The amount of memory that could be loaded from these files is restricted by the physical track length of the cartridge (blocks per track) and the amount of information that can be stored in each block (words per block) for the possible block sizes. The generic and translation data is stored on the tape cartridge in subfile units, where a subfile is the number of blocks which contain 4K words of data. This must be taken into account when determining if the amount of space available on the cartridge tape is sufficient to accommodate these files.

Presently to accommodate EOS applications which require a large generic, it is necessary to allow the generic file to reside on both track 1 (GENERIC2 file) and track 0 (GENERIC file). When bootstrapping the system with the two generic files, the files GENERIC, GENERIC2,TRNSLN, and PATCH are loaded.

4.32 Bootstrapping the 3A CC consists of loading memory from a cartridge tape unit using the 3A CC as the controlling device. Therefore, a reload capability is designed into the system to retrieve program data from the cartridge tape unit. Since the 3A CC is a microprogrammed system, a small sequence of microcode is dedicated to the reload mechanism. Loading files from the cartridge tape occurs in a certain order, depending on the number of cartridge tape units available and which files are included on the cartridge tape. If only one tape unit is available to the bootstrap operation, the files are loaded in the following order: GENERIC, GENERIC2, TRNSLN, and PATCH. The GENERIC and PATCH files are mandatory files, while inclusion of the other two files is application dependent. If both tape units are available, one tape unit will perform the loading of the GENERIC and GENERIC2 files while the second tape unit will load the TRNSLN file. When bootstrapping the system with only one generic file, the PATCH file is read into memory after the loading of the GENERIC file has been completed. If two generic files exist on the cartridge tape, loading of the PATCH file begins after both generic files have been loaded. This enables both generics to be loaded and retains the EOS convention of not permitting the translation file to be patched.

Bootstrapping Sequence

4.33 The CIPL program consists of three functional loading parts. The first part is microcode initiated and is responsible for loading the first sequence or segment of code from the cartridge tape into transient call store. The second part is the segment just loaded and it is responsible for loading the remaining CIPL system memory loading and associated programs. The third part is responsible for loading selective blocks of memory (partial loading) as determined by a checksum procedure. After an initialization of the 3A CC occurs, the microstore initialization program determines if both ISC bits (ISC1 and ISC2) are set, and if the bits are set the microcode initiates a bootstrap. The cartridge tape unit associated with the 3A CC being bootstrapped is initialized and the tape is rewound to the beginning of the cartridge. The microcode starts reading the tape. The first 128 words are read from the tape and stored at address locations 1 through 128. The microprogram then transfers control to location 6 which is the first executable word read from the tape. If the microcode experiences an error while executing, the program timer will time out and initiate a restart of the bootstrap sequence.

4.34 The second part or segment of the bootstrap sequence consists of 122 executable words. This sequence is used to unload program data from tape until the next segment is completely loaded into memory. This segment is capable of loading the remainder of the bootstrap sequence. The first and second segments are capable of some error recovery. Since the bootstrap file is duplicated on tracks 1 and 2, if an error is encountered the bootstrap program will stop reading, issue a backspace, and restart the read on the other track.

4.35 Program control is passed to the second part of CIPL to complete the loading of the bootstrap program, system initialization programs, checksum data, and miscellaneous data used in the third part of CIPL.

4.36 Successful completion of the second part of the bootstrap sequence ensures that the common CC initialization program (CINIT) is properly loaded. Control is passed to CINIT (entry point INITBOOT) for hardware initialization, a SANITY check on the active 3A CC, and disabling of the other (standby) 3A CC (to prevent interference) before further bootstrap action occurs in the active 3A CC. Upon return from CINIT, a checksum test is performed upon the stores. Those blocks of memory determined to have mutilated data will be flagged for reloading by the third part of CIPL. However, if requested, the entire memory will be reloaded regardless of the checksums (reload options are possible via the SSP keys or via the INITCC macro).

4.37 The third part of CIPL loads selective memory blocks into memory in an effort to more efficiently load the memory using the processor capability. To reduce the time needed to recover the system, both TDCs may be used to reload the system. For some applications up to three tracks of the tape may be utilized to store programs, system parameters, and overwrites. One TDC may

be used to reload from one or two tracks while the other TDC reloads from the other track. However, if the system has one TDC out-of-service, the system can reload using the available TDC but a longer recovery time is necessary. The bulk of the generic, translation, and patch (overwrite) files are reloaded and control is returned to CINIT.

4.38 ***Off-line Bootstrapping:*** The EOS is capable of off-line bootstrapping. The off-line main store memory may be loaded with data from the cartridge tape, and verification may be made that the bootstrap process functioned successfully for certain bootstrap actions. The off-line bootstrapping can be accomplished by two methods: (1) by indirect loading of files and (2) by a simulated bootstrap. Certain precautions must be observed when performing an off-line bootstrap. For the off-line bootstrap to proceed, the system must be in the locked state. Also, prior to the initialization of the off-line bootstrapping, the tape utilities must be allowed by inputting the command ALW:TAPEUTIL via the input terminal. After the completion of the bootstrapping, the tape utilities are stopped and both tape data controllers (TDCs) are initialized. Upon successful completion of the off-line bootstrap, a switch to the other 3A CC may be desirable. A switch may be accomplished (both CCs must be in the MANUAL mode) by the input command SW:INIT or by SW:INIT:STABLE which causes a level 1 initialization or a level 3 initialization, respectively.

4.39 ***Indirect Loading:*** The indirect loading of files method (either the GENERIC or the TRNSLN file) is accomplished (using input command LOD:OMAS;TAPEa;file where "a" is the tape number and file is either GENERIC or TRNSLN) by reading the specified tape file from the on-line or active 3A CC. The specified file is copied to the off-line 3A CC and a check is made to verify that the file data copied is correct. This method of bootstrapping does not have a test to verify the success of the bootstrapping procedure.

4.40 ***Simulated Bootstrap:*** The simulated bootstrap method is accomplished through a simulated bootstrap on the off-line 3A CC (by the input command LOD:OMAS;BOOTFULL). In addition to loading the GENERIC and TRNSLN files, the PATCH file is also loaded when the simulated bootstrap method is used. This method also has the additional feature of verifying that the bootstrap was successful for the cartridge tape specified. A reload based on checksums or a complete reload (FULL specified in the input command) of memory is possible depending upon the input command. The bootstrapping process will attempt to use both tapes for the loading process just as it does during an on-line bootstrap, and to load only from the off-line tape, the on-line tape must be removed from the cartridge tape transport.

4.41 ***Off-line Bootstrap Procedure:*** When off-line bootstrapping is necessary, refer the appropriate procedures in the documentation for the specific application of the EOS. Certain precautions must be observed to avoid service interruptions or mutilation of data in the main store memory.

4.42 ***Bootstrap Results:*** The results of a bootstrap is the loading of memory with good data, or the overwrite of bad memory with good data from the cartridge tape. The bootstrap operation requires no main memory since it is loaded into temporary memory and cleared as part of the system initialization. The EOS and its application may now resume normal processing.

5. POSTMORTEM DUMP

5.01 ***General:*** The postmortem dump is the primary source of data pertaining to the cause of an initialization. The dump may also be the basis of decisions which must be made in resolving a service-affecting situation in a central office. The dump is printed out via a terminal (TTY) either automatically, when the system experiences an initialization, or in response to an input message OP:POSTMORT. The dump consists of two 64-word groups or arrays. Each group is formatted into two 4-by-8 arrays (four lines by eight columns of data words). These two groups of data words represent the processor states during the last series of initializations.

5.02 The first six rows (1 through 6) of the postmortem dump (Table F) represents the data that is common to all 3A Processor-based systems. The last two rows (rows 7 and 8) of the dump contains data which is unique to the particular EOS application at the time of the initialization. For ease of readability the dump is formatted into two 4-by-8 arrays.

5.03 ***Processor Initialization Program - CINIT:*** The majority of the postmortem

dump initialization data is gathered by CINIT. Upon entry CINIT immediately saves the contents of the store error register (SER, Table K) in temporary data word SAVSER. The register is used by the microcode to save SER from Main Store 0 and must be retrieved by CINIT before the execution of certain instructions which use this register. After SER is saved, CINIT corrects all parity errors in the contents of the general purpose registers, and stores them into a designated data area. Before gathering further initialization data, CINIT clears the first (top) postmortem data area. Therefore, in certain situations, postmortem words of all-zeros may be interpreted as data words which have not been gathered. An analysis of the dump should reveal if an all-zeros data word is logical or if it indicates uncollected data. For example, if the system is not four or more levels down into the hold-get (HG) stack, some (or all) of the HG information may contain zeros which represent uncollected data. But initializations will occur that result of memory errors, and in this case the store error register will contain zeros (logical).

A. Gathering Postmortem Data

5.04 After clearing the first postmortem area, CINIT gathers initialization data from various sources in the order of presentation in the dump. There are three exceptions to this: (1) the initialization level, gathered by MAICCI, (2) the off-line program address register, and (3) the off-line error register. The off-line registers data cannot be gathered until the CC on-line decision (which CC goes on-line) is made.

5.05 During the process of gathering postmortem data, three important decisions must be made. The first decision involves determining which main store (in multiple store environments) was being accessed at the time of initialization. If the first main store (Main Store 0) was being accessed, the SER of interest has already been gathered by microcode and saved by CINIT in SAVSER register (word), and CINIT then copies the contents of SAVSER into the postmortem area. If, however, a supplementary main store (SMAS) was being accessed, CINIT must send an input/output order to retrieve the SER from the SMAS. The SER from Main Store 0 is still saved in the SAVSER register. The second decision involves determining if a stop-and-switch action is taking place. In this case, all data must be gathered from the now off-line (previously on-line) processor. This is accomplished by several CINIT subroutine loops which index into tables of various maintenance channel orders (REGTBL and LGPTBL). After these subroutine loops are completed, CINIT proceeds with the CC initialization.

5.06 The third decision is another (second) test for a stop-and-switch action. A pass of this test means that the hold-get data has been collected previously from the now off-line processor) and no further action is required to gather postmortem data. Control is passed to the EOS Initialization Program MAICCI.

5.07 In addition to gathering data concerning the conditions of the system, MAICCI determines if the current initialization is the first of a sequence of initializations. This decision is critical because the copy of the initialization data, from the top of the postmortem area to the bottom, is performed only under this condition. All sequential initialization will be stored in the top postmortem area. The postmortem copy decision is made immediately after the completion of data collection to reduce the possibility of losing the first initialization data. This decision is based on the value of data word POSTMCPY located in common systems temporary store definitions (CTSD). After completion of a first initialization sequence, MAICCI changes the value of POSTMCPY so that subsequent initializations will not destroy the data pertaining to the first initialization. The value of POSTMCPY is reset at the end of the initialization interval. The last postmortem operation performed by MAICCI is the copying of the postmortem data and POSTMCPY into the off-line processor. This provides more reliability for the postmortem data in multiple stop-and-switch occurrences.

B. Postmortem Dump Interpretation

5.08 ***Postmortem Validity:*** Before interpreting the postmortem data, the validity of the dump should be checked by interpreting the valid data indicator (INDR) word (Table D, row 5, word 8). The various status codes for the valid data indicator word are described in valid data indicator Table I. Another indicator of valid postmortem data is the comparison of the time indicated in the time field of the INIT level, time word (Table D, row 1, word 1 of the dump) with the time indicated in the RCVRY CU INIT message (via the terminal) associated with the postmortem dump of interest. The time indicated in the dump is

stored in Binary Coded Decimal (BCD) form as shown in Table D. Bits 4 through 9 represent seconds, while bits 10 through 15 represent minutes (past the hour).

5.09 ***Interpretation:*** After assuring the validity of the postmortem dump, interpretation of the dump to determine the cause of the initialization may begin. Refer to Postmortem Dump Table F and to the following for certain interpretation deta:

(1) The system state register (SYSTATE) and miscellaneous (MISC) bits (row 1, words 3 and 4) may be used to determine the state of the CC at the time of the initialization. The active (on-line) CU can be found by checking SYSTATE bit 12 or miscellaneous bits word bit 8. Refer to Tables L and M for the layout of these words.

(2) The Timer Interrupt (TI) register word (row 1, word 5) is mainly of interest if its contents indicate a first or a second timeout, which is noted by the setting of bits 4 and 15, respectively. Refer to Table N for a layout of the TI word.

(3) In a multiple store environment, the store error register (SER, row 2, word 5 and Table K) is gathered from the main store, which is being accessed at the time of the initialization. If the program receives an input/output error in attempting to retrieve the SER from a supplementary main store (SMAS) an error code (FEFE) will be written into the SER postmortem word. The saving of the contents of the SER and the store address registers (SAR , row 3, words 5 and 6) is dependent on the processor having certain microcode (board numbers 4C200 to 4C203 for ESS, or 4C200 to 4C205 for EOS. The layout of the SAR word is shown in Table O. If this microcode is not equipped, these postmortem words will contain the contents of the C register (SER) and the maintenance channel buffer (SAR).

(4) The off-line CU PA and ER registers (row 5, words 4, 5, 6, and 7) may be of use in multiple switch occurrences and during execution of an off-line program such as a diagnostics program.

(5) The EOS parameter word SYSCOUNT (row 7, word 1 bits 8 through 16) is set to zero at base level and to one or greater at interrupt levels. The number of increments above one is equal to the number of interrupts that are pending.

(6) The EOS parameter word MISTIME (row 7, word 1 bits 0 through 7) indicates the number of occurrences in which the timer interrupt has interrupted itself.

(7) Initialization caused by an overflow of system memory may be determined from the value of the RMGSYM parameter which indicates the remaining system memory (row 7, word 3).

(8) The interrupt status (IS, row 1, word 6) and the interrupt mask (IM, row 2, word 6) should be combined for observation of unusual situations, ie, an error interrupt (bit 5) being masked out, covers certain hardware problems.

(9) If a system timeout occurs, the priority of the process maintenance (MAINT) priority (row 7, word 2) is important in determining if this key process was being locked out by other processes. These other processes can be determined by inspecting the ready list information.

(10) Comparison of the store address register (SAR, row 3, words 5 and 6) and the program address (PA) register (row 1, words 7 and 8) will indicate if a data word was being accessed or if the next instruction was being fetched.

(11) The request list length data provided for the first three tasks or processes (rows 7 and 8) on the ready list indicates how much work is on queue for those tasks and also gives an indication of the task load on the system.

(12) On every system initialization the 3A Processor microcode saves the contents of the PA register in the AK register. Under stop-and-switch conditions, this operation is only performed in the on-line CU. Therefore, under these conditions, the PA of interest (off-line CU) would be the contents of the PA register, and not the contents of the AK register. If however, the stop-and-switch was preceded by a normal initialization, the AK register in the off-line CU may contain certain useful information, ie, the address of the program executing at the time of the first initialization. To provide more useful data, the postmortem collection routines, under

stop-and-switch conditions, will save the contents of the PA (row 1, words 7 and 8) and the contents of the AK registers (row 5, words 4 and 5 for the off-line CU). If an initialization did not take place prior to the stop-and-switch occurrence, the off-line PA words will contain irrelevant data.

5.10 ***Manual Request of Postmortem Data:*** The saving of the postmortem data is dependent upon the system successfully proceeding through the copy decision point in MAICCI. Automatic printing of the postmortem is initiated upon completion of the gathering of the postmortem data. The completion of the initialization is signified by the printing of the message INIT RCOVRY COMPL. This interval is established by a counter which is decremented on every pass of the base level monitor program (CBLM). On a one-to-zero transition of this counter the initialization level is reset. However, if it is necessary to print or display the dump prior to its automatic printing, the dump can be displayed via the OP POSTMORT message. In addition to the dump being displayed upon request, the contents of the PA and ER registers from the processor in which the fault was encountered may be displayed via the message RCOVRY CU INIT. This data is useful in situations where the system does not stabilize prior to the point where postmortem data can be gathered and printed out via the terminal.

5.11 Refer to the Input/Output Message Manuals (IM-4C001-01 and OM-4C001-01, respectively) for details of the printouts of messages received via the terminals or input via the terminals.

6. GLOSSARY

6.01 The following terms apply to the EOS initialization sequence.

Active —A 3A CC is in the active state when it is the on-line processor.

Application —A set of functional system programs which use the services of EOS.

Bootstrap —The most drastic initialization action, resulting in a memory reload.

Duplex Bus Selector (DBS) —An electronic switch which transfers control of a peripheral device from one CC to the other CC.

Event —A signal to a task that a previously specified state or condition change (event) has occurred.

Initialization Interval —The initialization interval is a period of time following a system initialization during which the system is considered to be in an unstable state.

Initialization Routine —A subroutine associated with a task that is called exactly like an event routine during a system initialization (prior to operating system restart). This routine is used to ensure that a task can initialize peripherals or circuits without the presence of interrupts and also allows an application to change values in the OSTABS after a bootstrap.

INITLVL —An initialization level (INITLVL) counter whose value determines the severity of action taken during a system initialization. Each initialization occurring during an initialization interval will increment the counter.

LBOOT —A particular value to which INITLVL counter is set when a bootstrap has occurred (highest level of initialization).

LCRIT —A particular value of INITLVL which, when reached, causes a specified level of initialization (lowest level).

LMSG —A particular value of INITLVL which, when reached, causes a reinitialization of all tasks.

Maintenance Reset Function (MRF) —A hardware generated signal within the 3A CC that initiates certain initialization functions to be accomplished by the hardware and microprogram of the 3A CC; the first step in an initialization.

OSTABS —Operating system prototype tables used by the application to define system resources, configuration, parameters, etc.

Postmortem Dump —A printout (dump) via a terminal of the contents of certain registers which represent the internal state of the active CC when a hardware or software fault occurred.

Psuedo-Initialization Routine —A standard type of task initialization provided by the EOS for those tasks that do not require a real initialization routine. This option is provided to allow the use

of all operating system services and is satisfactory for most application tasks.

Standby —A 3A CC is in standby when its memory is up-to-date and being updated by the on-line 3A CC via the serial maintenance channel. It causes a maintenance reset function (MRF) in the other central control.

Task —An execution module scheduled and run asynchronously at a specified priority.

Unavailable —A 3A CC is unavailable (UNAV) when it has been forced off-line manually from the System Status Panel (SSP).

TABLE A

ABBREVIATIONS AND ACRONYMS

ABBREVIATION	MEANING
ALW	Allow
BCD	Binary Coded Decimal
BHC	Block Hardware Check
BIN	Block Interrupt
BOT	Beginning of Tape
BTC	Block Timer Check
CB	Control Block
CC	Central Control
COMPL	Complete
Contd	Continued
CPY	Copy
CRC	Cyclic Redundancy Check
CTSD	Common Systems Temporary Store Definition
CU	Control Unit
DBS	Duplex Bus Selector
DMA	Direct Memory Access
DML	Data Mainipulation Logic
DR	Data Ready
DV	Device
EOS	Extended Operating System
ER	Error Register
FRZ	Freeze
HG	Hold Get
INDR	Indicator
INIT	Initialization
IM	Interrupt Mask Register/Input Message Manual
IS	Interrupt Set Register
ISC	Initialization Sequence Counter (1 or 2)
ISO	Isolate Bit
I/O	Input/Output
K	Kernel, Thousand
LOD	Load
LOF	Lock Off-Line
LON	Lock On-Line
MAIDTA	EOS Maintenance Data Layout
MAINT	EOS Maintenance Task
MAR	Microaddress Register
MAS	Main Store
MCH	Maintenance Channel
MINT	Microinterpret Mode
MIR	Microinstruction Register
MISC	Miscellaneous
MRF	Maintenance Reset Function
MSR	Maintenance State Register
OM	Output Message Manual
OMAS	Other Main Store
OOS	Out of Service
OP	Output Postmortem Dump

TABLE A (Contd)

ABBREVIATIONS AND ACRONYMS

ABBREVIATION	MEANING
OSTABS	Prototype Operating System Table
PA	Program Address Register
PCH	Parallel Channel
PH	Parity High
PL	Parity Low
PR	Print
PT	Program Timer
RAR	Return Address Register
RCOVRY	Recovery
RCVRY	Recovery
RU	Return Address Register Update
SAR	Store Address Register
SER	Store Error Register
SMAS	Supplementary Main Store
SS	System Status Register
SSP	System Status Panel
STP	Stop
SYS	System
TDC	Tape Data Controller
TLM	Trouble Locating Manual
UPD	Update
UTIL	Utility/Utilities

TABLE B

SYSTEM INITIALIZATION LEVELS

INITVL	SEVERITY	ISC VALUE	SOURCE	EFFECT ON EOS AND APPLICATION
< LCRIT	No Switch	ISC1 = 0 ISC2 = 0	Automatic	Below the level of LCRIT, only the task in control at the time of the initialization is initialized (plus any task for which the EVERY/MRF option was invoked).
> = to LCRIT	Switch	ISC1 = 1 ISC2 = 0	Automatic or Manual	Critical initialization level and every task in the system is initialized at this level or higher.
> = to LMSG	No Switch	ISC1 = 0 ISC2 = 1	Automatic or Manual	Dynamic system memory is zeroed at this level or higher. This implies that all transient intertask messages are lost. LMSG can be viewed as the level at which the EOS takes the most drastic action possible short of a bootstrap.
> = to LBOOT	No Switch	ISC1 = 1 ISC2 = 1	Automatic or Manual	A bootstrap will occur at this level or higher. This happens either if LBOOT is attained via a sequence of lower level initializations or if INITLVL is set to LBOOT via the INITCC macro. Note that if INITLVL is greater than or equal to LBOOT when another initialization occurs, another bootstrap will occur. INITLVL is never zeroed by the EOS.

TABLE C

ERROR (ER) REGISTER

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
-	-	-	-	-	-	-	-	-	-						

		ERROR (ER) REGISTER	NOTES		
BIT	SYMBOL	FUNCTION	1	2	3
0	TODER	TO DECODER ERROR	X		
1	FRMDER	FROM DECODER ERROR	X		
2	IBER	IB X, Y FIELD ERROR	X		
3	BUSER	BUS PARITY ERROR	X		
4	DMLER	DML MATCH ERROR	X		
5	MARERM	MAR ERROR	X		
6	CLKER	CLOCK ERROR	X		
7	---	MY STORE ERROR A	X		
8	MADER	MAR-RAR MISMATCH ERROR	X		
9	FRER	FUNCTION REG PARITY ERROR	X		
10	SRPER	STORE READ PARITY ERROR		X	
11	MSTRER	MY STORE ERROR		X	
12	MFSTM	MY STORE FAST TIME-OUT		X	
13	BAER	BRANCH ALLOWED ERROR		X	
14	OWRTER	OTHER STORE WRITE PROTECT			X
15	OSTRER	OTHER STORE ERROR			X
16	OFSTM	OTHER STORE FAST TIME-OUT			X
17	IOMLTER	I/O MULTIPLE CHANNEL SELECT			X
18	PTRER	PT RESET RECEIVED BY ON-LINE CC			X
19	SWER	SWITCH RECEIVED BY ON-LINE CC			X
20	IOER	I/O CHANNEL ERROR (PL)			X
21	IOPARER	I/O BAD PARITY RECEIVED (PH)			X

Notes:

1. Switch to standby 3A CC
2. Hardware initialization
3. Interrupts

TABLE D

INITIALIZATION LEVEL - TIME

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MINUTES						SECONDS						RECY	INIT LVL		

BITS	MEANING
0, 1, 2	Initialization Level: 000 <LCRIT 010 >=LCRIT 100 >=LMSG 100 >=LBOOT
2	Recycle
4-9	Seconds Past the Hour
10-15	Minutes Past the Hour

Note: Data is stored in Binary Coded Decimal (BCD) form.

TABLE E

SYSTEM STATUS REGISTER (SS REG)

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
-	-	-	-	-	-	-	-	-	-						

BITS	SYMBOL	DESIGNATION	MEANING
0	AME	Address Match Enable	Enables match between store address register and address input register.
1	BHC	Block Hardware Check	Disables output of error register (bits 0-13).
2	BIN	Block Interrupt	Disables all interrupts.
3	BTC	Block Timer Check	Blocks inputs and outputs from the program timer.
4	DME	Data Match Enable	Enables match between store data bus and data input register.
5	HLT	Halt	Drives HALT control panel lamp which indicates the 3A CC is in a program halt condition.
6	ISC1	Initialization Sanity Check 1	Checks sanity of hardware initialization routine. If a failure is detected, a switch to the other 3A CC will occur.
7	ISC2	Initialization Sanity Check 2	Checks sanity of hardware initialization routine. If a failure is detected and ISC1 = 1, the main memory will be reloaded.
8	LOF	Lock Off-Line	Disables input/output channel to prevent interference with the on-line 3A CC and allows power to be removed via the control panel POWER key.

TABLE E (Contd)

SYSTEM STATUS REGISTER (SS REG)

BITS	SYMBOL	DESIGNATION	MEANING
9	LON	Lock On-Line	Forces all hardware switch messages to initialize this 3A CC and to keep it on-line.
10	MAN	Manual	Drives the MANUAL lamp on the control panel which indicates that the off-line 3A CC is in the manual mode.
11	MINT	Maintenance Mode	Blocks gating from microstore into the microinstruction register (MIR) and enables gating from main store memory into MIR.
12	CC	Central Control	Indicates which CC is on-line, either CC = 0 or 1, and controls various functions which protect the on-line 3A CC from the off-line 3A CC.
13	RJE	Reject	Drives the control panel lamp (REJ) which indicates that a panel operation was not performed.
14	STP	Stop	Jams the main store (MAR) to a maintenance address causing all zeroes to be read out of microstore.
15	DISA	Disable	Disables the input/output channels to prevent interference with the on-line 3A CC.
16	PRI	Privilege	Used in implementation of the privilege mode instruction execution (not used in EOS).
17	DISP	Display	Disables the gating from the program address (PA) to the display buffer on all transfers.
18	BPC	Block Bus Parity Check	Disables the data bus parity checker.

TABLE E (Contd)

SYSTEM STATUS REGISTER (SS REG)

BITS	SYMBOL	DESIGNATION	MEANING
19	IPLTRK	Initial Program Reload Track	Used by microcode within the sequence which initiates program reloads from tape. The state of this bit determines which of the two identical tracks on the tape cartridge will be read. The bit is also used to determine when to perform the more drastic initializations of the main store (hard initialization) prior to reading data from the tape.
20	CC0	Central Control 0	ALWAYS 1 in 3A CC0 so that the program knows which 3A CC is running.
21	CC1	Central Control 1	ALWAYS 1 in 3A CC1 so that the program knows which 3A CC is running.

TABLE F

POSTMORTEM DUMP - FORMAT
PART 1

LINE (ROW)	COLUMN (WORD)							
	1	2	3	4	5	6	7	8
1	Init Level, Time	HG Reg	SYSTATE	Misc Bits	TI Reg	IS Reg	PA Reg (Bits) 19-16)	PA Reg (Bits 15-0)
2	R8	R9	R10	R11	SER	IM Reg	DB Reg (Bits 19-16)	DB Reg (Bits 15-0)
3	R13	R13	R14	R15	SAR	SAR	SS Reg (Bits 21-16)	SS Reg (Bits 15-0)
4	HG	HG+1	HG+16	HG+17	HG+32	HG+33	ER Reg (Bits 21-16)	ER Reg (Bits 15-0)

LEGEND

DB - Display Buffer Register
ER - Error Register
HG - Hold-Get Register
IM - Interrupt Mask Register
Init - Initialization
IS - Interrupt Set Register
Misc - Miscellaneous
PA - Program Address Register
R - General Registers
Reg - Register
SAR - Store Address Register
SER - Store Error Register
SS - System Status Register
TI - Timing Counter Register

Note: Lines (rows) 1 through 6 are common to all 3A CC-based systems. Rows 7 and 8 are unique to the application.

TABLE F (Contd)

POSTMORTEM DUMP - FORMAT
PART 2

LINE (ROW)	COLUMN (WORD)							
	1	2	3	4	5	6	7	8
5	HG+48	HG+49	Paging Segment Number	PA Reg (Bits 19-16) Off-line CU*	PA Reg (Bits 15-0) Off-line CU*	ER Reg (Bits 19-16) Off-line CU	ER Reg (Bits 15-0) Off-line CU	Valid Data Indr
6	R0	R1	R2	R3	R4	R5	R6	R7
7	Syscount (Bits 16-8) Mistime (Bits 7-0)	Process Maint Priority	Rmng System Memory (RMGSYM)	Kernel Audit Fault	Ready List Task 1 Id	PA Reg (Bits 19-16) Task 1	PA Reg (Bits 15-0) Task 1	Task 1 Request List Length
8	Ready List Task 2 Id	PA Reg (Bits 19-16) Task 2	PA Reg (Bits 15-0) Task 2	Task 2 Request List Length	Ready List Task 3 Id	PA Reg (Bits 19-16) Task 3	PA Reg (Bits 15-0) Task 3	Task 3 Request List Length

LEGEND

CU - Control Unit
ER - Error Register
HG - Hold-Get Register
Id-Identification
Indr - Indicator
Maint - Maintenance
PA - Program Address Register
R- General Register
Reg - Register
Rmng - Remaining

Note: Line (rows) 1 through 6 are common to all 3A CC-based systems. Rows 7 and 8 are unique to the application.

*Contains off-line AK if a stop-and-switch occurred. (Off-line PA is contained in words 7 and 8 of line 1.)

TABLE G

CINIT SUBROUTINES

FUNCTION	ENTRY POINT	DESCRIPTION
Switch CC (TTY Request)	SWCU, SWSYC	Handles the TTY requests (SW:CU) to switch the active-standby status of the two CUs. A switch occurs only if the other CC is in the standby state. If an unconditional (UCL) switch is specified, a normal switch is attempted first, and if it fails, a second attempt is made via a delayed switch. This causes an update to be performed and then attempts a switch after the update. If the system is in a locked state, the delayed switch is denied and no switch occurs.
Switch CC	UPD%SW	Handles the normal software request to switch the active-standby status of the two CUs. The other CC must be in the standby state. An immediate switch is attempted, and if it fails, a delayed switch is attempted as described above. Other entry points to the normal switch subroutine are described below.
	SWCCACHK	Checks to determine if a switch is permitted by the application and, if so, a switch is made. Otherwise the switch is denied.
	SWCCUPD	Switch CUs only if the system is in the update state.
	SWITCHCC	Permits a switch to be made only if the off-line CC is in the standby state.
	CSWCU	Performs all functions necessary for switching prior to switching control to the other CC, and then switches control to the other CC.
Test Access to Off-line MAS	OMASTEST	Tests the off-line MAS to determine if data can be written into and read from the MAS without an error being detected. Also tests whether or not each write-protected block can be written without generating a write-protect error.
Error Print Check	ERRPRTCK	Determines if a REPT ERR message should be printed as a result of an audit failure. The message is printed each time the audit fails after it has passed once.
Increment Display Buffer	AD1TODB	Provides a visual indication via the SSP that an initialization is actually in progress. The CC panel display buffer (DB) is incremented (AD1TODB) and gated to the SSP display buffer.
On-line Subroutine		Provides a general check of the CC and assures that it can successfully execute some code. The SANITY subroutine is executed in the on-line CC during initialization to ensure that the initializing CC is reasonably sane before allowing it to gain control. The SANITY subroutine is executed on the off-line CC to determine if it should be removed from service. If either CC fails the sanity check it is stopped and removed from service.

TABLE G (Contd)

CINIT SUBROUTINES

FUNCTION	ENTRY POINT	DESCRIPTION
On-line Subroutine (Contd)	SANITY	Determines if the CC can successfully execute some code. Much of the CC capability is checked, and it is assumed that the self-checking functions are operating and most paths are exercised. The check circuits will detect many faults if any exist. However, validity checks are performed to further verify successful operations. The testing strategy is to exercise to, from, and miscellaneous crosspoints as much as possible. Testing is done by sending ones and zeros through the gating paths and by exercising the data manipulation logic functions as much as possible in a short period of execution time.
	REGTEST	Verifies the ability to uniquely gate to and from the 14 general registers.
	HGET	Obtains the address of the hold-get area one level higher and stores the address in registers R14 and R15. The present hold-get address is stored in register R13.
	GETIME	Obtains the time from the timing counter (TC) register and adjusts it for unusual incrementing as used by the SANITY subroutine.
Off-line Subroutine	OSANITY	Performs a sanity check of the off-line CC. First a test is made to determine if the CC is in the out-of-service state, and, if so, there is no need to execute the sanity check.

TABLE H

INITIALIZATION CONTROL (INITCTL)

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-	-	-	-	-	-	-	-	-	-	-	-				

BITS	SYMBOL	MEANING
0	INIT_ALL	Set by INITCC macro if not a checksum bootstrap.
1	INIT_BKDT	Set by INITCC macro if a backdate is required.
2	INIT_TSK	Set by INITCC if INITCC caused initialization.
3	INIT_OSS	Off-line CU to remain out of service.

TABLE I

VALID DATA INDICATORS

INDICATOR	MEANING
F0F0	Normal completion of initialization data collection under first-of-a-sequence conditions. A copy of the top postmortem into the bottom postmortem has taken place.
F0F2	Normal completion of initialization data collection under the program interpretation of a stop-and-switch situation. All applicable data has been collected from the off-line (previously on-line) control unit.
C0C0	Normal completion of the CINIT phase of initialization data completion, no postmortem copy has occurred.
B0B0	A stop-and-switch was detected by CINIT. In an attempt to read data from the major registers (TI, IS, IM, etc) from the off-line control unit, a maintenance channel error was encountered. Therefore, the majority of the postmortem data may be unreliable.
A0A0	Similar to B0B0 except that a maintenance channel error was encountered while executing a later section of CINIT. Postmortem data pertaining to registers 0 through 7 may be unreliable.
A0A5	A maintenance channel error was encountered in attempting to read the PA and ER registers from the off-line control unit. This code is executed during every initialization and is not related to the stop-and-switch situations mentioned above.
CODE	The recovery software has detected an application- requested bootstrap. Prior to initiating the reload, the current postmortem data has been saved in a protected scratch area in memory. This data has been retrieved upon return from the reload.

TABLE J

MAICCI SUBROUTINES

FUNCTION	ENTRY POINT	DESCRIPTION
Get Initialization Routine	Get_INIT_ ROUTINE_CRIT or NONCRIT	Determines address of a process initialization or a pseudo-initialization routine and places a pointer to INITPARM in RA0 and to the initialization routine in RA1.
Place Process in State Requested	INIT_PROCESS_ STATE	Places a process in the state requested by the initialization routine. State is passed as a parameter in INITPARM. The process is linked to the READY list if required.
Free Memory Blocks from Linked List	Tear_DWN_ LNK_LST	Free all memory blocks associated with elements of a linked list by returning the blocks to the free list.
Mark Requests	Mark_REQ	Marks as bold all requests that were received before the process was initialized. Any request retried with a response expected will not be marked as an old request.
Get Register Table	GETREGTBL	On a stop-and-switch of CCSs, places the contents of the off-line CC general purpose registers in the maintenance table.

TABLE K

STORE ERROR REGISTER (SER)

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-	-		-												

Note: All 13 bits of the SER are "OR" ed together into bit 7 (store error A) of the Processor Error Register.

BITS	MEANING
0	A chip select signal in an unselected memory module was enabled.
1	A functional test which monitors the control of a main store cycle has failed.
2	More than one main store command was decoded from the command register during a main store cycle.
3	The store complete signal was fired prematurely during a nonrefresh cycle.
4	An illegal state of multiple bits in the timing shift register was detected.
5	More than one memory module has responded to a main store memory access request.
6	A test which checks for both memory modules responding during a refresh cycle has failed.
7	The proper responses were not received from the memory modules for a particular command code which was received from the main store bus.
8	Incorrect address parity detected during a memory access cycle.
9	Incorrect data parity detected during write and blind write cycles. This test is inhibited during complement write and refresh cycles.
10	A check of the extended data parity high bit has failed (wide stores only).
11	A mismatch was detected between the two circuit packs which decode address bits 18 and 19 and determine which main store is to be selected.

TABLE K (Contd)

STORE ERROR REGISTER (SER)

BITS	MEANING
12	Not used.
13	Incorrect parity was detected in the parity low field during a write or blind write cycle. This test is inhibited during complement write and refresh cycles.
14,15	Not used.

TABLE L

SYSTEM STATE (SYSTATE)

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

BITS	MEANING
0	Off-line 3A CC is in standby
1	Off-line 3A CC is out-of-service
2	Off-line 3A CC is unavailable
3	Initialization timing interval is in progress
4	Off-line 3A CC is out-of-service due to a fault
5	Off-line 3A CC has been manually removed from service
6	Off-line 3A CC memory is being updated by the system
7	Off-line 3A CC is being used by a program
8	Off-line 3A CC is in the manual mode
9	Indicates which 3A CC is on-line (active 0 - 3A CC No. 0 is active (on-line) 1 - 3A CC No. 1 is active
10	Off-line main store is out-of-service (OOS)
11	System Status Panel (SSP) is out-of-service
12	Maintenance channel is out-of-service
13	A 3A CC restoral is in progress
14	System Status Panel memory reload request
15	System Status Panel initialization request

TABLE M

MISCELLANEOUS BITS, INITIALIZATION DATA

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
		-	-	-	-										

BITS	MEANING
0	If 1, no switch occurred. If 0, a switch occurred.
1	If 1, an MCH message caused initialization.
2,3	If 01-A first timeout caused initialization. If 11 - A second timeout caused initialization.
4	If 1, control unit locked or forced on-line.
5	If 1, memory reload from tape caused initialization.
6	If 1, main store was out-of-date.
7	If 1, no access exists to other main store.
8	If 0 - Control Unit 0 initializing. If 1 - Control Unit 1 initializing.
9	If 1, maintenance channel failed while the off-line registers were being retrieved for the postmortem dump.
10-13	Not used.
14	If 1, memory reload request via System Status Panel.
15	If 1, initialization request via System Status Panel.

TABLE N

TIMING COUNTER (TI)

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
2ND TO	1ST TO	25MS COUNTER (CTR)						TIMING COUNTER (TI)							

BITS	SYMBOL	MEANING
0-7	TI	Timing Counter.
8-13	25CTR	25MS Counter.
14	1ST_TO	First CC timeout. When set by overflow from 25CTR, this bit causes a stop-and-switch.
15	2ND_TO	Second CC timeout. When set by overflow from 1st timeout or the 25CTR, this bit causes the CC to initialize independent of the on-line/off-line status of the CC.

TABLE O

STORE ADDRESS REGISTER (SAR)

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ROW & COLUMN ACCESS												CHIP		

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
-	-	-	-	-	-	-	-	-	-	-	-	MAS		MCD	

BITS	MEANING
0-2	Chip Select
3-14	Row and Column Access
15-17	Module Select (0-7)
18-19	Main Store Select