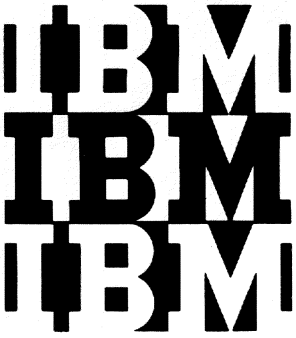
File No. S360-36
GY28-2041-4



Program Logic

Version 8.1

IBM System/360 Time Sharing System
Task Monitor

This publication describes the internal logic of the group of routines that make up the task monitor. The task monitor deals with the reception and control of those interruptions that directly concern the user of the system. It resides in virtual memory and is composed of ten modules.

Program logic manuals are directed to systems engineers and programmers who require descriptions of the internal logic of a system. Together with program listings, they can be used to examine specific areas of program logic for purposes of maintaining or altering the system.

PREFACE

The task monitor resides in virtual storage and directs the processing of task-oriented interruptions. This book attempts to make clear the internal logic of the group of routines which comprise the task monitor.

There are five sections in this manual. Section 1, the introduction, provides background information essential to an understanding of task monitor processing. Section 2, Method of Operation, contains a general overview of that processing. From this section the reader should be able to gain an understanding of how the task monitor accomplishes its objectives. Section 3 describes program organization. It consists of flowcharts and detailed routine descriptions. Section 4, Data Areas, contains a brief description of the control blocks used by the task monitor, and how they are used. Section 5, Diagnostic Aids, contains helpful information for the customer engineer; for example, a microfiche directory and layouts of the task monitor's enter tables and log.

Fourth Edition (September 1971)

This is a major revision of, and makes obsolete GY28-2041-3. This edition should be reviewed in its entirety.

The task monitor has been changed to improve handling of attention interruptions, program interruptions, and MCB headers. In addition, improvements to the HOLD and DROP commands have necessitated some changes.

This edition is current with Version 8, Modification 1, of IBM System/360 Time Sharing System (TSS/360) and remains in effect for all subsequent versions or modifications of TSS/360 unless otherwise noted. Significant changes or additions to this publication will be provided in new editions or Technical Newsletters. Before using this publication, refer to the latest edition of IBM System/360 Time Sharing System: Addendum, GC28-2043, which may contain information pertinent to the topics covered in this edition. The Addendum also lists the editions of all TSS/360 publications that are applicable and current.

Specifications contained herein are subject to change from time to time. Any such change will be reported in subsequent revisions or Technical Newsletters.

This publication was prepared for production using an IBM computer to update the text and to control the page and line format. Page impressions for photo-offset printing were obtained from an IBM 1403 Printer using a special print chain.

Requests for copies of IBM publications should be made to your IBM representative or to the IBM branch office serving your locality.

A form is provided at the back of this publication for reader's comments. If the form has been removed, comments may be addressed to IBM Corporation, Time Sharing System/360 Programming Publications, Department 643, Neighborhood Road, Kingston, New York 12401.

© Copyright International Business Machines Corporation 1970,1971

PREREQUISITE PUBLICATIONS

IBM System/360 Principles of Operation, GA22-6821
IBM System/360 Time Sharing System: System Logic Summary, GY28-2009

For more information about interruption handling macro instructions, SVC codes, linkages, time sharing support and virtual support systems, and system control blocks, see:

IBM System/360 Time Sharing System: Assembler User Macro Instructions, GC28-2004
Resident Supervisor PLM, GY28-2012
System Programmer's Guide, GC28-2008
Time Sharing Support System, GC28-2006
System Control Blocks PLM, GY28-2011

CONTENTS

ILLUSTRATIONS

INTRODUCTION

TASK MONITOR FUNCTIONS

The task monitor is a group of modules that receive and handle those interruptions that will be brought to the user's attention. They also provide services to the user, in the form of macro instructions, that enable him to handle such interruptions. The task monitor's queuing/scanning mechanism provides linkage to system or user-written interruption handling routines. These routines may be either dispatched immediately or queued for later dispatching. The task monitor also ensures that the integrity of each task is maintained throughout this procedure.

Except for SVCs 64-95 and 128-255, the interruptions listed in Table 1 are handled by the task monitor.

INTERRUPTIONS

Table 1. Interruption Types (Part 1 of 2)

	Type	Codes	Description
1	Program	1-15	Indicate error conditions; such as illegal operations, or overflow or underflow in arithmetic operations.
		16-17	Indicate paging (16) or segment (17) relocation exceptions. The dynamic loader must be called to resolve addresses in required page when it is first read into main storage.
2	SVC		Usually request system services; may indicate error conditions.
		0-63	Reserved for users who want to specify their own interruption-handling routines.
		64-95	Time sharing support system (TSSS) SVCs (not received by task monitor). These regulate ATs and provide system services for the virtual support system (VSS).
		96-127	Task monitor SVCs; documented in "SVC Interruption Processor," in this book.
		128-255	Handled by resident supervisor.
3	External		Result from intertask communication. A VSEND macro instruction is issued by task that wants to send message to another task.
4	Asynchronous I/O		Result from I/O activity that was not initiated by the resident supervisor. Generally, they occur when user presses "attention" key on his terminal.

Table 1. Interruption Types (Part 2 of 2)

	Type	Codes	Description
5	Timer		Occur at end of user-determined time interval which is set in one of the task monitor's clocks.
		Real	Real time is continuously decremented.
		Task	Task time is decremented only when task that set timer is in execution.
6	Synchronous I/O		Occur when I/O operations that have been initiated by resident supervisor are completed.
7	Virtual support system		Occur (1) when VSS command is invoked, (2) when task system programmer is connected to task (3) when TSSS reaches AT command embedded in code written by a system programmer.
8	Data set paging error		Occur when input paging errors are detected by resident paging mechanism. The task monitor will receive these interruptions when page cannot be read, device has been disabled, or data check error has occurred.

TASK MONITOR CONTROL BLOCKS

For easy reference, the abbreviations used in this book for control blocks are listed in Table 2.

Table 2. Control Blocks

Control Blocks		System ID
COM	(COMAREA) Communications area	CHACOM
DCB	Data control block	CHADCB
DEB	Data extent block	CHADEB
ICB	Interruption control block	CHAICB
ISA	Interruption save area	CHAISA
ITB	Interruption table	CHAITB
DE	Device entry	CHAIDE
RE	Request entry	CHAIRE
QE	Queue entry	CHAIQE
MCB	Message control block	CHAMCB
PDSA	Pushdown save area	CHAPDS
SDAT	Symbolic device allocation table	CHASDA
TMPSECT	Task monitor prototype control section	

LINKAGES

The linkages from one program to another within a task, which are of concern to the task monitor, and are most widely used in TSS/360 are type-I, type-II, and type-III, listed in Table 3.

Table 3. Linkages

Linkage	Description	Conventions
Type-I	Connects privileged program to privileged program or nonprivileged program to nonprivileged program	• Use of standard 19-word save area • Use of specific registers to transfer certain kinds of information • Use of BAS (BASR) instruction to transfer control • Preservation of register integrity
Type-II	Connects nonprivileged program to privileged program. Task monitor provides the save area; control is transferred via LVPSW instruction through resident supervisor	• Standard save area • Standard content and use of general registers • Standard method of transferring control from calling program to called program • Preserve register integrity
Type-III	Connects privileged program to nonprivileged program. The task monitor's leave privilege routine obtains and initializes standard save area. When called routine returns, control will be transferred to RSPRV routine of task monitor SVC interruption processor	• Standard save area • Standard usages of registers • Standard method of transferring control from calling program to called program

USER-SPECIFIED INTERRUPTION HANDLING ROUTINES

System/360 uses interruptions to provide efficient use of system resources and to handle errors. A series of macro instructions is available to the user who may want to supply his own interruption servicing routines.

Six kinds of interruptions can be handled in this way:

Interruption	Macro Instruction
Program	SPEC
SVC	SSEC
External	SEEC
Timer	STEC
Synchronous I/O	SIEC
Asynchronous I/O	SAEC

These macro instructions are used to define the user-built ICB, which contains:

- Type of interruption to be serviced
- Entry point of user-written servicing routine
- Address of user-supplied COMAREA
- (I/O interruptions only) address of DCB.

Each of these macro instructions must be followed by the specify interruption routine (SIR) macro instruction to identify the ICB to the task monitor. SIR attaches a device entry, if necessary, and a request entry to the ITB and establishes the priority of this routine.

Two methods of handling attention interruptions are also provided:

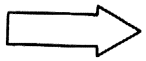
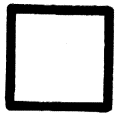
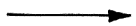
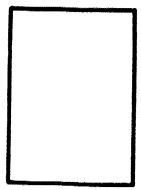
1. SAEC - followed by SIR - used with user attention (USATT) macro instruction
2. Attention entry table (AETD) macro instruction.

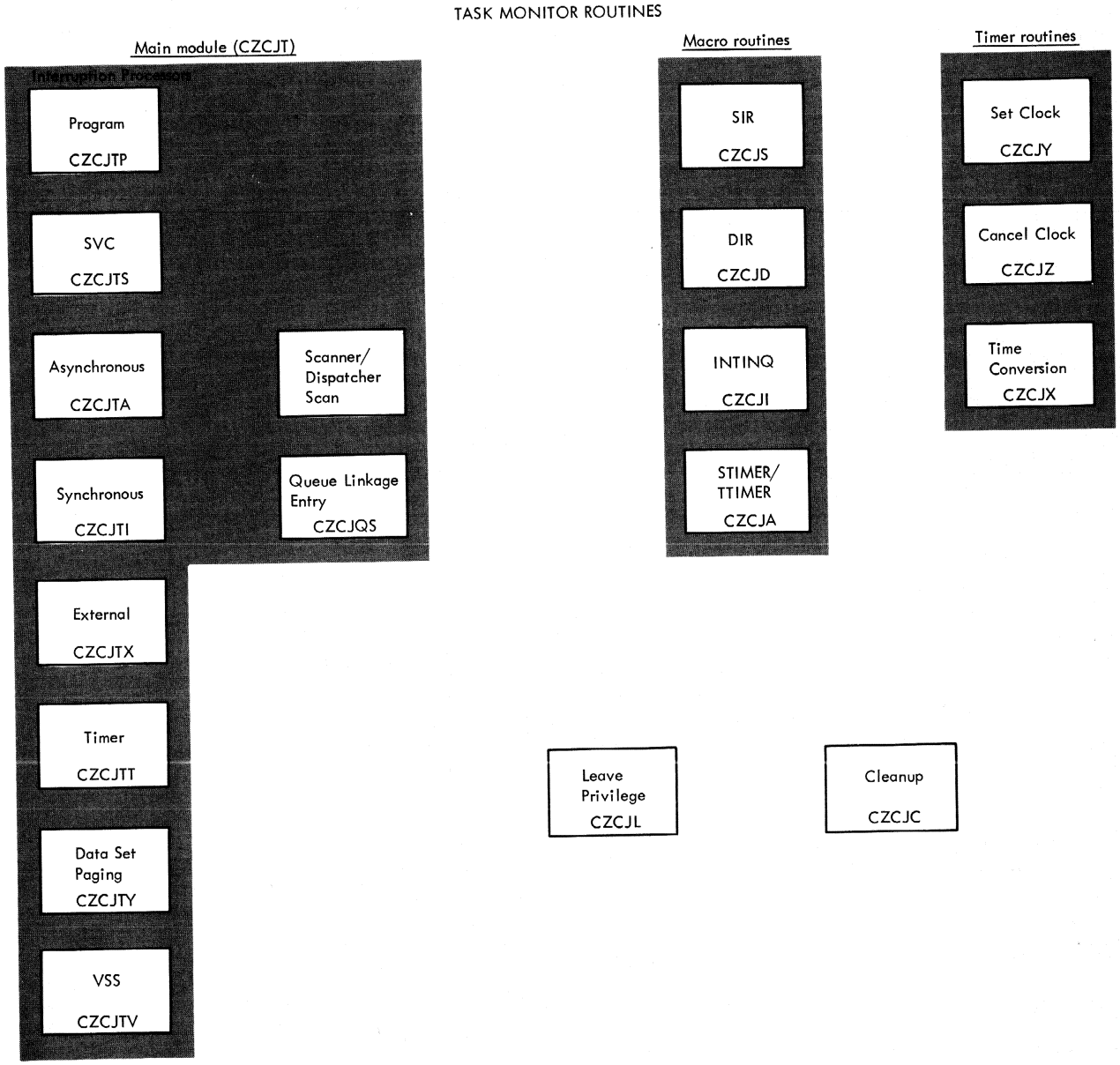
All user-specified interruption handling routines may be disabled by the user. A routine established by SIR may be deleted by the delete interruption routine (DIR) macro instruction. The procedure established by USATT may be deactivated with a clear attention (CLATT) macro instruction.

Method of Operation

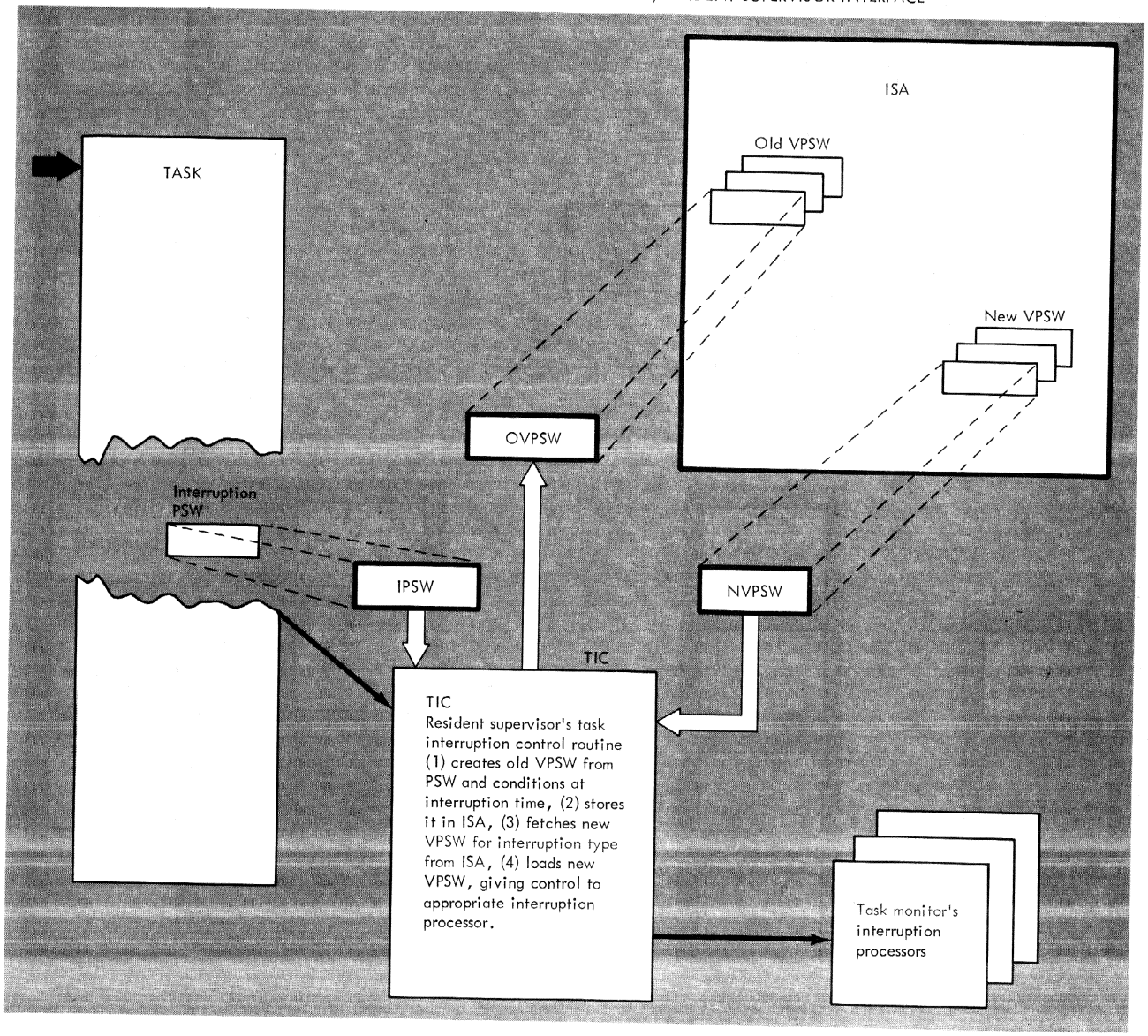
For MO Diagrams

METHOD OF OPERATION KEY

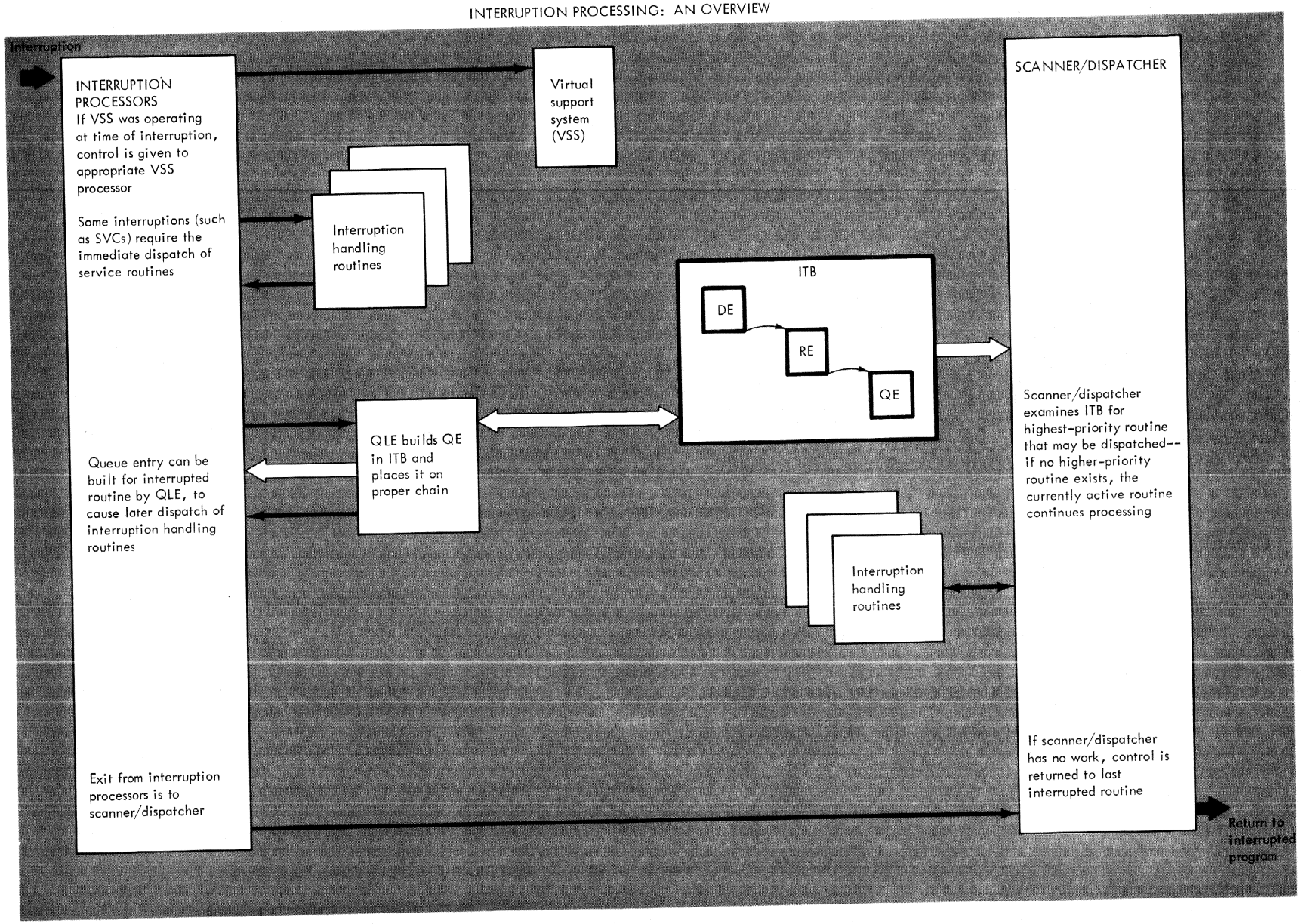
Large black arrow		Entrance to and exit from diagram
Large white arrow		Data flow
Thick line		Flow of control
Thickly outlined box		Control block
Thin line		Pointer
Thinly outlined box		Module


TASK MONITOR ROUTINES
Main module (CZCJT)
Interruption Processors
Program
CZCJTP
SVC
CZCJTS
Asynchronous
CZCJTA
Scanner/
Dispatcher
Scan
Synchronous
CZCJTI
Queue Linkage
Entry
CZCJQS
External
CZCJTX
Timer
CZCJTT
Data Set
Paging
CZCJTY
VSS
CZCJTV
Macro routines
SIR
CZCJS
DIR
CZCJD
INTINQ
CZCJI
STIMER/
TTIMER
CZCJA
Timer routines
Set Clock
CZCJY
Cancel Clock
CZCJZ
Time
Conversion
CZCJX
Leave
Privilege
CZCJL
Cleanup
CZCJC

WHEN AN INTERRUPTION OCCURS: TASK MONITOR/RESIDENT SUPERVISOR INTERFACE



INTERRUPTION PROCESSING: AN OVERVIEW

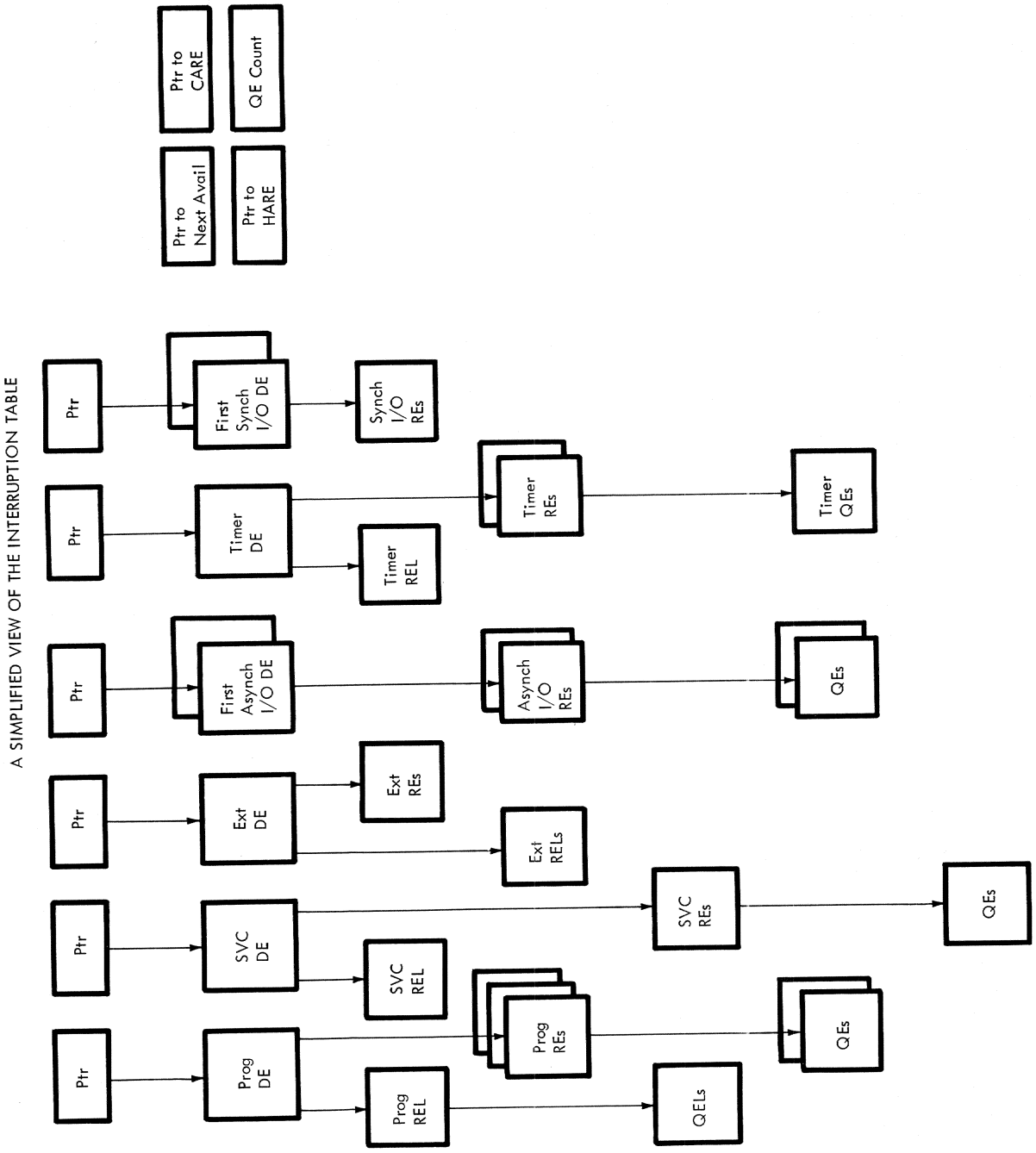


THE INTERRUPTION PROCESSORS

These requests for services require immediate dispatching:

1. Page or segment relocation request.
2. Explicit dynamic linkage (DLINK SVC).
3. Delete (DELET), ENTER, PCS, and restore privilege (RSPRV) SVCs.
4. The resident supervisor's external interruption processor, CZCHC (can be either queued or dispatched immediately).
5. Synchronous I/O handler (these routines check to see if a user-written routine exists; if one does, the request is queued; otherwise the handler is dispatched immediately).
6. An interruption during VSS operation (unless there has beeen a relocation request).
7. A paging error (unless VSS is in operation).

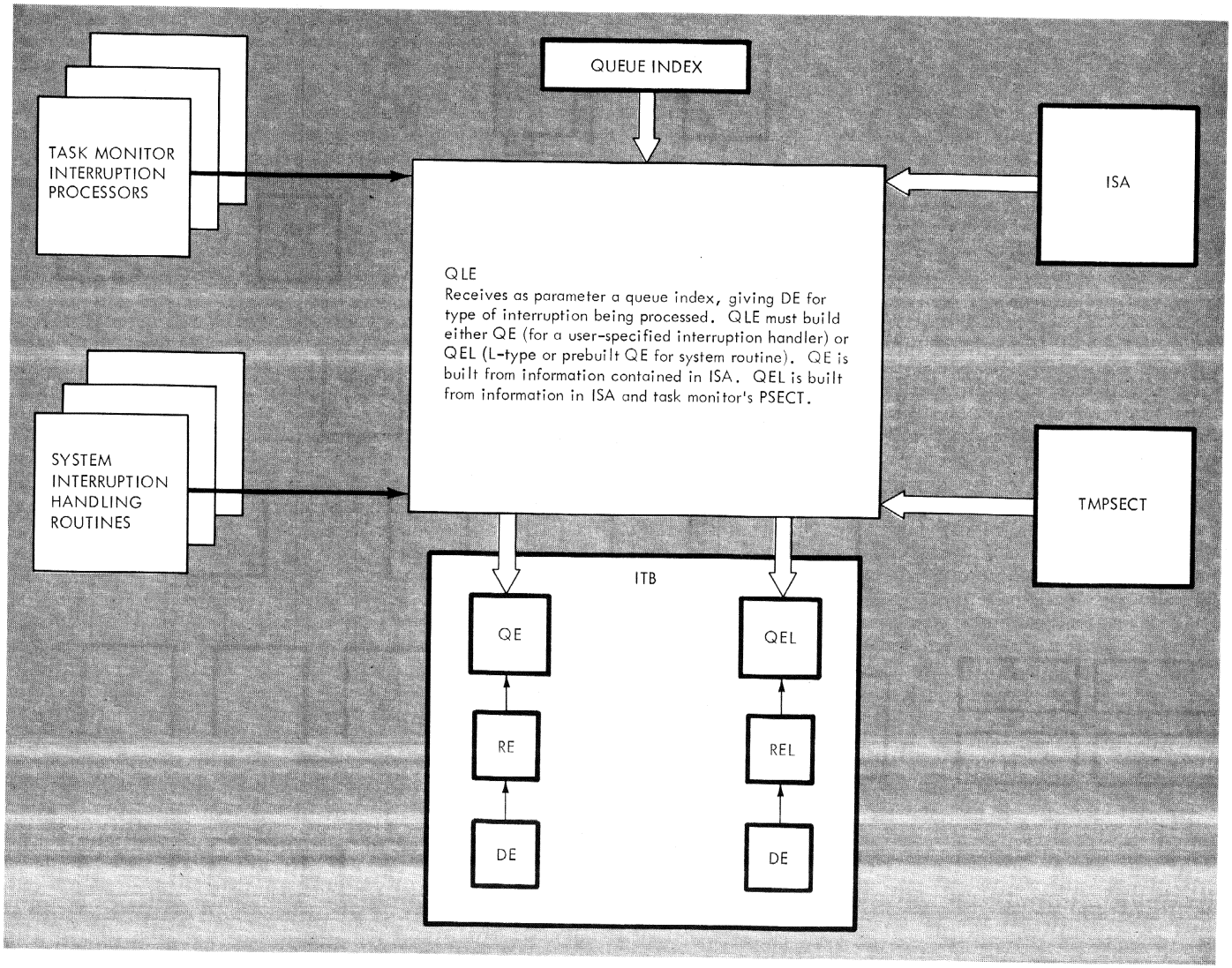
When a routine is dispatched directly from an interruption processor, it returns to that processor. When further servicing is required, a branch is taken to QLE; otherwise exit is to the SCAN entry of the scanner/dispatcher, or to the standard optimal procedure (SOP) subroutine of the scanner/dispatcher if there was no long save in the ISA. On an exit to SCAN, the scanner's save area will contain the old VPSW of the interrupted routine and the contents of the general registers at the time of the interruption.



A SIMPLIFIED VIEW OF THE INTERRUPTION TABLE

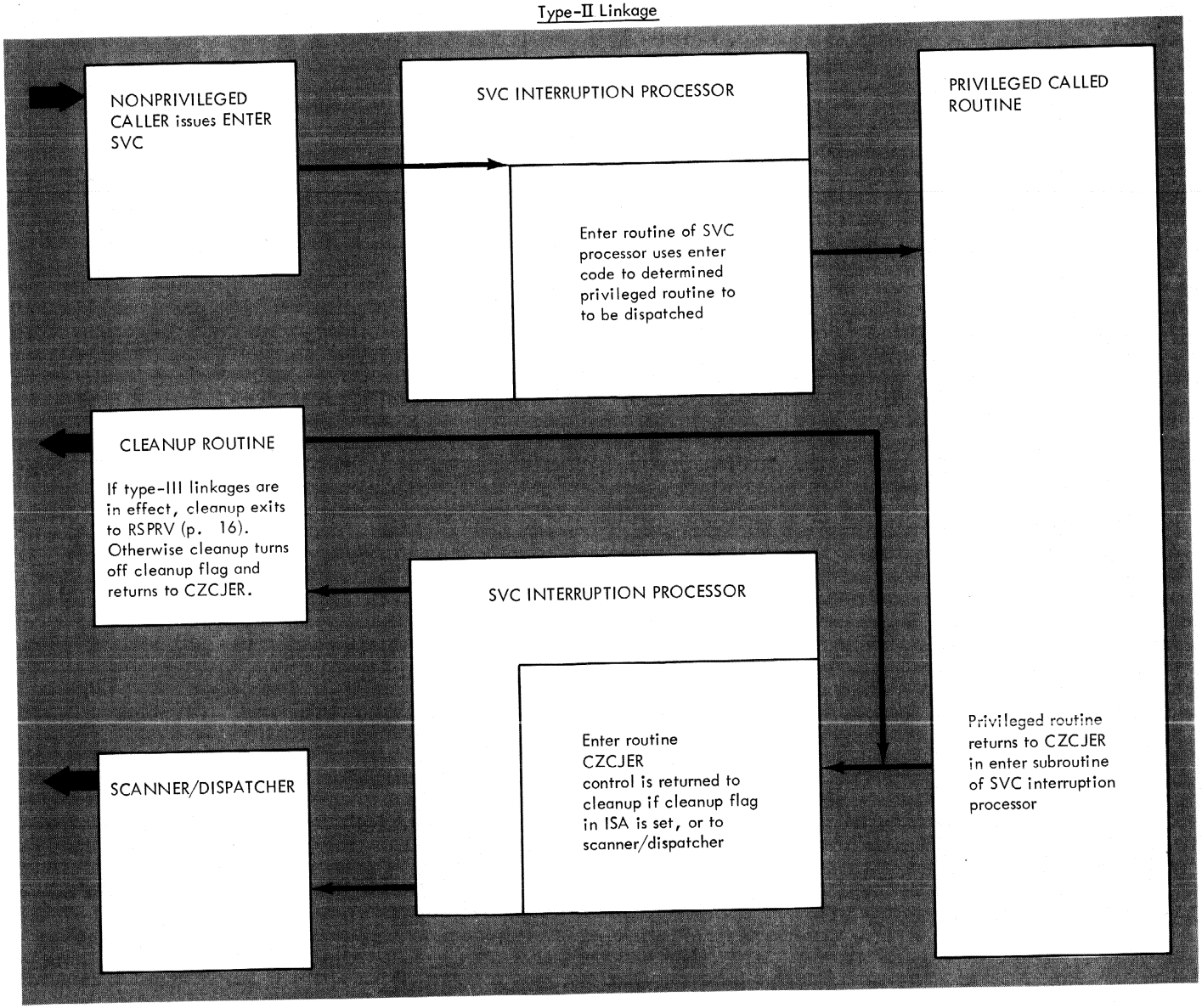
QUEUING PROCEDURE

QLE receives control from the task monitor's interruption processors when the appropriate interruption handler does not require an immediate dispatch, or from a handling routine to dispatch (1) a user-specified routine, (2) another handling routine, or (3) itself at a later time.



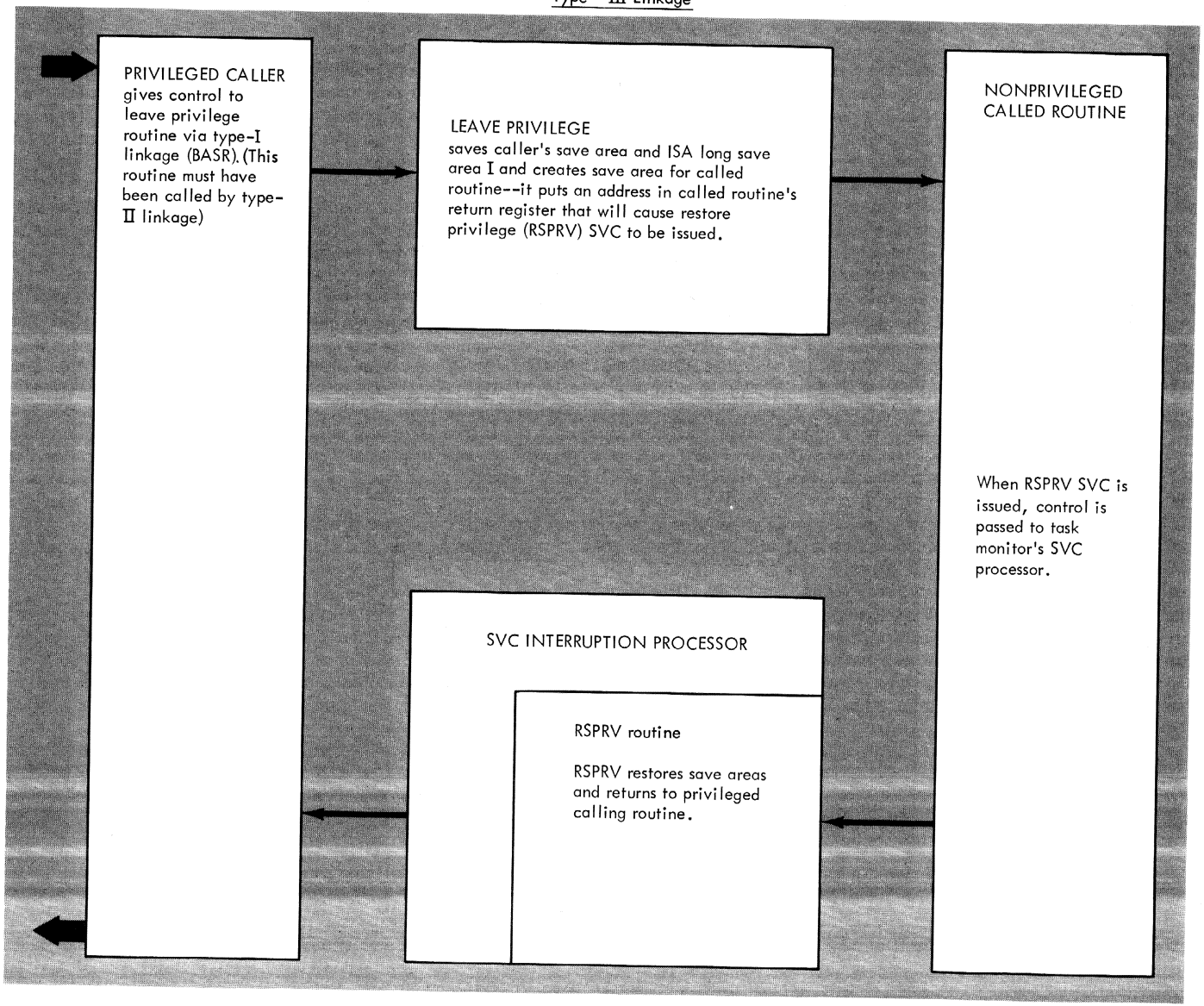
ENTERING, LEAVING, AND RESTORING PRIVILEGE (I)

Type-II Linkage



ENTERING, LEAVING, AND RESTORING PRIVILEGE (II)

Type - III Linkage



THE SCANNER/DISPATCHER

A QE represents a need by a task for the dispatch of an interruption handling routine. The RE and the ICB hold information about that routine. The scanner/dispatcher searches for the highest priority RE that has a QE attached to it. This is the highest active request entry (HARE), and it will be dispatched.

If the scanner/dispatcher receives control before the interruption handling routine that was last dispatched has completed processing, that routine is referred to as the current active request entry (CARE). If there is no RE with a higher priority than CARE, the scanner/dispatcher will return to the CARE. All active REs are considered for dispatch.

Inhibiting Interruptions: A nonprivileged interruption handling routine may prevent the scanner/dispatcher from giving control to any other nonprivileged interruption handling routine until the first has completed its processing. It does this by setting the U1 flag in the ISA and the interruptions disabled flag in the RE. When these flags have been set, no other interruption handler in the ITB will be dispatched unless it is privileged.

The task monitor may still receive interruptions, dispatch those requiring immediate dispatch, and queue those whose dispatch is to be deferred; these must wait for the completion of the interruption that has the U1 bit set. An interruption handling routine could be entered for an interruption while it is still processing a previous interruption, thus losing the first processing. Recursive routines, which call themselves, must protect themselves in this way.

When the interruption handling routine has finished its processing, it returns control either to (1) the privileged return point in the scanner/dispatcher or (2) if it is nonprivileged, to the RSPRV routine in the task monitor's SVC processor, which returns control to the scanner/dispatcher at the nonprivileged return point.

AN EXAMPLE OF TASK MONITOR INTERRUPTION PROCESSING

A programmer, writing PROGRAMA foresees the possibility of program interruption 8 occurring at a point where he does not want PROGRAMA to lose control. He writes a routine that will correct the fixed-point overflow condition that causes program interruption 8. To insure the execution of his routine as an interruption handler, he needs an ICB and an RE.

To provide the ICB, he issues a SPEC macro instruction; using the parameters given, SPEC creates an ICB. The ICB contains the interruption handling routine's entry point, the address of a communications area provided by the programmer, and the type of interruption to be handled. These are generated as declared constants.

PROGRAMA must then issue a SIR macro instruction, which gives control to the SIR routine, which builds the RE and returns to PROGRAMA.

PROGRAMA continues execution. The foreseen possibility occurs; PROGRAMA is interrupted for a fixed-point overflow error. The resident supervisor transfers control to the task monitor's program interruption processor.

The interruption processor then diagnoses the condition as a non-relocation program interruption in the nonprivileged state, and branches to the task monitor's queue linkage entry routine (QLE).

QLE finds the request entry built by SIR for this interruption. It then builds a QE containing the interruption conditions and attaches it to the RE. This indicates that the interruption has occurred and activates the RE. QLE then branches back to the program interruption processor, which branches to SCAN.

Eventually, SCAN issues an LVPSW instruction to transfer control to PROGRAMA's interruption handling routine, which, when it has completed processing, returns control to SCAN, which eventually executes an LVPSW to return control to PROGRAMA.

SIR PROCESSING--LINKAGE CONVENTIONS IN ACTION

In the previous example, PROGRAMA issues a SIR macro instruction to build a request entry. Since PROGRAMA is nonprivileged, the expansion of SIR must generate an ENTER SVC to implement type-II linkage to SIR.

The execution of an ENTER SVC causes the resident supervisor to transfer control to the task monitor's SVC interruption processor.

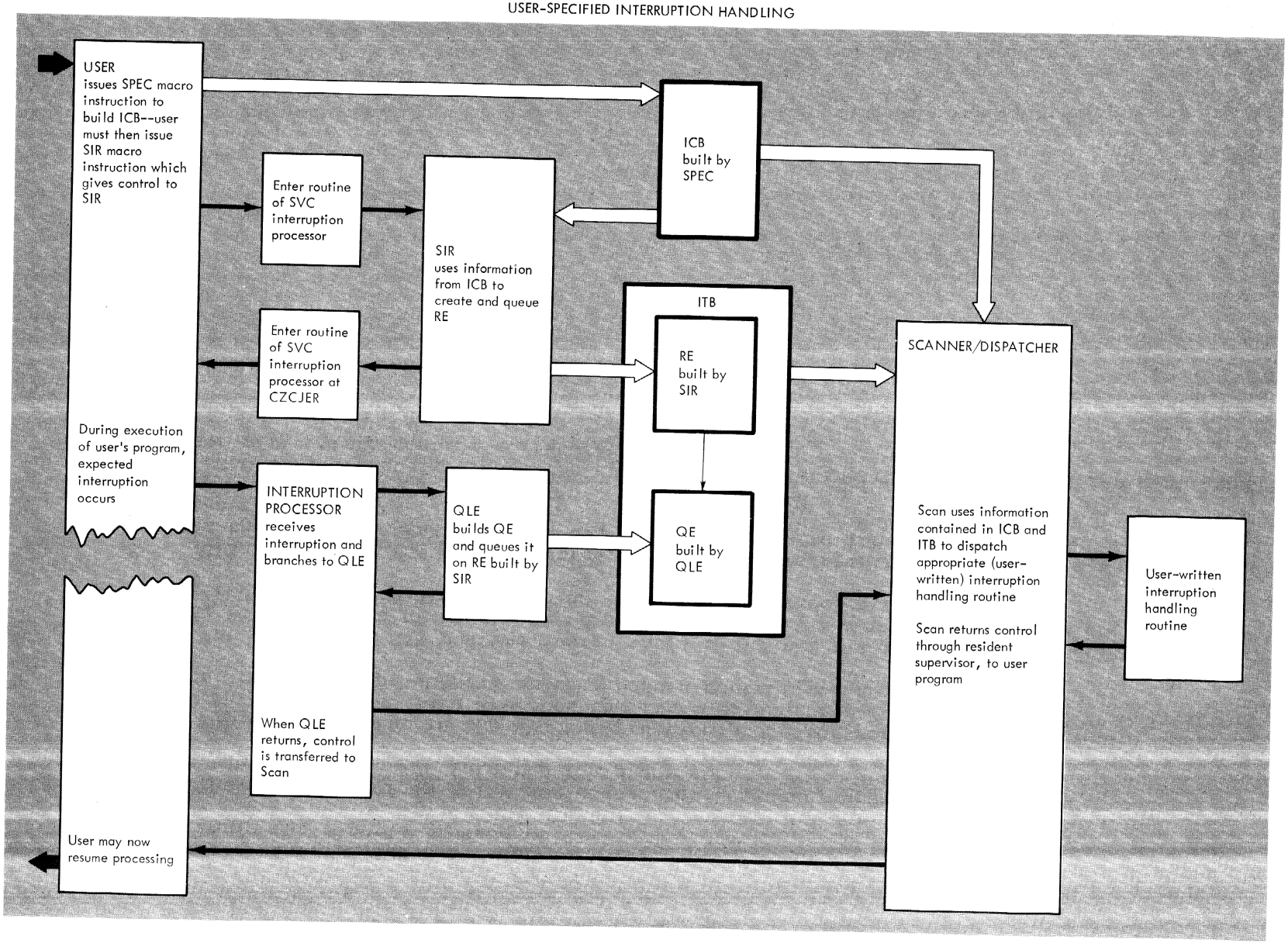
The enter routine in the SVC interruption processor sets up the entry point address, return address, and save area for the linkage, and dispatches via a load VPSW instruction. The resident supervisor then passes control to the SIR routine.

The privileged SIR routine, using the ICB and the priority for the interruption handling routine (both of which are parameters of the SIR macro), builds the RE for program interruption 8, and attaches it to the program device entry.

SIR returns to the enter routine, which restores machine conditions to what they were when the enter routine received control, and exits to the task monitor scanner/dispatcher.

SCAN will eventually issue the appropriate LVPSW instruction, and the resident supervisor will return control to PROGRAMA.

USER-SPECIFIED INTERRUPTION HANDLING



PROGRAM ORGANIZATION

Program Interruption Processor

Entry Point: CZCJTP; the resident supervisor will pass control to this point via an LVPSW instruction.

Operation: This routine first checks to see if VSS was in effect at the time of the interruption. If it was, and if the interruption code is for a page or segment relocation exception, and if the interruption did not occur during explicit linkage, processing continues at TPIDLX, described below. If all three conditions are not met, exit is to CZHNP, the VSS program interruption routine.

When VSS was not in effect at the time of an interruption for page or segment relocation, processing continues at TPIDLX. For any other type of program interruption, this routine checks to see if the interruption came from within the task monitor.

- Second occurrence of expected interruption in task monitor - Exit is to the ABEND routine.

On the first occurrence, if the VSS flag was set before the interruption, exit is to CZHVN, the VSS activation routine. Otherwise the "interruption expected" flag is turned off and control is returned to the point of interruption at TMPIREC.

If this was not a task monitor interruption, and the interrupted task was nonprivileged, QLE is called at TMP01 to create a queue entry for processing of the interruption, and exit is to the standard optimal procedure subroutine of the task monitor's interruption processors.

If the interrupted routine is privileged, and the interruption code is greater than 32, a check is made to see if an asynchronous program interruption is permissible. If not:

- Program interruption in privileged state with code greater than 32 - At TMP258 the SYSER macro instruction is issued (unless the code is 202), and exit is to the ABEND routine.

If so, an asynchronous interruption code table is searched. If it contains a code, processing resumes at TMP01; if not, processing continues at TMP258.

If the interrupted routine is privileged, if the interruption code is less than 32 and if the interruption occurred in PCS, processing resumes at TMP01. If the interruption occurred in the task monitor, the "task monitor interruption expected" flag is turned off, and control is returned to the interrupted routine.

- Program interruption in privileged state with code less than 32 caused by neither the task monitor nor PCS - The SYSER macro instruction is issued, and exit is to ABEND.

Interruption Code 16 or 17--First Relocation Request

At TPIDL, the ISA flag will be referenced to determine whether the interruption occurred in the virtual memory input error recovery routine (VMIER).

- Interruption in VMIER - SYSER is issued and exit is to ABEND.

This routine will then make a long save and pass control to the dynamic loader via an LVPSW instruction. On return, if the second implicit dynamic linkage (or relocation request) flag is on the ISA, control will be given again to the dynamic loader. If the flag is not on, exit will be to SCAN in the scanner/dispatcher. However, if VSS is in effect when

the interruption occurs, exit will be to the VSS activate interruption processor.

Second Relocation Request

When the flag indicating this condition is on in the pushdown save area after the test for VMIER is made, the second implicit dynamic linkage flag will be turned on in the ISA and control will be returned to the task.

Third Relocation Request

Two is the maximum number of relocation requests allowed by the system.

- Third request - The SYSER macro is issued, and exit is to ABEND.

Routines Called: QLE (CZCJQS), dynamic loader (CZCDL4).

Exits: VSS program interruption processor (CZHNPA), VSS activate interruption processor (CZHNVA), SOP, SCAN, ABEND, and return.

SVC Interruption Processor

Entry Point: CZCJTS; used by the resident supervisor.

Operation: When the interruption occurs during VSS operation, exit will be to CZHNV, the VSS activate interruption processor. For an SVC greater than 127 or less than 116, issued by a nonprivileged routine, QLE will be called to queue a linkage entry on the SVC device entry. Exit is then to SOP.

- SVC greater than 127 or less than 116 issued by a privileged routine - SYSER will be issued at TMSEE, which will give control to CEAIS, the system error processor. When CEAIS returns control, the task will be terminated by ABEND.

When the SVC code is between 116 and 127, a branch to the appropriate section of the SVC interruption processor will be taken:

SVC 116 - EXIT

SVC 118 - CLIP (pause)

SVC 119 - CLIC (command): This routine will check for this error:

- Old VPSW privileged - Processing continues at TMSSE.

Otherwise, QLE is called to queue a linkage entry on the command system's user control routine to process the CLIP, CLIC, or EXIT macro instructions; exit is to SOP.

SVC 117 - RAE: The restore and enable SVC causes a long save in the scanner's save area and in the ISA save area 1 or 2, depending on whether the routine is nonprivileged or privileged. It then exits to SCAN.

SVC 120 - RSPRV: Restore privilege will check for the following error:

- Old VPSW privileged - Processing continues at TMSSE.

Usually processing begins at this routine's second entry point, CZCJR2 (forced return). This entry is used by enter and cleanup when the P3 flag is on. This flag indicates a type-III linkage and means that the task is noninterruptable. Restore privilege is used to restore that task to its original status. If the pushdown pointer in the ISA is

zero, this routine branches to QLE. QLE causes linkage to CZAMZ, the command system's user control routine. On return from QLE, exit is to SOP. Otherwise, when type-III linkage is not in effect, this routine exits to the nonprivileged return point in the scanner/dispatcher. If type-III linkage is in effect, FREEMAIN is used to return the storage used for the nonprivileged save area, the P3 flag is turned off, and this routine returns to the privileged caller, via a load VPSW instruction.

SVC 121 - ENTER: The ENTER SVC routine first performs the following test:

- Old VPSW privileged - Processing resumes at TMSSE. If the old VPSW is nonprivileged, the enter code is tested for validity.

- Invalid enter code - QLE is called at TMSE9 to queue a linkage entry on the program interruption device entry. The P3 flag is then tested. If it is on it indicates a type-III linkage with no interruptions permitted, and control is given the RSPRV SVC routine. Otherwise, exit is to SOP.

When the enter code is valid and the P3 flag is set, the privileged force/shutdown flag is inspected. If it is set, a long save is done in the privileged save area and the force flag is set. If not, a long save is done in the ISA save area 1. The linkage parameters are adjusted and moved to the VPSW. When the force/shutdown flag is on it is turned off, and the return parameters are set so control will return to CZCJER. The appropriate routine is then entered via a load VPSW instruction.

If the P3 flag is not on, but the routine is not interruptable anyway, processing is the same as if the P3 flag were on, except that the force/shutdown flag is not tested. The long save is done in the ISA save area 1.

When the routine is interruptable, an interruption code of 60 is placed in the VPSW, and processing resumes at TMSE9.

The routine that is entered must return to CZCJER. At that point, task interruptions are disabled. When the cleanup flag is on, exit is to the cleanup routine (CZCJC). Otherwise, the ISA save area 1 is moved to the scanner save area and exit is to SCAN.

SVC 122 - RTRN: The RETURN SVC routine first checks for the following error:

- Old VPSW privileged - Processing resumes at TMSSE.

The RETURN SVC is placed in the old VPSW, and this routine branches to the task monitor's cleanup routine. On return from cleanup QLE causes a linkage entry to be queued on the command system's user control routine, and exit is to SOP.

SVC 123 - DELETE: This routine sets the explicit dynamic linkage flag in the pushdown save area and dispatches to the dynamic loader's delete routine via a load VPSW. On return from delete the dynamic linkage flag in the ISA and the explicit dynamic linkage flag in the pushdown save area are turned off. Exit is to SCAN.

SVC 125 - PCS CALL: The PCS CALL SVC routine first checks the privilege flag in the ISA.

- Privileged caller - Processing continues at TMSSE.

This routine then dispatches to the program control system via a load VPSW instruction. On return from PCS, the ITI macro instruction is

issued to inhibit task interruptions, the VPSW is set nonprivileged, and exit is to SCAN.

SVC 127 - DLINK (load and call): The DLINK SVC causes a long save in either the task monitor's save area 1 or 2, depending on whether the task is nonprivileged or privileged. This routine then dispatches to the dynamic loader via a load VPSW instruction. On return from the dynamic loader the ITI macro instruction is issued to disable interruptions. If the request was for a call, the V-con for the routine to be called is placed in register 15, and the R-con in the calling program's save area. Exit is to SCAN.

Routines Called: Queue linkage entry (CZCJQS), cleanup (CZCJC), dynamic loader (explicit link) (CZCDL), dynamic loader (delete) (CZCDU).

Exits: SCAN, SOP, scanner/dispatcher (TMRETNP), VSS activate interruption processor (CZHNVB).

Timer Interruption Processor

Entry Point: CZCJTT. It is used by the resident supervisor.

Operation:

Initial Processing -- This routine first tests a flag in the ISA to see if VSS was in operation at the time of the interruption. If it was, exit is to CZHNVD. Otherwise the interruption is logged. The interruption code passed in the VPSW is then tested for validity--code 1 for a real time interruption, and code 2 for a task time interruption.

- Invalid code - the SYSER macro instruction is issued for an invalid timer interruption code, QLE is called to queue a linkage entry on GATWR, and exit is to SOP.

Real Time Interruption Processing -- The value in the timer that caused the interruption should equal the value used in the set timer (SETTR) macro instruction. If so, and the clock was not canceled, and the ignore indicator is not on, QLE queues a linkage entry for the real time interruption, and the clock is set inactive. When there are no more timers to set, exit is to SOP.

- Canceled clock - In this case this routine goes through a loop at TTR6 to determine if there are any other timers to be set. If there are not, exit is to SOP.

- More timers to be set - The SETTR macro is issued and when the time limit has not passed, exit is to SOP. When it has, QLE is again called and the loop at TTR6 is executed.

- Ignore indicator on - SETTR is issued. When the time limit has not passed exit is to SOP; when it has, QLE is called and the loop at TTR6 is executed.

- No active timers - If the value received matches the value used in SETTR, exit is to SOP.

- Timer value received does not match SETTR value - The SYSER macro instruction is issued for an invalid real time interruption. This routine then branches to QLE in order to write the error message via GATWR. Exit is to SOP.

- System timer limit reached - After SETTR is issued if the system timer limit for real-time interruptions has been reached, this routine branches to QLE to queue a linkage entry on the program interruption device entry. Exit is to SOP.

Task Time Interruption -- The timer value received should match the accumulated task time. If so, and if the clock is not cancelled, the ignore indicator is not on, and the interrupted clock is not task clock 15, this routine branches to QLE for the timer interruption and the clock is set inactive. If there are no more timers to set, exit is to SOP.

- Clock was canceled - When there are no more timers to set and the clock is not task clock 15, exit is to SOP. If the clock is task clock 15, exit is to the forced return point in the SVC interruption processor, CZCJR2. This is the second entry point in the restore privilege routine (SVC 120).

- Ignore indicator on - When the ignore indicator is on, processing resumes at TTT7A. The XTRTM macro instruction is issued to compute the task's accumulated CPU time. If the task has one millisecond or more to run, the SETTU macro instruction is used to update the user's timer. When the clock is task clock 15, exit is to CZCJR2; otherwise, exit is to SOP. If the task does not have at least one millisecond to run, QLE queues a linkage entry for the timer interruption, the clock is set inactive, and if there are no more timers to set, exit is to SOP, or for clock 15, to CZCJR2.

- Task clock 15 - In this case, clock 15 is canceled. If the asynchronous timer indicator is on in the prefixed storage area, it is changed to indicate clock 15 exit. If there are no more timers to be set, exit is to CZCJR2.

- More timers to be set - A loop is entered at TTT6 to determine which clocks need to be set. Processing then begins at TT7A as described above.

- Timer value received does not match accumulated task time - The SYSER macro instruction is issued for an invalid task time interruption. Exit is to ABEND.

Routines Called: QLE (CZJQS).

Exits: SOP, SCAN, VSS activate interruption routine (CZHNVD), RSPRV SVC (SVC interruption processor), ABEND.

External Interruption Processor

Entry Point: CZCJTX. It is used by the resident supervisor. When the task initiation flag in the ISA is not on, the task monitor's asynchronous interruption processor also branches to this point.

Operation: When VSS is in operation at the time of the interruption, this routine exits to the VSS external interruption processor, (CZHNE). If the task is logged on or was in the process of logging on when interrupted, a message code is placed in the old VPSW, and this routine links to the command system external interruption processor (CZAHC). If the return code from that module is not zero, exit is to SOP. Otherwise, QLE is used to queue a linkage entry on the external interruption device entry before that exit is taken.

When the task is not logged on or in the process of logging on, this routine processes the interruption as a logon. The logon in progress flag in the ISA is turned on, interruptions are permitted, and the VPSW is set nonprivileged. If the interruption was initially received as an asynchronous interruption, QLE is called to queue a linkage entry to the initial attention interruption processor, CZAHB, and exit is to SOP. Otherwise, processing continues as if the task were already logged on.

Routines Called: Queue linkage entry (CZCJQS), command system external interruption processor (CZAHC).

Exits: VSS external interruption processor (CZHNE), SOP.

Synchronous I/O Interruption Processor

Entry Point: CZCJTI. It is used by the resident supervisor.

Operation: If VSS was in operation when the interruption occurred, this routine exits to CZHSB, the VSS synchronous I/O interruption processor. Otherwise, the interruption is logged, and QLE is called to queue a linkage entry on the synchronous I/O interruption device. This module then branches to the appropriate I/O posting routine. On return from the posting routine, exit is to SOP.

Routines Called: Queue linkage entry (CZCJQS), I/O posting routine.

Exits: VSS I/O interruption processor (CZHSB), SOP.

Asynchronous I/O Interruption Processor

Entry Point: CZCJTA. It is used by the resident supervisor.

Operation: When VSS is operating at the time of the interruption this routine exits to CZHNV (VSS activate interruption processor). Otherwise, the interruption is logged and the task initiation flag in the ISA is tested. If it is on, this routine branches to QLE to queue a linkage entry on the asynchronous interruption device entry, and exit is to SOP.

When the task initiation flag is off, and logon is not in progress, the asynchronous logon flag is turned on and this routine exits to the task monitor's external interruption processor, CZCJTX.

When logon is in progress, the interruption is ignored and this routine returns control to the calling program via an LVPSW instruction.

Routines Called: Queue linkage entry (CZCJQS).

Exits: VSS activate interruption processor (CZHNVC), SOP, task monitor external interruption processor (CZCJTX), return.

Data Set Paging Interruption Processor

Entry Point: CZCJTY. It is used by the resident supervisor.

Operation: If VSS was in effect when the interruption occurred, this routine exits to CZHNV, the VSS activate interruption processor. Otherwise, the interruption is logged, and a flag in the ISA is tested to see if the interruption occurred during error recovery for a previous data set paging error. If so, the SYSER macro instruction is issued and exit is to ABEND. If not, this routine branches to the virtual memory input error recovery routine (VMIER). On return from VMIER, exit is a return to the calling program via an LVPSW instruction.

Routines Called: Virtual memory input error recovery (CZCEI1).

Exits: VSS activate interruption processor (CZHNV), return, ABEND.

Virtual Support System Interruption Processor

Entry Point: CZCJTV. It is used by the resident supervisor.

Operation: This routine first logs the interruption, and then tests a flag in the ISA to determine if the old VPSW was masked, indicating that interruptions were not permitted. When the flag is set and the implicit dynamic linkage flag is on, or if the interruption occurred in the task monitor's front end, the VSS interruption flag is turned on in the push-down save area and the routine returns via a load VPSW instruction. Otherwise, this routine exits to the VSS activate interruption processor, CZHNV.

Routines Called: None.

Exits: VSS activate interruption processor (CZHNVA), return.

Standard Optimal Procedure Subroutine (SOP)

Operation: When the VPSW is privileged, the force/shutdown flag is tested. If it is not on, the registers are restored at SOPSR01 and this routine returns to the last interrupted routine via a load VPSW instruction. If the force/shutdown flag is on, it is turned off. If task initiation is not over, processing continues at SOPSR01. Otherwise the dynamic loader and P3 flags are turned off, and ABEND is called via the task monitor's ENTER SVC processor.

When the VPSW is nonprivileged and this routine was given control by a task monitor interruption processor, the force/shutdown flag is again tested. If it is on, it is turned off. If it is off, or if the P3 flag is off, this routine exits to SCAN. Otherwise, the P3 flag and the enter flag are turned off and exit is to SCAN.

Routines Called: None.

Exits: ABEND, SCAN, return.

Queue Linkage Entry Routine

Entry Point: CZCJQS. QLE is entered via type-I linkage.

Operation: Upon entrance to QLE, task interruptions are inhibited. The queue index is tested. When it is less than 6, this routine branches to one of the standard queue processors. When it is between 6 and 10, the QLE PREBUILT subroutine is used to chain a QEL to a prebuilt REL, and exit is to the COM subroutine.

- Queue index greater than 10 - The SYSER macro instruction is issued and this routine branches to the COM subroutine.

SYNCHQE: This section of QLE, entered to process synchronous I/O queue entries, first branches to the SDATCHCK subroutine to convert the symbolic device address into the correct SDAT entry. When that is done, this routine checks the existence of a synchronous I/O device entry matching the SDAT.

- No synchronous DE - Exit is by means of QLE RET subroutine.

When there is a synchronous DE, a check is made for a nonzero RE that has not been deleted. When one is found, the CKICB subroutine is called to determine if the ICB is addressable. If it is not, the test for a valid RE continues. When the RE is valid, a check is made to see if its SDAT matches the SDAT of the interrupted routine. If there is a match, the ITGETMN subroutine is called to make sure the ITB can accommodate another entry, and when ITGETMN returns, the BLDLEQE subroutine is called to build the common areas of a QE. The status and sense bytes peculiar to this QE are moved in and exit is to COM.

- No active RE matching interruption RE - Exit is to the RET subroutine.

PROGQE: At PRELOP, if there is a program interruption RE whose ICB is addressable and contains an interruption code which matches the interruption code in the ISA, and it has not been deleted, and there is a V-con for the interruption routine and a COMAREA, this routine checks to see if the interruption occurred in PCS. If it did not, and there is no CARE, or the priority of this RE is greater than that of the CARE, and the P3 flag is off, then the ITGETMN subroutine is used to guarantee that the ITB has enough main storage. The QE is built and added to the chain of this RE. The U1 flag in the ISA is turned off, and exit is to the COM subroutine.

- No matching REs - The P3 flag, which indicates a type-III linkage with no interruptions allowed, is tested at PDIAG. When it is on, the forced return switch is turned on before a branch to the DIAGNO section of the PREBUILT subroutine is taken. This code builds a QEL for DIAGNO (CZAHA1). When the forced return switch has been turned on, an error code of X'66' is placed in the VPSW and it is moved to the QEL. The U1 flag is then turned off, and exit is to the COM subroutine.

- The ICB and ISA interruption codes don't match -

- RE was deleted -

- No V-con for interrupted routine -

- No COMAREA - In these cases the pointer to the RE is obtained and the next RE is tested against these conditions at PRELOP.

- The interruption occurred in PCS - The P3 flag is not tested before the branch to ITGETMN at BLDQE.

- The CARE has priority higher than this RE - Processing resumes at PDIAG.

- The P3 flag is on at P3CHK - Processing is the same as that under the same condition at PDIAG.

SVCQE: This routine searches the request entries on the SVC device entry until it finds one which has an addressable ICB with an SVC code matching the code in the ISA. The P3 flag is then tested. If it is not on the ITGETMN subroutine is called to make sure there is enough space for another ITB entry, and exit is to the BLDNQE subroutine.

- P3 flag on - Processing continues at TOFRT2. The forced return switch is turned on and the DIAGNO section of the PREBUILT subroutine receives control to build a QEL for DIAGNO (CZCHA). The forced return flag is then tested. If it is on, a X'67' error code is placed in the VPSW. If it is off, a X'62' error code is placed in the VPSW. The VPSW is then moved to the QE and exit is to the COM subroutine.

- No matching RE found - The P3 flag is tested at PDIAGS. Processing continues at TOFRT2. If the P3 flag is not on, the forced return flag is not turned on.

EXTQE: If an RE can be found that has an addressable ICB and the RE contains a message number that matches the message number in the ICB and the RE was not deleted, this routine branches to the ITGETMN subroutine. On return from ITGETMN, control is given to BLDLEQE to build the QE. This routine then branches to the GETMAIN (byte) routine (CZCH2), at GETMSGMN in order to get main storage for the MCB and associated message

text. A pointer to the MCB is then placed in the QE, and exit is to COM.

- No suitable RE found - The DIAGNO section of the PREBUILT subroutine receives control in order to build a QEL for XMIS (CZAHD3), the command system external interruption message routine. Processing continues at GETMSGMN.

ASYNQE: This routine branches to the SDATCHK subroutine to convert the symbolic device address into the proper format for a SDAT entry. This routine then branches to ITGETMN, and on return, builds a QE for this interruption.

- Interruption from SYSIN - This routine checks whether the system or a user will handle this interruption. If the system will receive the interruption the attention counter is incremented. Exit is to the COM subroutine.

If there is any asynchronous device entry matching this SDAT, and an RE queued on the DE whose ICB is addressable and contains an interruption code that matches the code placed in the QE, exit is to COM.

- No matching DE -
- No matching RE -
- Request entry deleted - Under any of these conditions the QE is cleared and returned to the available chain. Exit is to the RET subroutine.
- ICB not addressable - The search for an acceptable RE continues.
- No matching SDA and SDAT - The SYSER macro is issued and exit is to ABEND.

TIMQE: When there is a timer RE whose ICB is addressable and contains the timer type and number matching those of the interrupted timer, and if that RE has not been deleted, the ITGETMN subroutine is used to make sure the ITB has enough main storage. The timer type is then placed in the QE, and exit is to the BLDNQE subroutine.

- No matching RE - The SYSER macro is issued, and exit is to ABEND.

QLE Subroutines

PREBUILT: The message control block contains data used in building a QEL. This subroutine first gets a pointer to the MCB for the necessary data. At DIAGNO, it uses ITGETMN and BLDLEQE to insure that there is storage for the QE and to initialize it. If this is a logon linkage entry, the address of the SDAT pointer is placed in the QE, the linkage to the initial attention interruption processor (CZAHB1) is set up, and exit is to COM. If it is not a logon linkage entry the QE is finished, and this subroutine returns to its caller.

SDATCHCK: When the symbolic device address is zero, this subroutine returns control to the calling portion of QLE. Otherwise it multiplies the address by 64 and adds it to the starting address of the SDAT to get the address of the correct SDAT entry. If the device is detached or partitioned, the entry is invalid and the SYSER macro is issued. Exit is to ABEND. Otherwise, exit is a return to the calling section of QLE.

ITGETMN: This subroutine first checks to see that all entries in the ITB have been filled. If they have not, the registers are restored and control is returned to the branch point. The GETMAIN (page) routine, CZCG2, then receives control to get the required pages of main storage.

The pointers to the available pages are adjusted and control is returned to the branch point.

BLDNQE: This subroutine places the interruption code, VPSW pointer, and device type in the QE and returns to the branch point.

BLDLEQE: This subroutine places the LE code and interruption code in the QEL, and then tests to see if the QEL is for a prebuilt routine. If it is, the device type is added to the QEL before the return to the branch point.

COM: If the RE is active, the QE is queued on it at QUEUE and a test is made for a forced return. If the forced return flag is off, this routine exits to the RET subroutine at RESMSK. If the forced return flag is on, control is returned to the forced return point in the task monitor's SVC processor, CZCJR2.

- RE inactive - The active code is set in the RE. When there are no active REs on the chain, this RE becomes the highest active (HARE). Processing resumes at QUEUE. When there are other REs and this RE is lower in priority than the HARE, it is merely placed on the RE chain. If it is higher in priority than the HARE, it becomes the HARE. In either case, processing resumes at QUEUE.

RET: This subroutine first zeroes the return code. At RESMSK, if a QE was queued, the interruption was an asynchronous attention, and the P3 flag is on, this subroutine branches to the task monitor's set clock routine to set clock 15 at 100 milliseconds to allow attention interruptions. Then at RESMSKG, interruptions are allowed if they were inhibited before, the registers are restored, and this routine restores control to the routine that called QLE.

- No QEs queued -
- Interruption was not an asynchronous attention -
- P3 flag was not on - under any of these conditions, set clock is not called and processing continues at RESMSKG.

CKICB: This subroutine performs a null operation (one that has no net result) on both the first and last bytes of the ICB. It then checks to see if a program interruption 5 (addressing) or 6 (specification) was received. If not, the ICB is addressable and control is returned to the branch point within QLE.

- ICB not addressable - The RE is deleted. If there are QEs on this RE the task is aborted; otherwise, control is returned to the branch point with a return code of 4.

Routines Called: GETMAIN (page) (CZCG2), set clock (CZCJY).

Exits: ABEND, SVC interruption processor (CZCJR2), return.

Scanner/Dispatcher

Entry Point: SCAN. This entry is used only by other task monitor routines. There are also two return entry points: TMRETNP for nonprivileged routines, and SYSPRT for privileged routines.

Operation: The scanner/dispatcher receives control to dispatch the highest priority active request entry -- HARE. It is given a pointer to the task monitor's PSECT, from which it obtains pointers to the other control blocks it uses. The scanner/dispatcher first checks for several conditions which inhibit its processing:

- QE count is zero - No RE is active, so control is returned to the interrupted routine.

- HARE=CARE - Highest active routine is currently processing, so control is returned to the interrupted routine.

- No QE on HARE corresponds to the ICB - The QEs are removed and the RE is deactivated. The next highest RE becomes HARE and processing starts again at SCAN.

- The COMAREA pointer (or the entry point 1 pointer for a nonsystem routine) is missing - Control is returned to the last interrupted routine.

- The P3 or U1 flag is on - The interrupted routine was processing under type-III linkage, and should not be interrupted, so control is returned to it.

- The interrupted routine was privileged - Control is returned to it.

Scanner/dispatcher processing then begins. The scanner save area in the TMPSECT is moved to either the privileged pushdown save area (PDSA) for a privileged routine, or to the nonprivileged PDSA for a nonprivileged routine. HARE becomes CARE, and the old CARE is rechained by priority on the active RE queue, and must wait until it is again the highest active request entry to complete its processing. Processing for an L-type QE (one that requests the services of a system interruption handling routine) continues at SCANLE. If the QE is for a user-written routine, the existence of a user-written communications area (COMAREA) is checked for at SCANQE.

- No COMAREA for privileged CARE - A SYSER macro instruction is issued, and exit is to ABEND.

- No COMAREA for nonprivileged CARE - At JTCOM2, QLE is called for DIAGNO, to inform the user of his error. At DEQ the QE is deleted, and the currently-processing interruption handling routine is dispatched.

Then, at JTCOM, if the CARE is nonprivileged, the CKCLS macro instruction is issued to make sure that the COMAREA is also nonprivileged.

- Privileged COMAREA or ICB EP1=0 for nonprivileged CARE - Processing continues as above at JTCOM2.

When the COMAREA and CARE match and the proper ICB entry point pointer is present, a check is made at SCANQE1 to see if the QE is external. If so, the message in the MCB must be moved to a user-specified area.

- User-specified message area has incorrect privilege - The message is moved into an area provided by the task monitor.

Information about the message is then placed in the COMAREA.

At DEQ, the QE is removed from the QE chain. If CARE is privileged, the appropriate flags are set and processing resumes at CHKSW0. Otherwise, if interruptions are inhibited, the U1 flag is turned on. Then the VPSW is flagged nonprivileged and one is added to the counter of dispatched nonprivileged routines. If the ICB entry point 2 pointer is not zero, processing resumes at EP2NZ.

At CHKSW0, the currently processing interruption handling routine is dispatched.

At EP2NZ, if the pushdown save area pointer is zero, the currently processing interruption handling routine is dispatched.

At SCANLE, if the requested system interruption handling routine is DIAGNO, special DIAGNO information is placed in the QE. Other system information is placed in the QE and processing continues at DEQ.

When an interruption handling routine completes its processing, it returns control to the scanner/dispatcher at either SYSPRT or TMRETNP, depending upon whether it was privileged or nonprivileged.

At SYSPRT, task interruptions are inhibited, and the ISA save area 1 is moved into the privileged pushdown save area. Processing continues at DELCHK.

At TMRETNP, the pushdown save area is moved into the ISA save area 1. If the counter of nonprivileged interrupted routines dispatched is not greater than zero, processing continues at DELCHK. Otherwise, one is subtracted from the counter, and the addressability of the ICB is checked.

- User deleted the ICB for a returning interruption handling routine - Exit is to ABEND.

If the ICB entry point 2 pointer is zero, processing continues at DELCHK. If not, it means that the task has issued an INTINQ macro instruction and relinquished control at this point. Should it be interrupted again, it will resume processing here. Information about this task is moved into the pushdown save area. Otherwise, processing information is moved into the pushdown save area.

Then, at DELCHK, the following series of checks is performed:

- RE not deleted - Processing continues at QECHK.
- ICB not addressable - Exit is to ABEND.
- DE code invalid - At INVTYP, exit is to ABEND. At QECHK, the checks continue.
- HARE=CARE - Processing continues at CACHK.
- No QEs on CARE - Next active RE becomes CARE and processing continues at QECHK.

If there are QEs on CARE, the CARE PDSA is moved to either the privileged or nonprivileged PDSA, depending upon the privilege of CARE. Then, if there are QEs on CARE, processing resumes at SCANIN. Otherwise, processing resumes at SCANR.

At CACHK, if there are no QEs on CARE, processing continues at LOWER. Otherwise, if the QE is L-type, processing resumes at SCANLE. If the QE is for a user-written routine, the CHKMCH subroutine is called to find a QE-ICB match.

- No QE-ICB match - Processing continues at CACHK.
- COMAREA pointer = 0 -
- ICB EP1 pointer = 0 - Processing continues at GNACT.

If the COMAREA pointer and ICB entry point 1 pointer contain nonzero values, processing continues at SCANQE.

At LOWER, certain fields in the RE are zeroed. If CARE is not HARE, CARE is removed from the active RE chain. A search is then made for CARE's DE. If it cannot be found, the next highest priority active RE becomes CARE. When the CARE DE is found, it is flagged as active, and the CARE pointer in the interruption table is zeroed.

Then, at GNACT, if there are more active REs, the next highest-priority RE becomes CARE and the following checks are performed:

- RE not an insert -
- RE is L-type - Processing continues at PUSHUP.
- No QEs on this RE - Processing continues at LOWER.

If there are no more active REs, processing continues at PUSHUP.

If none of the bulleted conditions are true, the CHKMCH subroutine is called to find a QE-ICB match for this RE.

- No QE-ICB match found - If there are no more QEs, processing continues at LOWER; if there are, CHKMCH is called as above.

At PUSHUP, the pushdown save area is moved to the scanner's save area.

- Last RE dispatched -
- RE not an insert - Control is returned to the interrupted routine.

Otherwise processing continues with the fourth error check after entrance to the scanner/dispatcher.

At SCANR, the following checks are performed:

- CARE or first nonprivileged RE is on miscellaneous queue - Processing continues at SETRENA.
- Invalid DE code - Exit is to ABEND.

The CARE DE is found and flagged active. At SETRENA, if HARE is CARE, the next highest priority active RE becomes HARE, and CARE is deactivated.

At SCANIN, HARE becomes CARE. If the first QE on the new CARE is for a system routine, processing resumes at SCANLE. Otherwise, CHKMCH is called to find a QE-ICB match.

- No QE-ICB match found - Processing continues at LOWER. When the match is found, processing resumes at SCANQE.

CHKMCH: This subroutine first makes sure that the DE type is valid.

- Invalid DE type - At INVTYP, exit is to ABEND.
- No QEs on CARE - An error return code of 4 is set and control is returned to the branch point within the scanner/dispatcher.

A branch is then taken according to DE type. Various checks are made depending on the DE type.

Program: The interruption codes in the QE and the ICB must match.

Timer: The task or real codes must match, and the timer codes must also match.

SVC: The SVC codes must match.

External: The message codes must match.

Asynchronous: The attention codes must match.

Synchronous: A match is assumed.

If the match has been found, a successful return code is placed in register 15 and control is returned to the branch point.

If no match is found, any QEs tested are zeroed and their space is flagged available. The error return code of 4 is placed in register 15 and control is returned to the branch point.

Scanner/Dispatcher Routines Called: None.

Exits: Dispatch to the appropriate interruption handling routine, or return to the last interrupted routine.

SIR--Specify Interruption Routine

Entry Point: CZCJSA. The standard type-I or type-II linkage and a parameter list are generated during expansion of the SIR macro instruction.

Operation: This routine first disables interruptions if they were enabled when it gained control. It then checks for these errors:

- Parameter list is invalid -
- A nonprivileged user wishes to build a privileged ICB -
- A nonprivileged user wishes to build an ICB with a priority greater than 127 - In these cases the ICB is set invalid and an error return code of 12 is placed in register 15 at ICBRTN. The PTI macro is then issued at SIRTRN to enable interruptions if they were enabled at start of execution, and this routine returns to the calling program.
- There is no ICB parameter - Processing continues at SIRTRN.

The pointer to the ICB is obtained from the parameter list supplied in the macro instruction. More error checks are then performed:

- The ICB is not on a fullword boundary -
- The user is not permitted access to this ICB. (This is determined by issuing the CKCLS macro) - In these cases, processing continues at ICBRTN.

The COMAREA is then checked for valid construction:

- User is not permitted access to this COMAREA -
- COMAREA is not on word boundary -
- ICB type not legal - In each of these cases, processing continues at ICBRTN.

If the ICB is for a synchronous or asynchronous I/O interruption, the DCB is checked:

- DCB is not in virtual memory -

- DCB is invalid -
- DCB is not opened - In each of these cases, processing continues at SIRTRN.

If the DCB is valid, this routine checks for a device entry. If none exists, it builds one. It then checks for an RE queued on the ICB. If there is one, and the ICB is for a timer, the cancel clock routine (CZCJZ) is called to cancel the clock.

- Error in cancel clock processing - Processing continues at SIRTRN.

Set clock is then called at RESETIME to reset the clock.

- Error in set clock processing - Processing continues at SIRTRN.

If the reset flag is set, it is turned off and this routine checks for more parameters, processing them at SIRLOOP if there are any and returning at SIRTRN if there are not. If the reset flag is not on, a new RE is built and inserted in the RE chain before the check for more parameters is made.

When there is no RE, or if it was deleted, this routine checks for a timer DE. If there is one, processing continues at RESETIME. If there is no timer DE, a new RE is built and inserted in the RE chain. A check for more parameters is made. If there are none, exit is at SIRTRN. If there are more, their processing begins at SIRLOOP.

Routines Called: Set clock (CZCJY), cancel clock (CZCJZ).

Exits: Return to the calling routine with the appropriate return code.

DIR--Delete Interruption Routine

Entry Point: CZCJDA. The standard type-I or type-II linkage and a parameter list are generated during expansion of the DIR macro instruction.

Operation: This routine first issues the ITI macro instruction to disable interruptions if they were permitted on entry. The first parameter from the list generated by the macro is obtained at DIRLOOP. A series of error checks is then performed:

- Invalid parameter list length -
- No ICB pointer -
- Nonprivileged routine attempting to delete privileged ICB - In these cases, processing continues at DIRTRN. Interruptions are permitted if they were permitted on entrance to this module. Register 1 is loaded with the invalid ICB pointer, if there was one, and this routine returns control to the calling routine with an appropriate return code in register 15.

When there is an asynchronous DE chain and a device entry match for the DCB in question, the DQRE subroutine is used to dequeue all REs and QEs that match this DCB. If there is no asynchronous chain or no match for the DCB, this processing is repeated for the synchronous chain. When there is no synchronous DE match for the DCB, this routine returns at DIRTRN.

A series of error checks is then made:

- Invalid ICB -

- No QEs on ICB chain -
- DCB not valid -
- No DE matching DCB -
- RE has been deleted - In any of these cases, processing continues at DIRTRN.

If the ICB is for a clock, the task monitor's cancel clock routine (CZCJZ) is called to cancel that clock.

- Error in cancel clock processing - Processing continues at DIRTRN.

If the RE is not active, it is dequeued, together with all the QEs on that RE. A test is then made for more parameters. If there are more, their processing begins at DIRLOOP. If not, this routine returns control at DIRTRN.

If the RE is active, it is left intact, but all its QEs are dequeued. This routine then exits at DIRTRN.

Routines Called: Cancel clock (CZCJZ).

Exits: Return to the calling routine with the appropriate return code.

INTINQ--Interrupt Inquiry

Entry Point: CZCJIA. The standard type-I or type-II linkage and a parameter list are created during expansion of the INTINQ macro instruction.

Operation: This routine first disables interruptions if they were permitted on entrance. It then performs the following series of error checks:

- ICB pointer is zero -
- A nonprivileged user is checking a privileged ICB -
- DE type is invalid - In these cases, processing continues at RETRN08. A return code of 8 is placed in register 15, task interruptions are permitted if they were permitted when this routine received control, and control is returned to the calling routine.

This routine then branches on DE type to check the validity of the interruption type. Basic processing is the same for each interruption type, but various checks must be made first as follows (processing continues at the label given):

APROGRAM: Interruption code zero: RETRN08.
Otherwise: ATIMER.

ATIMER: Interruption code greater than 15: CATEST.
Otherwise: TESTOK.

ASVC: SVC code greater than 64: CATEST.
Otherwise: TESTOK.

ASYNCH: Interruption code zero: RETRN08.
Interruption code greater than six: CATEST.
Otherwise: TESTOK.

Other Types: TESTOK.

At CATEST, this routine tests two of the parameters provided by the expansion of the INTINQ macro instruction. If neither "CLEAR" nor "ANY" were specified, processing continues at RETRN08. Otherwise, at TESTOK, if the ICB was for a synchronous or asynchronous I/O interruption, the following six error checks are performed; if not, only the last is done:

- ICB does not point to a DCB -
- No DE-ICB match -
- DCB not open -
- No DEB pointer - In each of these cases, processing resumes at RETRN08.
- No DE-ICB match -
- No RE-ICB match - Processing continues at RTRN04. A return code of 4 is placed in register 15, and, at INTRTRN, task interruptions are permitted if they were on entrance, and control is returned to the calling routine.

When there are none of the above errors, but there are no QEs on the RE, a test is made at RETRNMOC to see if the interruption routine is in the wait state, pending the availability of the desired information. When the routine is in the wait mode, the AWAIT macro is issued. Task interruptions are then disabled, and control is returned to the calling routine.

When there are QEs on the RE, a test is made on the INTINQ type. If it is "CLEAR", all QEs are zeroed and dequeued, any associate message areas are freed, and return processing continues at INTRTRN, with a return code of 0 being placed in register 15. When the interruption was asynchronous and the ICB is for an attention from SYSIN, the attention counter is decremented.

When the INTINQ type is "ANY", a check is again made to see if the interruption code in each type of QE matches one allowed by the ICB.

- No QE-ICB match - Processing continues at RETRNMOC.

When the match is found, processing continues at MOV. If the RE is non-privileged, the privilege class of the COMAREA is tested, using the CKCLS macro instruction.

- Privileged COMAREA being used by nonprivileged user - Processing continues at RETRN08.

The necessary data is then moved from the QE into the COMAREA and the attention counter is decremented. If the QE is for an external interruption, the message is moved from the QE message area to the user area. CKCLS is used to test the user area.

- Nonprivileged user has privileged message area - The message is moved into an area obtained by issuing the GETMAIN macro instruction, rather than the user-specified area.

FREEMAIN is called to free the original message area. Information about the message and message area is moved into the ICB, and control is returned to the calling routine.

If the type is not "ANY," a QE must be found with an interruption type that matches the one specified in the parameter list.

- No QE-parameter list match - Processing continues at RETRNMOC.

When the proper QE has been found, processing continues at MOV.

Routines Called: FREEMAIN (CZCHA3).

Exit: Return with the appropriate return code.

STIMER/TTIMER Routine

Entry Point: CZCJA1. The linkage is created during expansion of the STIMER/TTIMER macro instruction.

Operation: This routine performs the following functions:

1. Sets a clock.
2. Cancels a clock.
3. Sets a clock and waits for the interruption from that clock.
4. Tests a clock.

On entry this routine checks for the following error conditions:

- Clock number is invalid -
- Clock number indicates a privileged clock used in a function other than "TEST" - In either of these cases, an error return code of four is set at RETURN04 and control is returned to the calling routine.

Test Task Clock -- When the clock has not been set, control is returned to the calling routine with an error return code. Otherwise, the XTRTM macro is used to find out how much task time has already been used. This is subtracted from the requested amount to get the remaining task time. Control is returned to the calling program with this remaining time as a parameter.

Test Real Clock -- When the clock has not been set, processing is the same as above for that condition. Otherwise, the REDTIM macro is issued to find the amount of real time that has been used by the task. The real time remaining to it is then computed, and converted to milliseconds. This number is stored in register 0, and control is returned to the calling routine.

Set Function -- If there is no ICB pointer, GETMAIN is used to get main storage for an ICB and communications area. The address of this ICB is stored in the SIR parameter list. This routine then branches to the SIR routine (CZCJSA) to specify the ICB for the timer interruption.

- SIR request brings total task time above 7 1/2 hours - A return code of eight is placed in register 15, and control is returned to the calling routine.
- Other abnormal return - Processing continues at RETURN04.

If SIR operation was normal, a successful return code of zero is set at RETURN00, and control is returned to the calling program.

Cancel Function -- This section of the program immediately checks for the following error:

- No ICB pointer - In this case, processing continues at RETURN04.

If there is an ICB address, it is placed in the DIR parameter list and this routine branches to DIR in order to delete the ICB.

- Error return from DIR - Processing continues at RETURN04.

Upon successful return from DIR, processing terminates at RETURN00. The time remaining when the task was canceled is passed to the calling routine.

Wait Function -- If there is no ICB address, GETMAIN is called to get main storage for the ICB and a COMAREA. The ICB address is then placed in the SIR parameter list, and the entry point in the ICB is zeroed. This routine then branches to INTINQ in the wait mode.

- Error return from INTINQ - Processing continues at RETURN04. Upon normal return from INTINQ, exit is at RETURN00.

Routines Called: SIR (CZCJS), DIR (CZCJD), INTINQ (CZCJI).

Exit: Return to the caller with the appropriate return code.

Set Clock Routine

Entry Point: CZCJYE. It is called by other task monitor routines via type-I linkage.

Operation: On entrance, this module calls the task monitor time conversion routine (CZCJX), to convert the input time into microseconds. Checks are made on the success of CZCJX, the clock number, and the clock type:

- CZCJX unsuccessful -
- The clock number is invalid -
- The clock type is invalid - In each of these cases, an error return code is set at ST01, and control is returned to the calling routine.

Real Time Clock -- If the clock to be set is presently active, cancel clock (CZCJZ) is called to turn if off. The desired interval is then set in the clock and it is activated. When the pointer in the header to the first clock is zero, and the time is not to be set on the last clock in the chain, the clocks are sorted into time order, a successful return code is set, and control is returned to the calling program. If the clocks do not need to be sorted, the SETTR macro is issued. The return code from SETTR is then tested:

- Time requested has passed -
- System limit was reached - On either of these conditions, an error return code is set, and control is returned to the calling routine.

If the time set was accepted, the interval is placed in the clock header, a successful return code is set and control is returned to the calling routine.

Task Time Clock -- The requested time interval is first changed from microseconds into milliseconds. If the clock to be used is active, it is cancelled by the cancel clock routine (CZCJZ). The XTRTM macro is then used to extract the amount of CPU time the task has already used. The amount of further task time requested is then added to this.

- Total task time is greater than 7 1/2 hours - In this case, an error return code is set and control is returned to the calling routine.

When the amount of requested task time is permissible, it is moved into the clock and the clock is activated. If the pointer in the header to the first clock in the active chain is zero, and if the clock being set is not the last on the chain, the clocks are sorted by time, a successful return code is set and control is returned to the calling routine. When a sort is not necessary, the SETTU macro is issued, the successful return code is placed in register 15, and control is returned to the calling routine.

Routines Called: Cancel clock (CZCJZ), time conversion (CZCJX).

Exits: Return to the calling program with the appropriate return code.

Cancel Clock Routine

Entry Point: CZCJZE. It is called by other task monitor routines via type-I linkage.

Operation: The following error conditions are checked:

- Clock number invalid -
- Clock type invalid - In either of these cases, an error code is set and control is returned to the calling routine.

When the clock to be canceled is not active, a successful return code is set at CCRET, and control is returned to the calling routine. When it is active, it is canceled. If it is the first clock on the active list, the pointers on the active list are adjusted, and processing continues at CCRET.

- Pointer to the active list is zero - An error return code is set and control returns to the calling program.

When the canceled clock is the last clock on the active list, the forward pointer on the previous clock is adjusted and processing continues at CCRET.

If the canceled clock is not the last active clock, it is removed from the active list and the pointers of the clocks around it are adjusted. Processing continues at CCRET.

Routines Called: None.

Exits: Return to the calling routine with the appropriate return code.

Time Conversion Routine (CZCJX)

Entry Point: CZCJZA. The time conversion routine is called by set clock via type-I linkage.

Operation: This routine contains a group of subroutines that convert time parameters, in any of several formats, into a form suitable for input to the SETTR supervisor program or, when converted to milliseconds, for the SETTU supervisor program. Initially, the validity of the input data, as specified in the parameter list, is checked. If this input is invalid, control goes to CZCJXB to cause a SYSER. If the input is valid, control goes to CZCJXC for conversions required for task time, or control passes to CZCJXD for conversions required for real time. At these two points, the conversion code is used as an index to a transfer vector that causes control to pass to either the proper data conversion program, or to an error routine if the type of conversion indicated is invalid for the type of time required.

As the data is being converted, and if another conversion routine is needed for immediate processing, a direct branch is made to the required routine. When that routine has completed processing, it uses the conversion code to determine the point at which it should return. In addition, for real-time requests, the system time-of-day value is added to the requested time, if a decimal or binary interval was specified.

The discussion below provides a descriptiom of the various conversion subroutines:

DECIMAL INTERVAL (CZCJXF): A decimal interval, given as eight unpacked decimal characters, is converted to binary by first converting to packed decimal format and then isolating hours, minutes, and seconds. The hours, minutes, and seconds are converted to microseconds, and an accumulated total is formed. This total is then converted into binary form. An exit is made to either the time-of-day (CZCJXHI) routine or the conversion executive (CZCJXE).

BINARY INTERVAL (CZCJXG): A binary interval, expressed as a 32-bit number of milliseconds, is converted to microseconds by multiplying it by 1000. The output is stored, and an exit is made to the conversion executive (CZCJXE).

TIME-OF-DAY (CZCJXH): To convert a time-of-day, given as eight unpacked decimal numbers, to a unique calendar or real time, the value is first converted to microseconds. Then the current time is obtained from the supervisor, as well as the current serial day number. Today's time-of-day is obtained and compared with the desired time-of-day. If the present time-of-day is less than the requested value, the current serial day number is added to the input time-of-day; the result is the final output. If the present time is greater than the requested time-of-day, the serial day number is increased by one before adding it to the input time-of-day. Exit is made to CZCJXE.

DAY-OF-THE-WEEK (CZCJXI): The current serial day number is obtained from the supervisor. The serial day number of a known Sunday is subtracted from this number. The result is divided by seven to obtain a number of weeks and a remainder. A search is then made to determine which day-of-the-week is being requested. The number of the day-of-the-week that is wanted is then used to determine the days to be added to the current search day number. If the number to be added is found to be negative, a week (seven days) is added to the serial day number. Otherwise, the positive number of days is added. The result is converted to microseconds and stored. Exit is made to CZCJXE.

DAY-OF-A-MONTH (CZCJXJ): The current serial day number is obtained from the supervisor and converted by CZCJXM to a calendar date. From this date the current day-of-the-month is extracted. Then, if the requested day-of-the-month is beyond the current one, 1 is added to the current month. If the month becomes 13, 1 is added to the year, and the month is reset to 1. A search is then made for the next month in which the requested day-of-the-month is valid. When found, the resulting calendar data is converted by CZCJXL to a serial day, and then to microseconds. Exit is made to CZCJXE.

DAY-OF-A-YEAR (CZCJXK): The day part of the input is used to determine the month and day that corresponds to the calendar date, leap year being considered. This is converted to a serial day number and then to microseconds. The output is stored and an exit is made to CZCJXE.

CALENDAR DATE (CZCJXL): This subroutine converts a calendar date to a serial day by using an algorithm. If the month portion of the input exceeds 2, 3 is subtracted from it; otherwise, 9 is added, and 1 is subtracted from the year. The serial year is computed as (1461y)/4. Next, the month is converted as (153m+2)/5. The year, month, and day are then

combined and converted to binary to form a serial day, which is stored as output. Control returns either to CZCJXJ2 or CZCJXK1.

SERIAL DAY (CZCJXM): This subroutine converts a serial day number to a calendar date. First, the input is converted to packed decimal, and the final year, month, and day portions of the output are computed. If the month is less than 10, 3 is added. Otherwise, the month is reduced by 9, and 1 is added to the year. Exit is made to CZCJXJ1. After the necessary conversions have been made, control returns to CZCJXE, where registers are restored, and control is returned to the calling program.

Routines Called: None.

Exit: Return.

Leave Privilege Routine

Entry Point: CZCJLE. Leave privilege may only be called by a privileged program, via type-I linkage.

Operation: Task interruptions are disabled using the ITI macro instruction. The following errors are then checked:

- The enter flag is off -
- The P3 flag is on -
- The current VPSW is nonprivileged - When any of these are true, the SYSER macro is issued for an invalid use of the leave privilege routine, and exit is to ABEND (CZACP).

Otherwise, the ISA save area 1 and the enter save area are combined into a LVPRV save area. The GETMAIN routine (CZCH2) is used to get a nonprivileged short save area, and the P3 flag is turned on to indicate type-III linkage. This routine then executes an LVPSW instruction to give control to the nonprivileged program.

Routines Called: GETMAIN (CZCH2).

Exits: ABEND, nonprivileged routine via LVPSW.

Cleanup Routine

Entry Point: Either CZCJCU via type-I linkage or CZCJC2 from the ENTER SVC routine.

Operation: At CZCJCU this routine first inhibits task interruptions if necessary, using the ITI macro instruction. The cleanup flag is then turned on, and cancel clock (CZCJZ) is called to cancel all nonprivileged clocks, real or task time.

When the HARE is nonzero, its privilege is checked. If it is nonprivileged, it is zeroed. If it is privileged, the pointers to any nonprivileged REs chained to it are zeroed.

When CARE is not zero but nonprivileged, it is zeroed. This routine then checks for nonprivileged QELs. If there are any they are cleared, and their space is set available. When any of the QELs are external, the message areas associated with them are freed. After the QELs are cleared, or if there were no nonprivileged QELs, processing continues at CC10 with the next DE. When the HARE on this DE is nonzero and nonprivileged, it is zeroed. The REs are then examined, starting with highest priority RE. Privileged REs are skipped. When the pointer to the RE is not zero it is zeroed. At CC40, if there are any QEs on the RE, they are cleared, and their space is made available. The message areas for

any external QEs are freed. When there are no more QEs, the RE is cleared and made available. If there are any more REs on this DE, their processing continues at CC40. When there are no more REs, or if their pointers were originally zeroed, processing of the next DE continues at CC10. When there are no more DEs, any nonprivileged save areas are zeroed. At CC20, if there are no pending type-III linkages, the pointer to the pushdown stack is zeroed, the PTI macro instruction is issued to permit task interruptions if they were permitted on entrance to this module, and control is returned to the calling program. If there are any pending type-III linkages, return is to the SVC processor's forced return point, CZCJR2.

When this routine is entered at CZCJC2 from the enter SVC processor, processing begins at CC20.

Routines Called: Cancel clock (CZCJZE), FREEMAIN (CZCH3).

Exits: Task monitor's ENTER SVC processor (CZCJR2), return to the calling routine.

FLOWCHARTS

The flowcharts in this manual have been produced by an IBM program, using ANSI symbols. The symbols are defined in the left column below, and examples of their use are shown at the right.

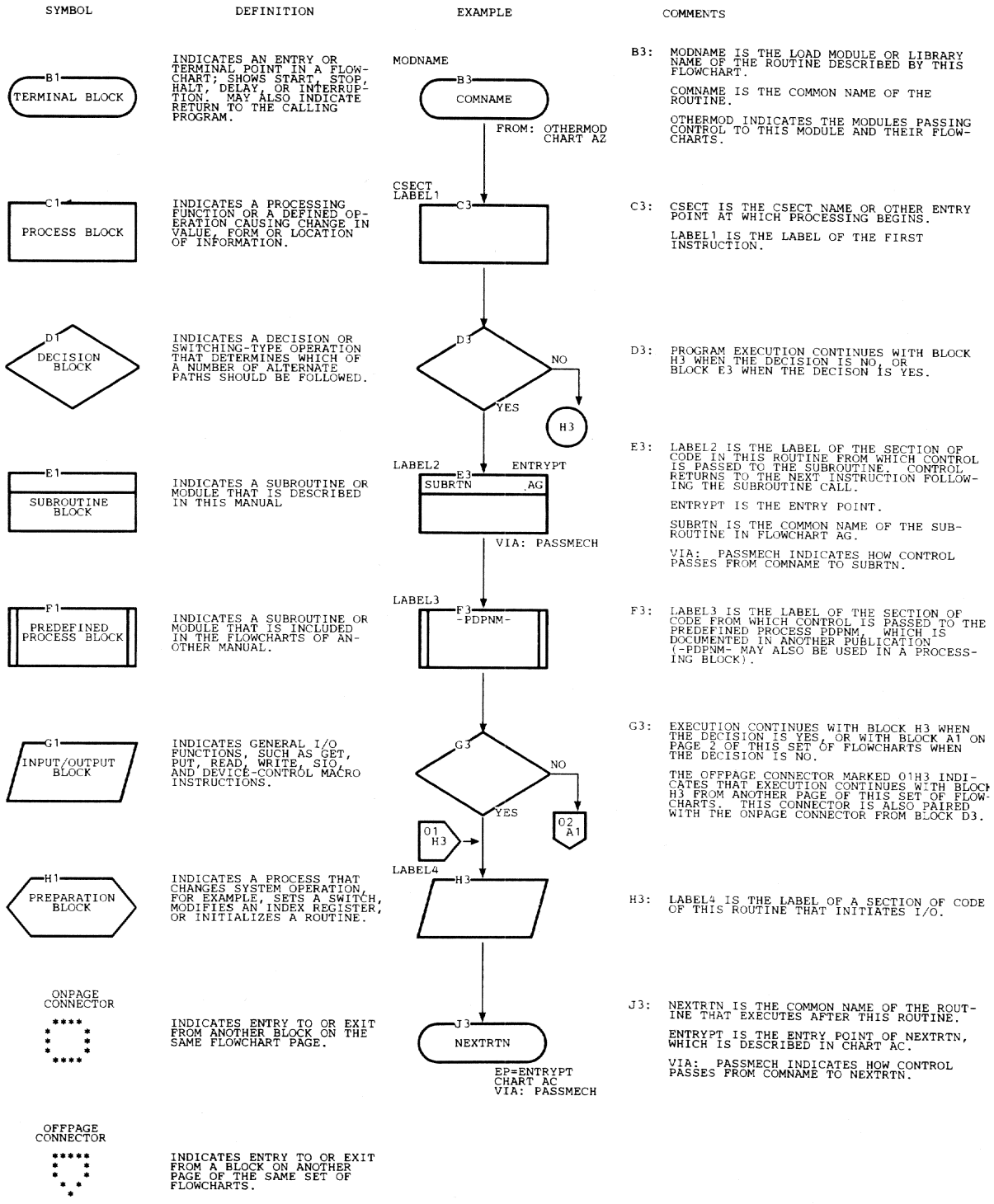


Chart AA. Program Interruption Processor (CZCJT) (Part 1 of 2)

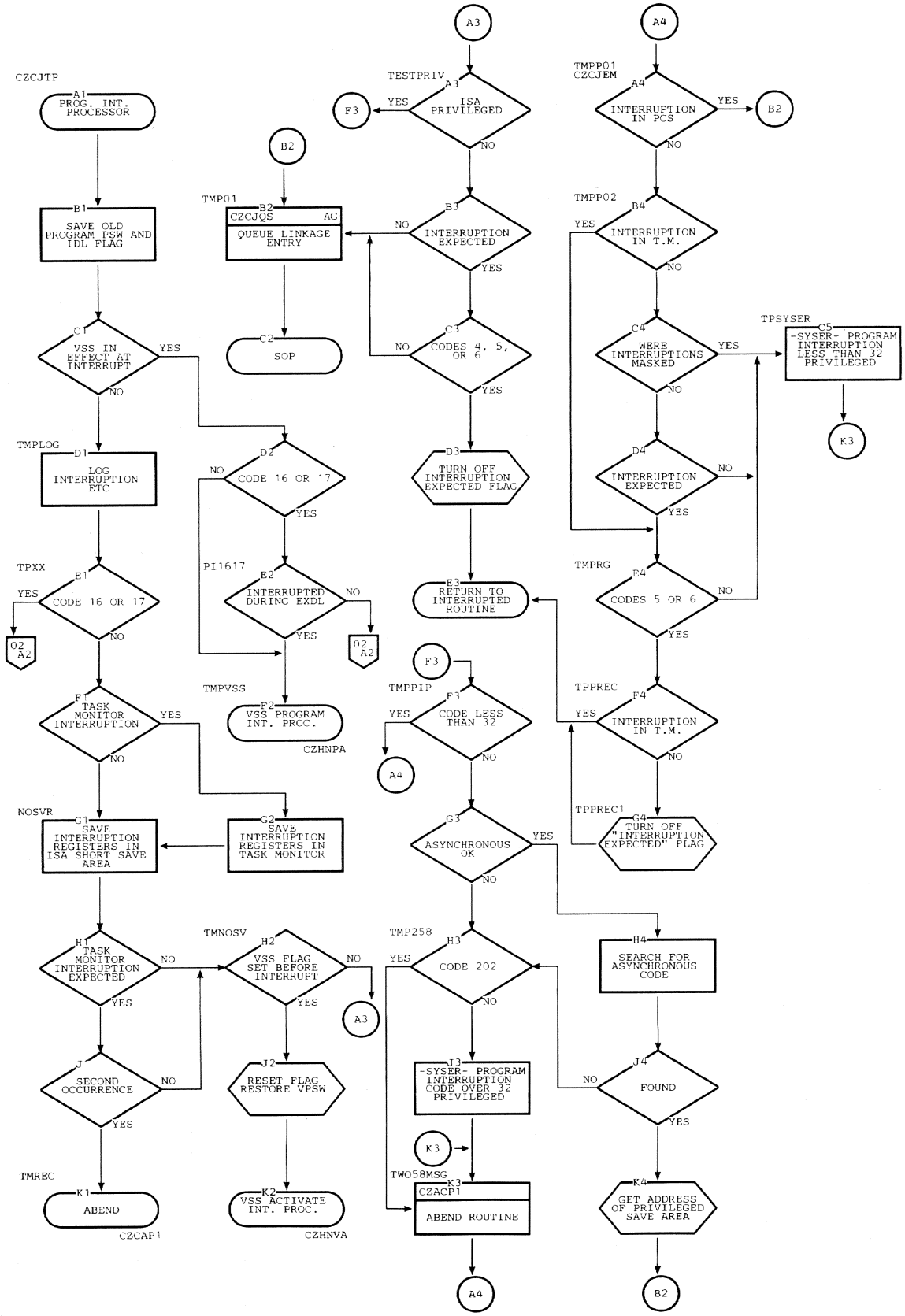


Chart AA. Program Interruption Processor (CZCJT) (Part 2 of 2)

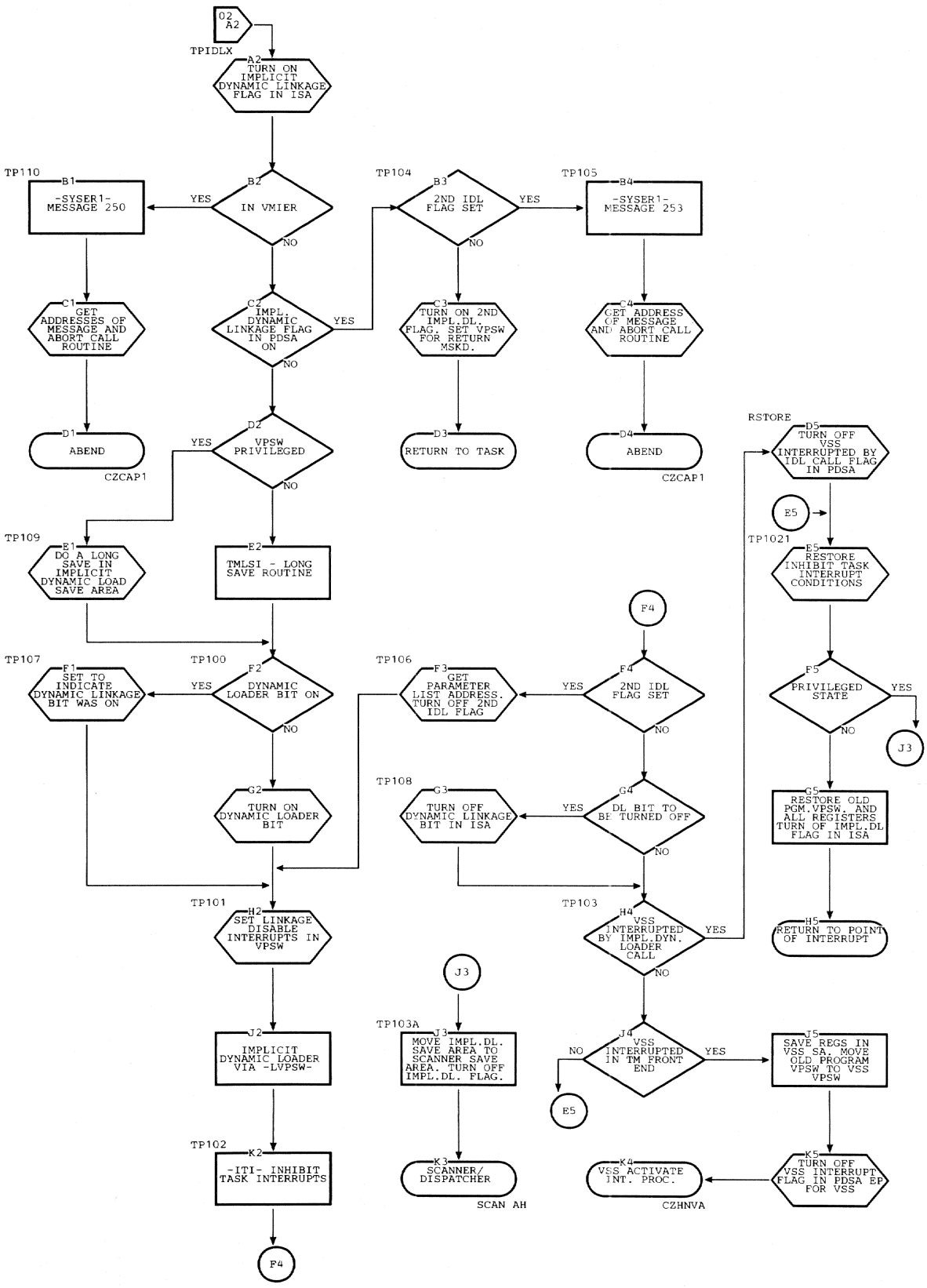


Chart AB. SVC Interruption Processor (CZCJT) (Part 1 of 3)

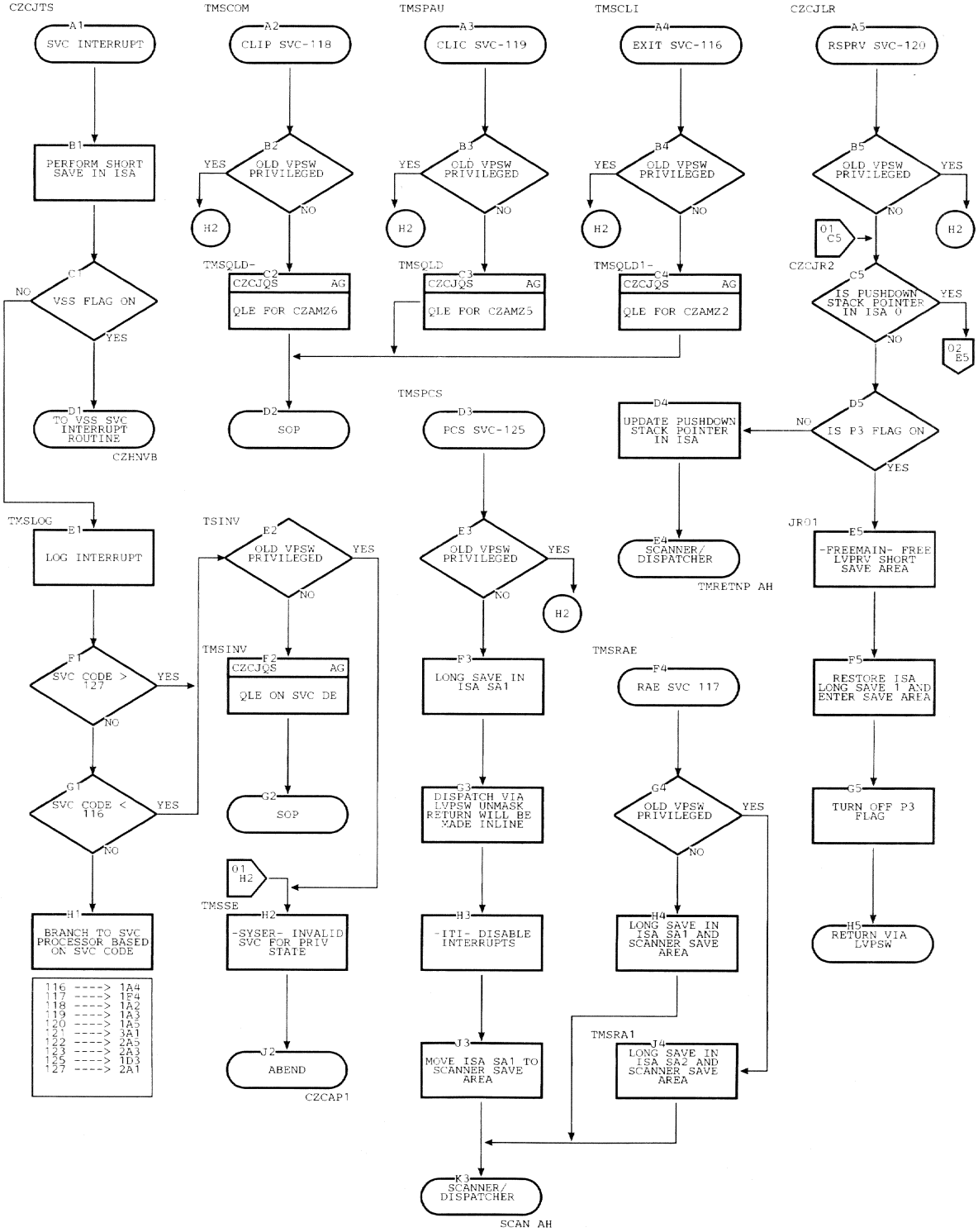


Chart AB. SVC Interruption Processor (CZCJT) (Part 2 of 3)

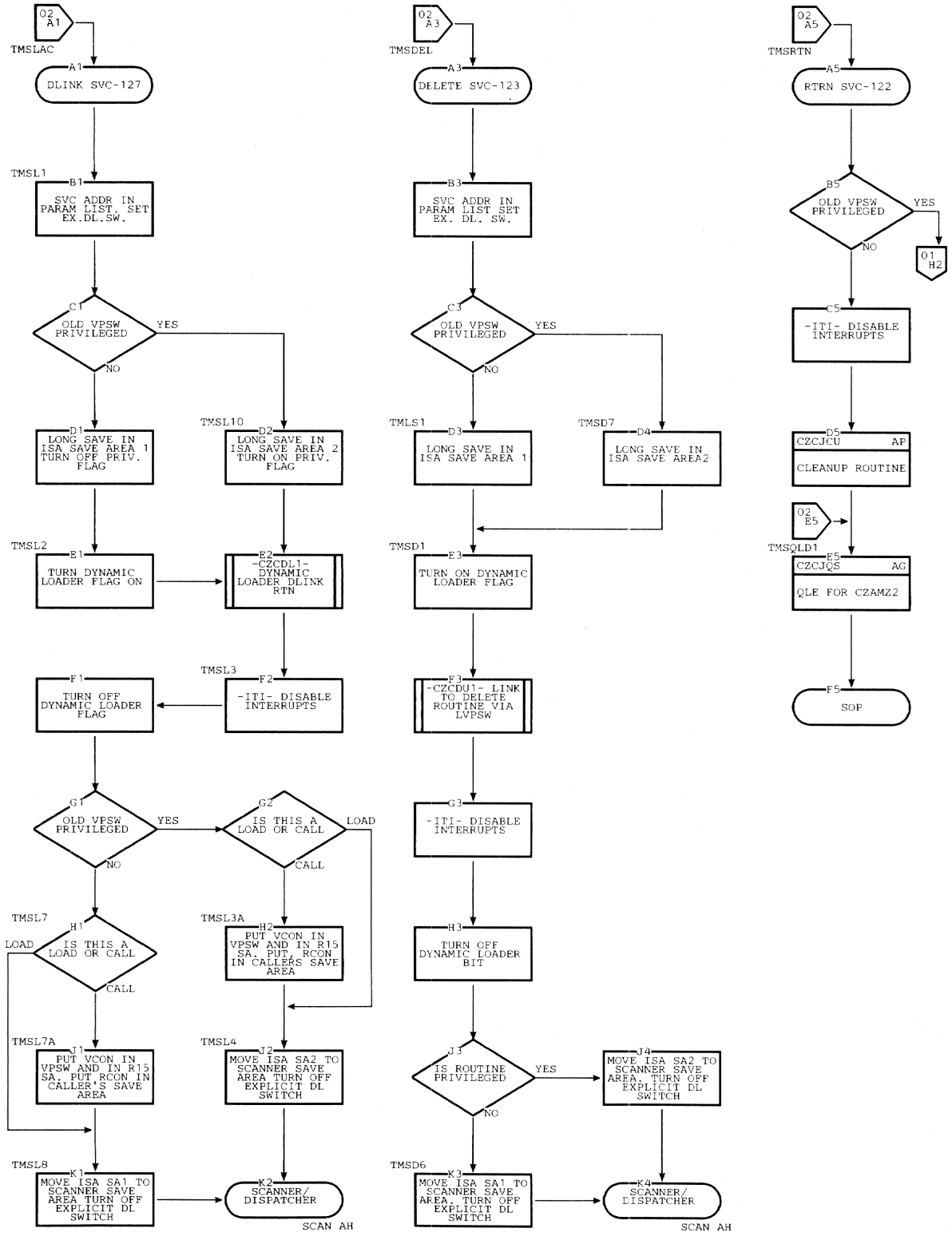


Chart AB. SVC Interruption Processor (CZCJT) (Part 3 of 3)

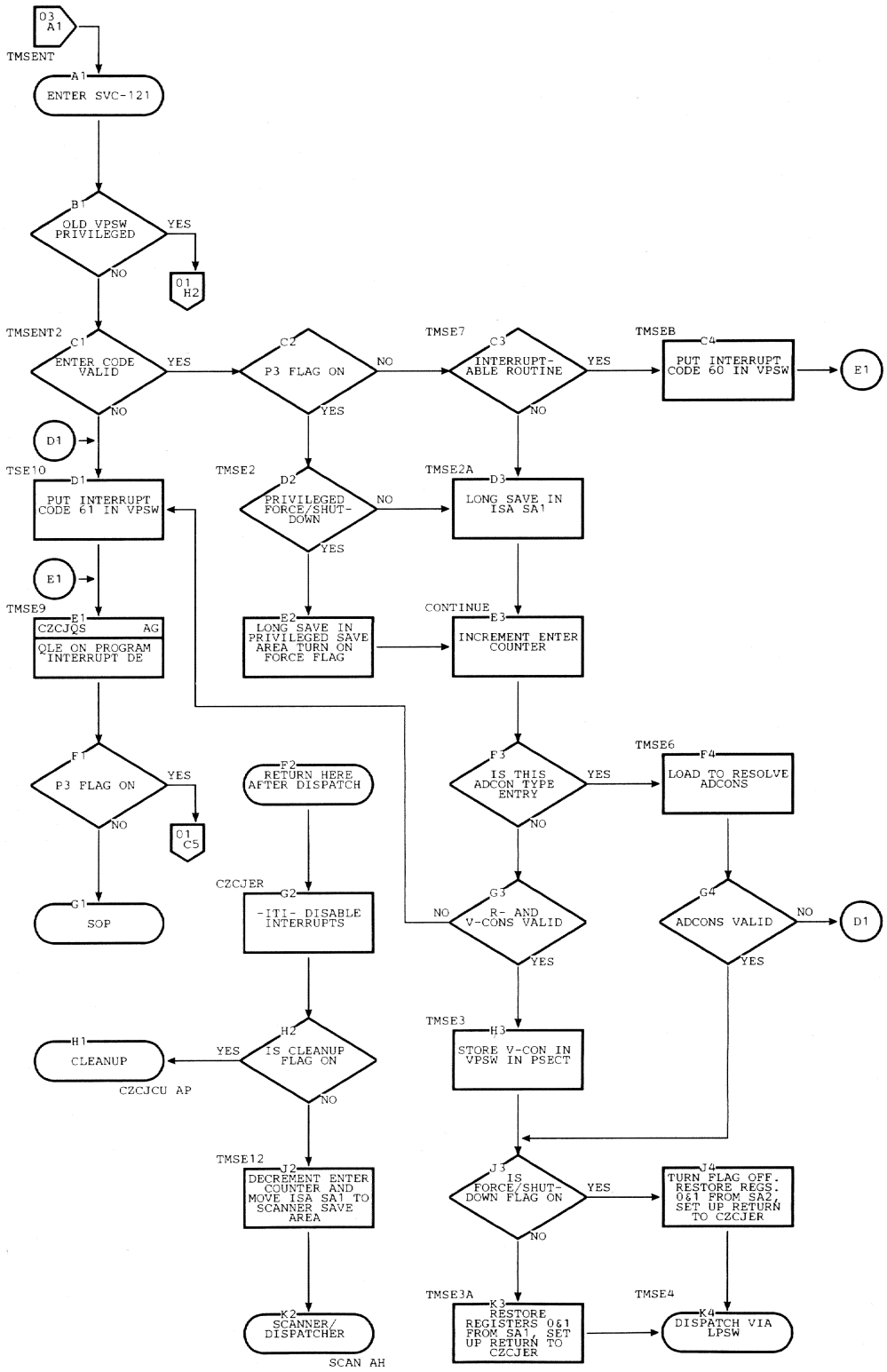


Chart AC. Timer Interruption Processor (CZCJT) (Part 1 of 2)

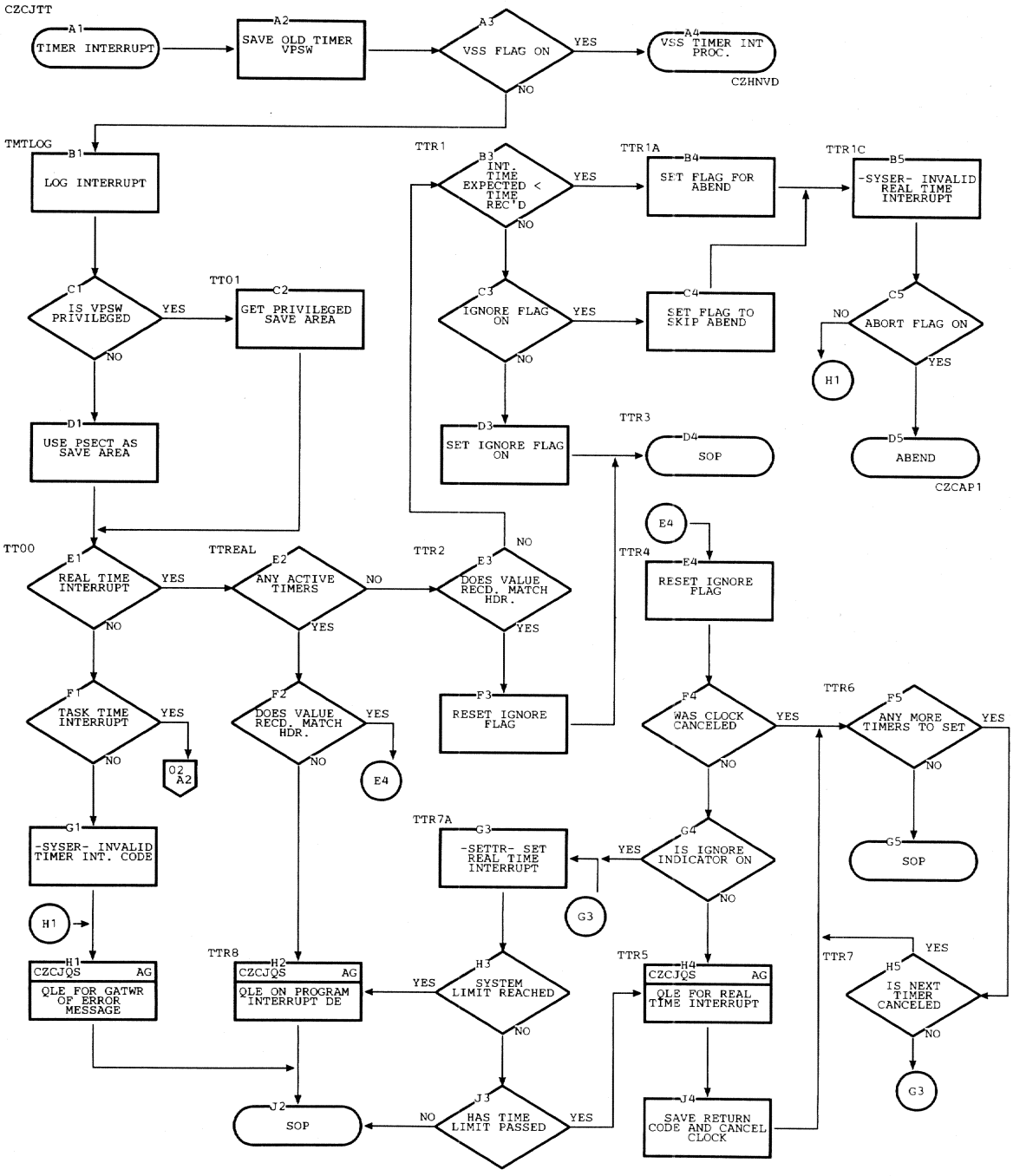


Chart AC. Timer Interruption Processor (CZCJT) (Part 2 of 2)

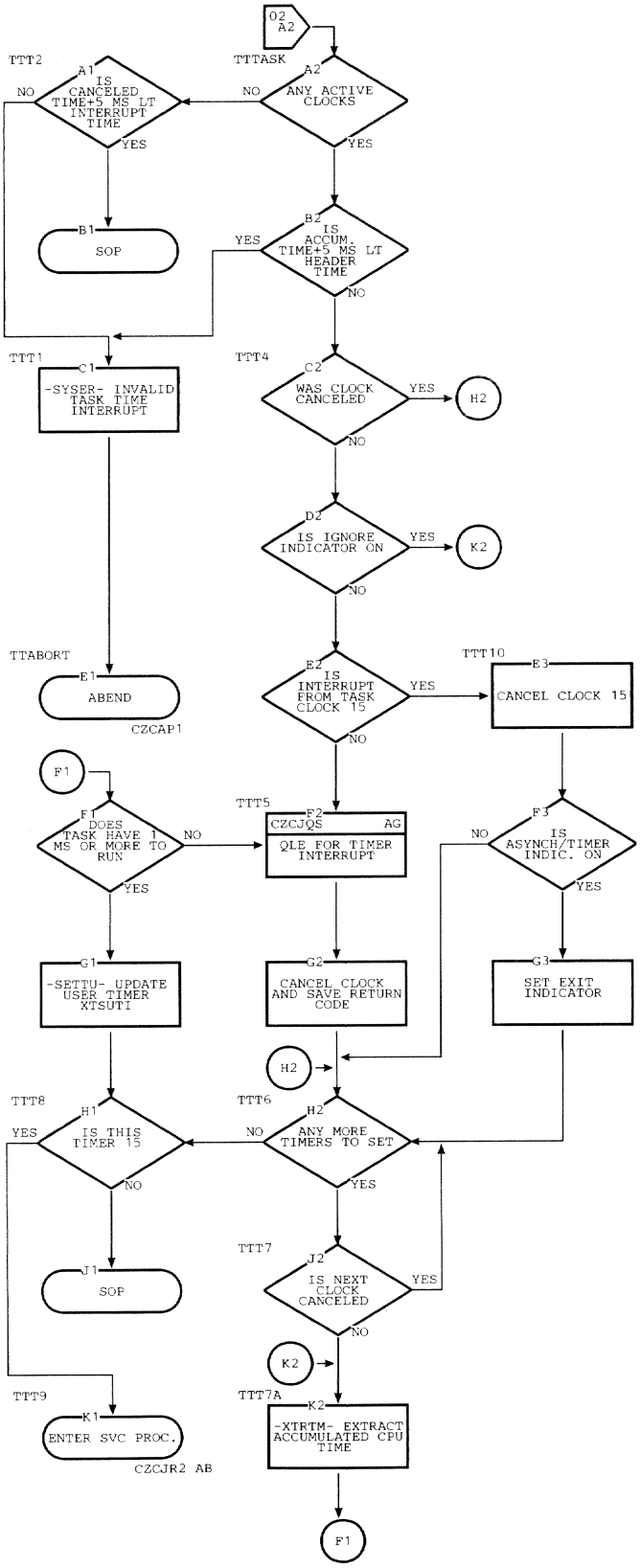


Chart AD. External Interruption Processor (CZCJT)

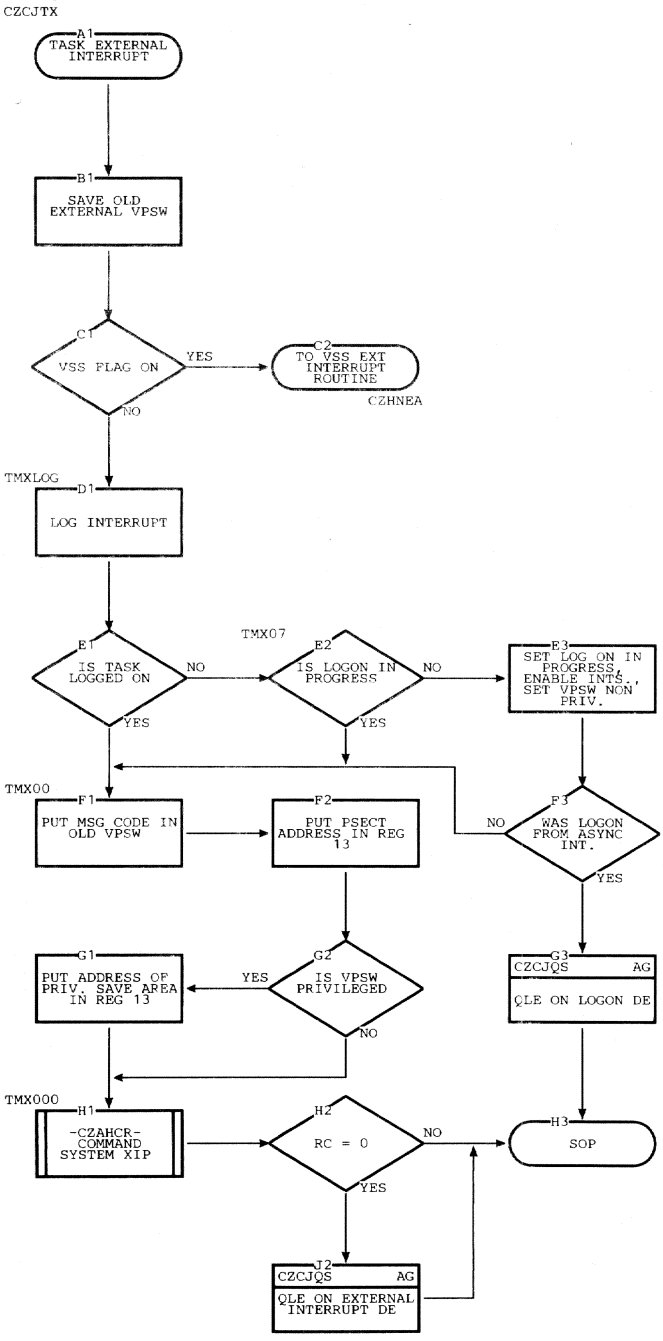


Chart AE. Synchronous and Asynchronous Interruption Processors (CZCJT)

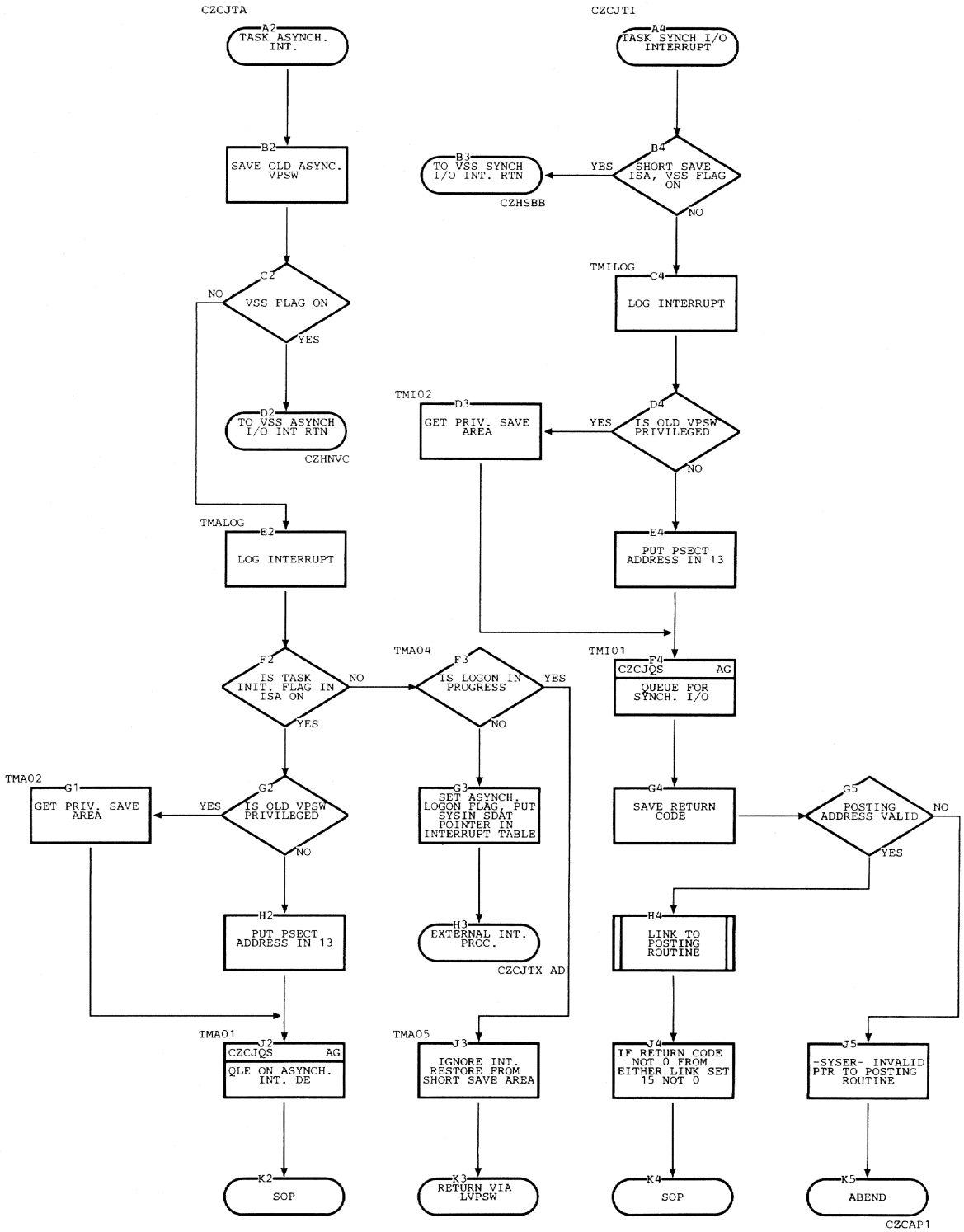


Chart AF. Data Set Paging and VSS Interruption Processors (CZCJT)

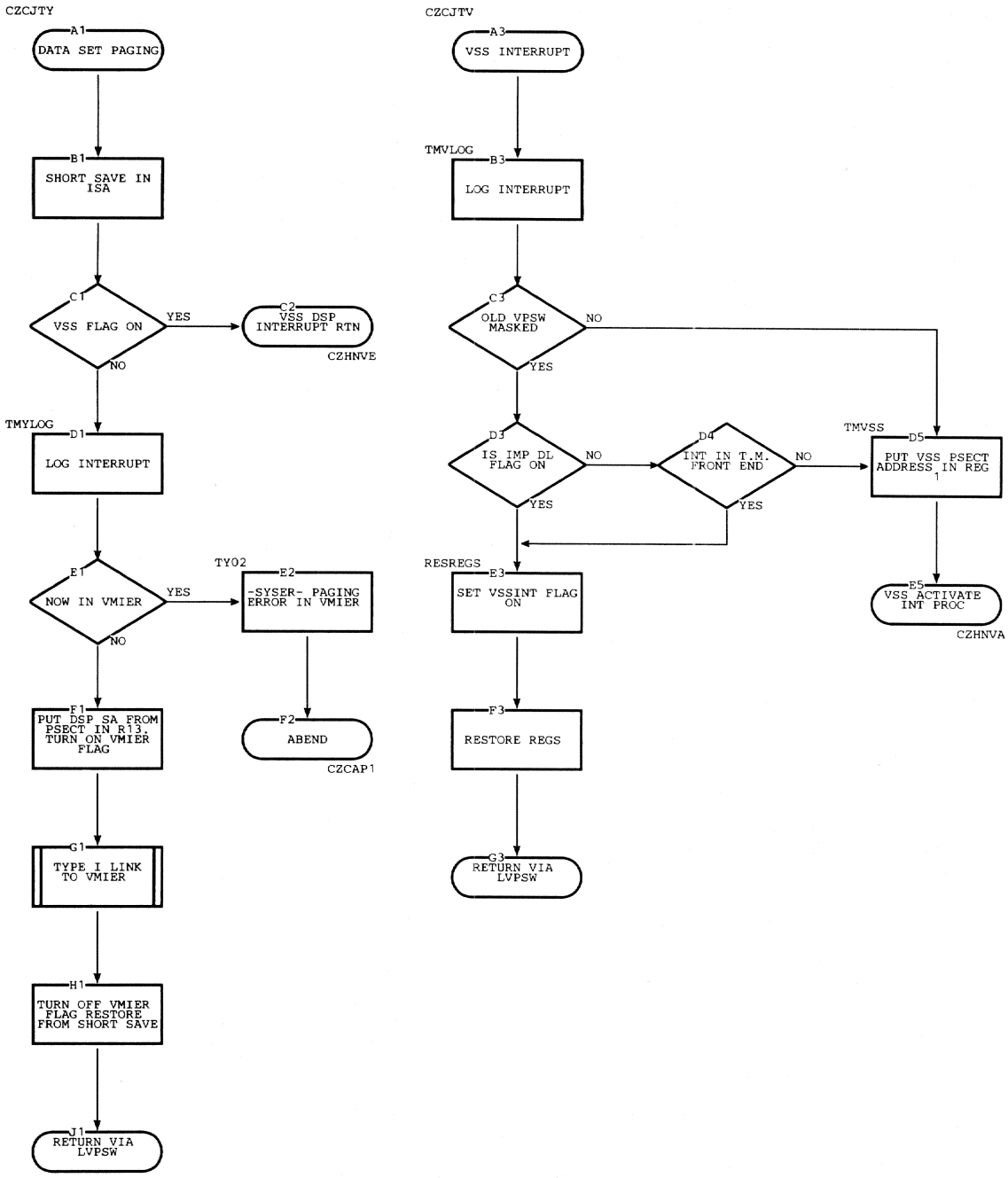


Chart AG. Queue Linkage Entry Routine (CZCJT) (Part 1 of 4)

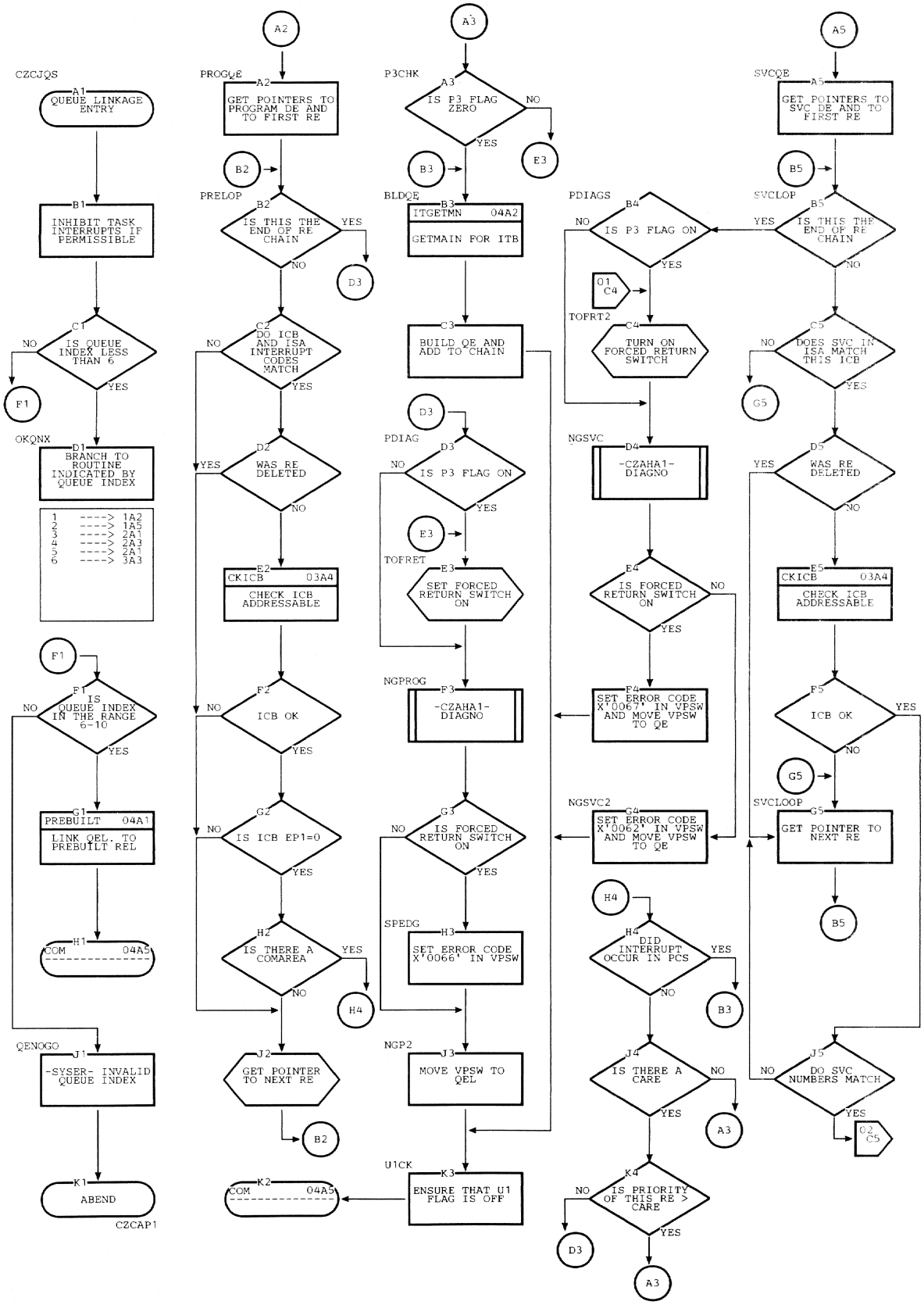


Chart AG. Queue Linkage Entry Routine (CZCJT) (Part 2 of 4)

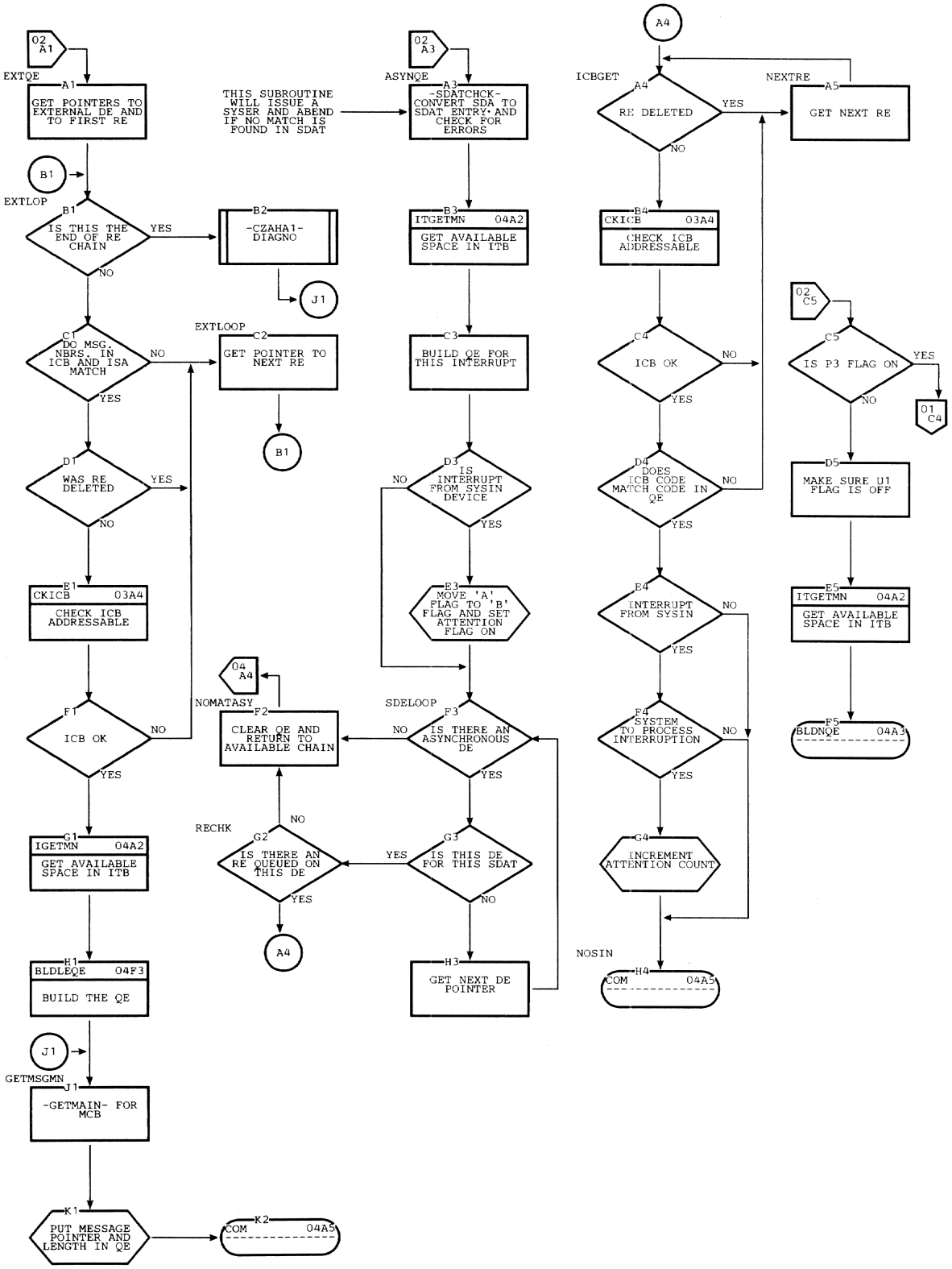


Chart AG. Queue Linkage Entry Routine (CZCJT) (Part 3 of 4)

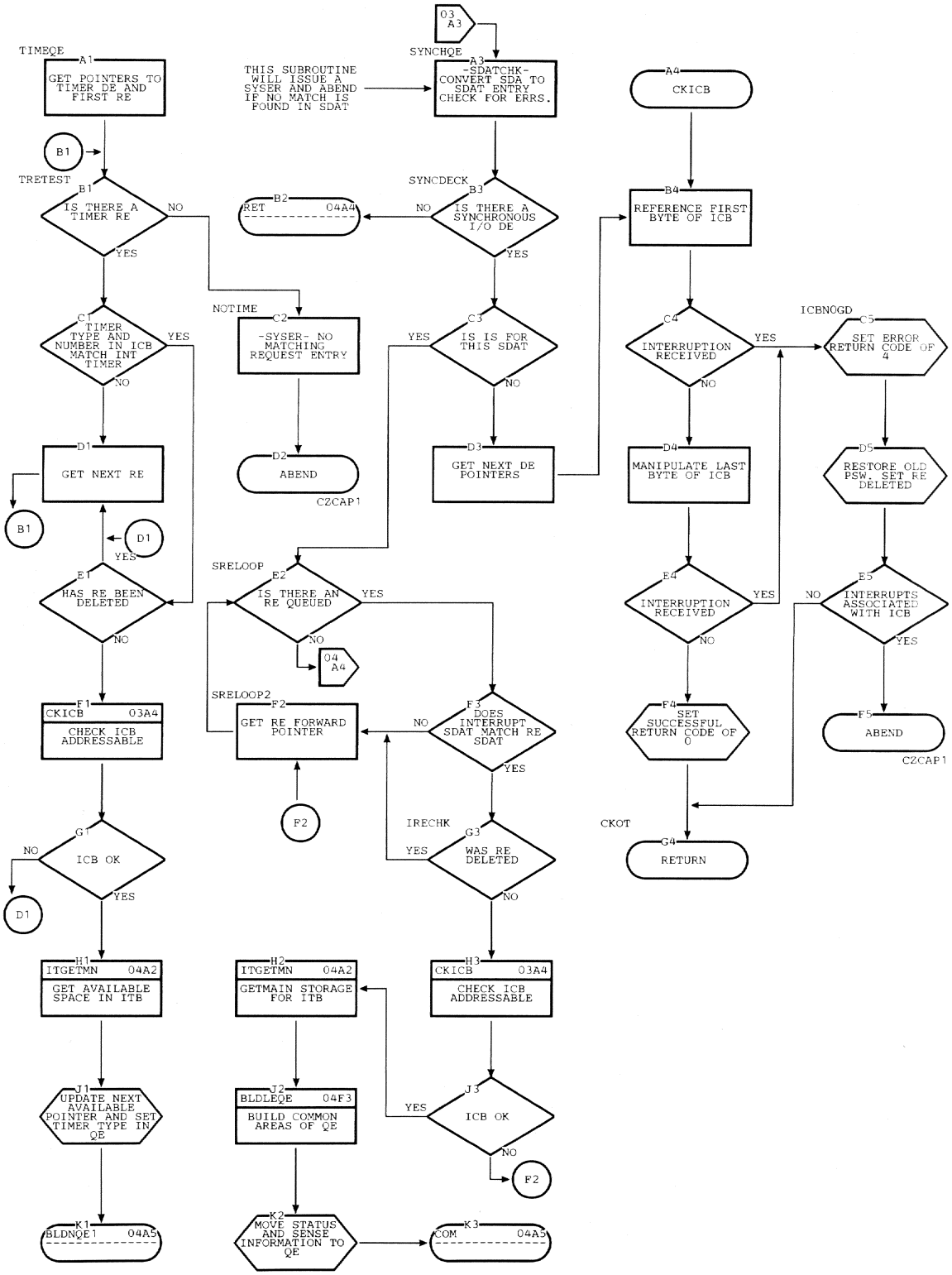


Chart AG. Queue Linkage Entry Routine (CZCJT) (Part 4 of 4)

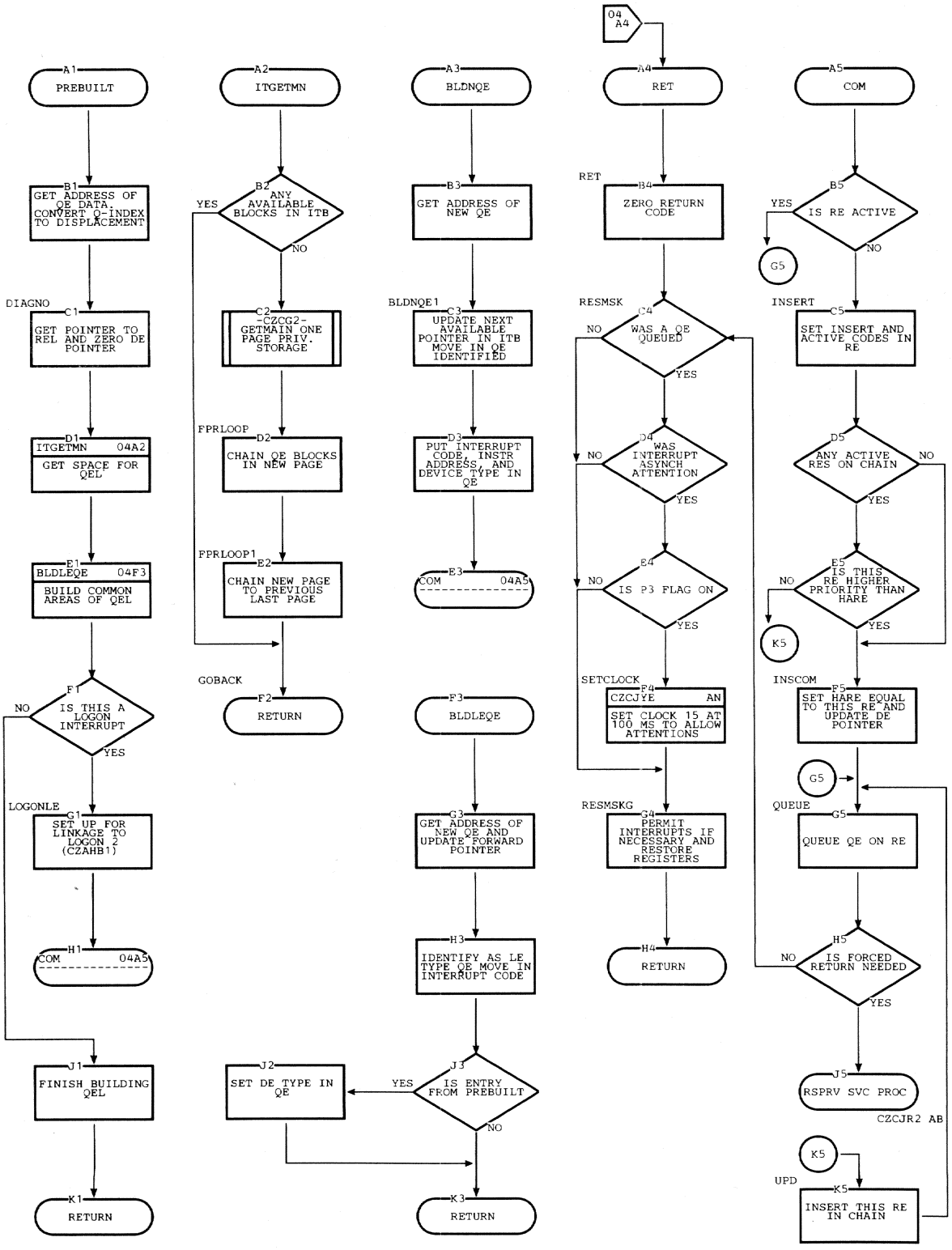


Chart AH. Scanner/Dispatcher (CZCJT) (Part 1 of 5)

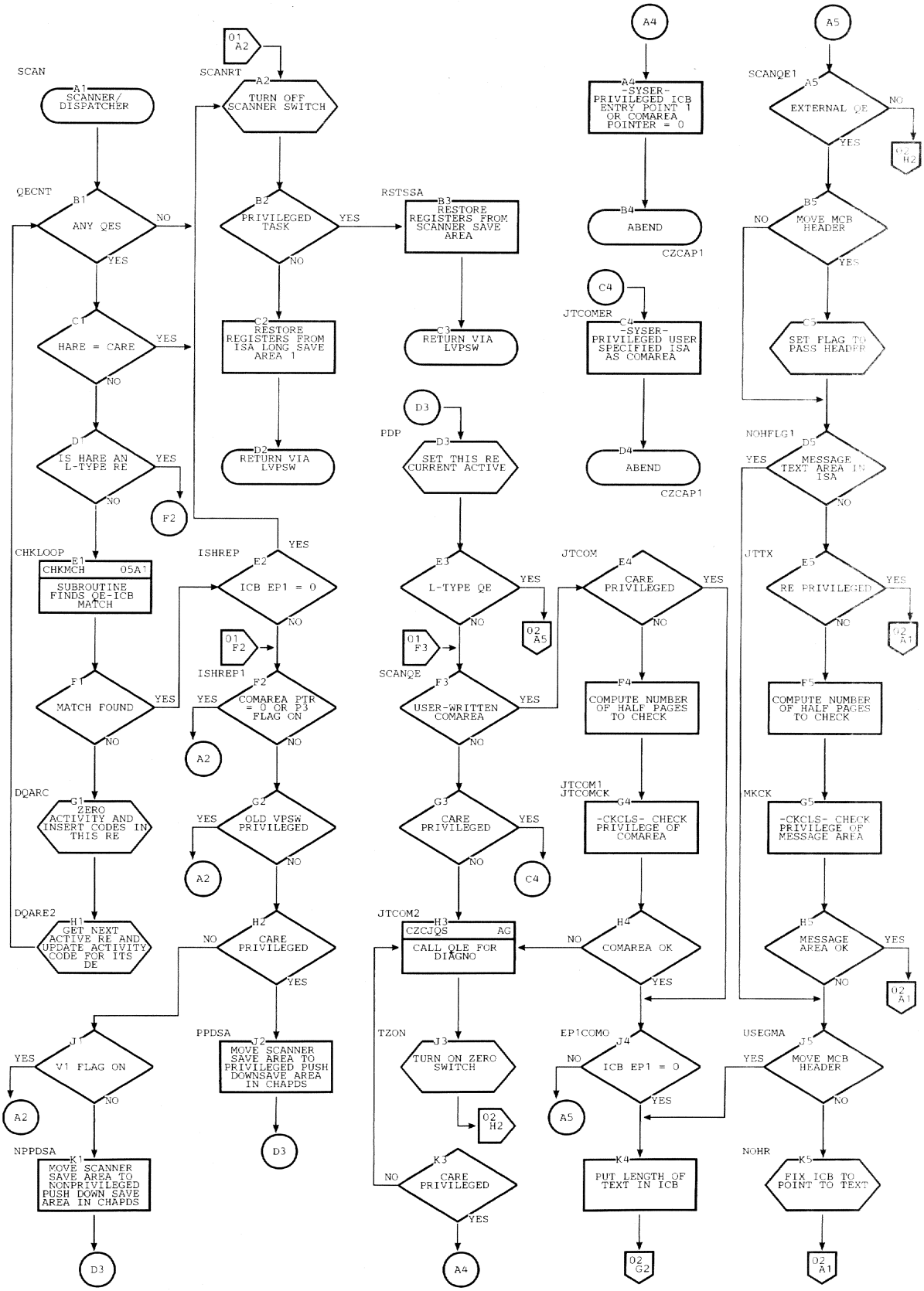


Chart AH. Scanner/Dispatcher (CZCJT) (Part 2 of 5)

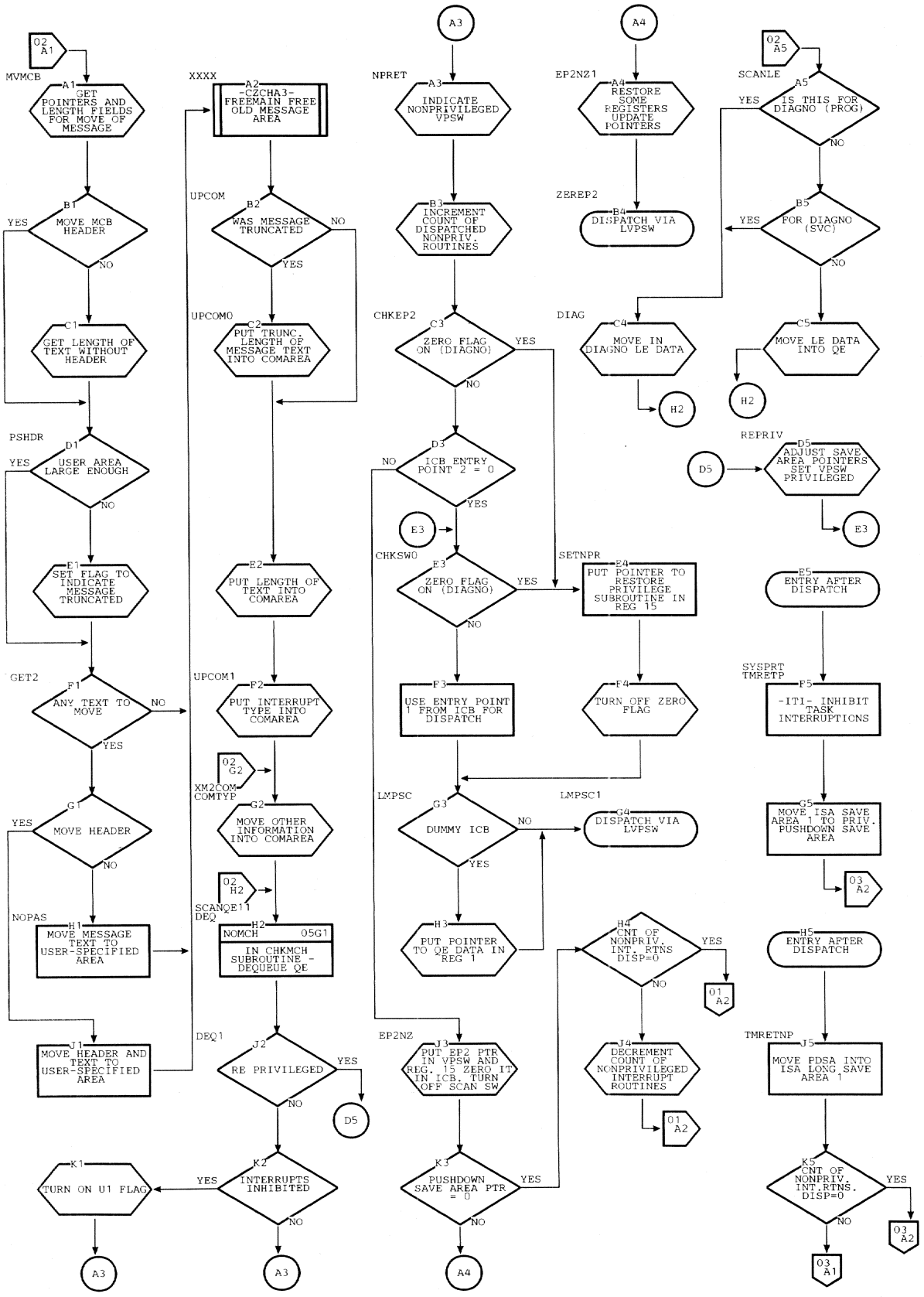


Chart AH. Scanner/Dispatcher (CZCJT) (Part 3 of 5)

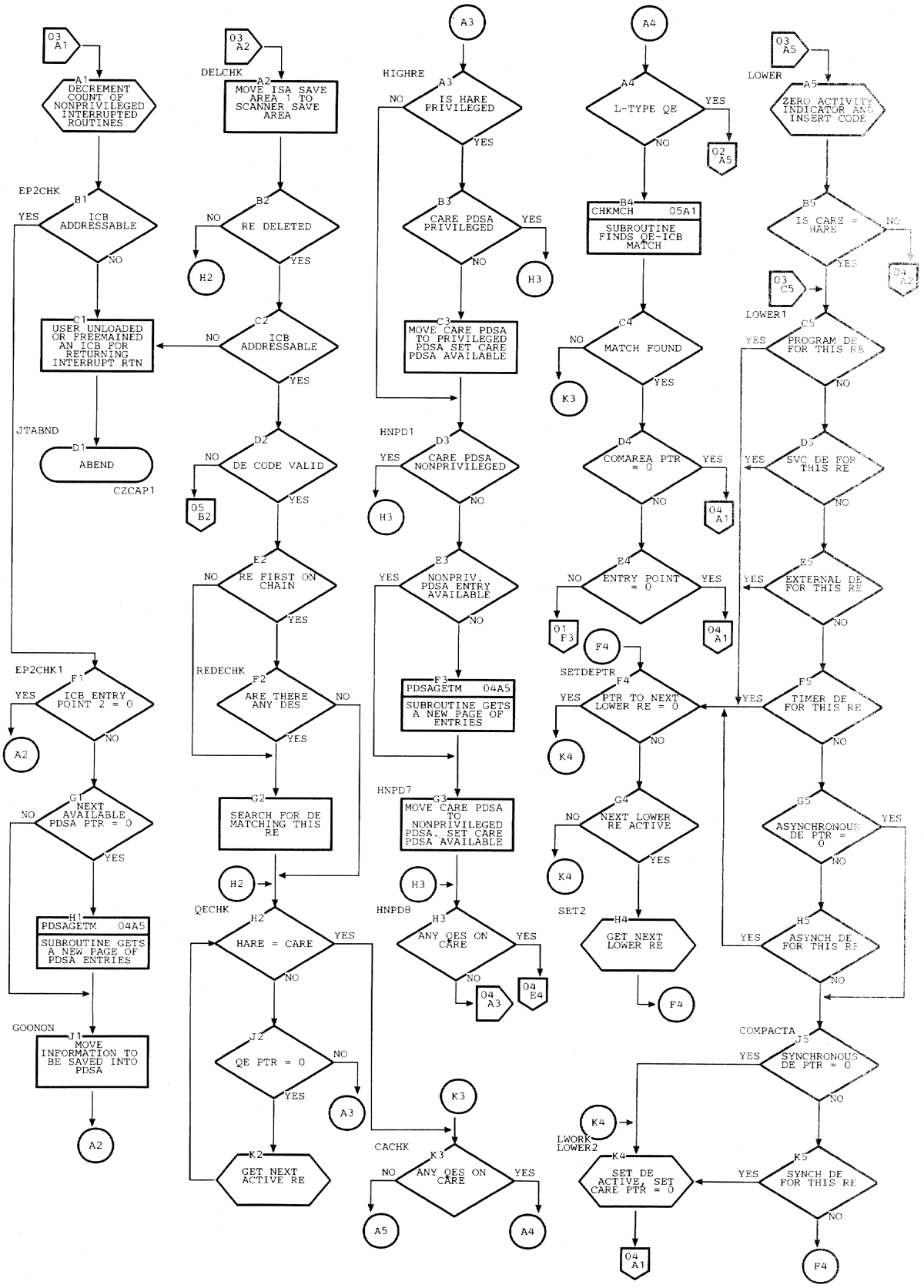


Chart AH. Scanner/Dispatcher (CZCJT) (Part 4 of 5)

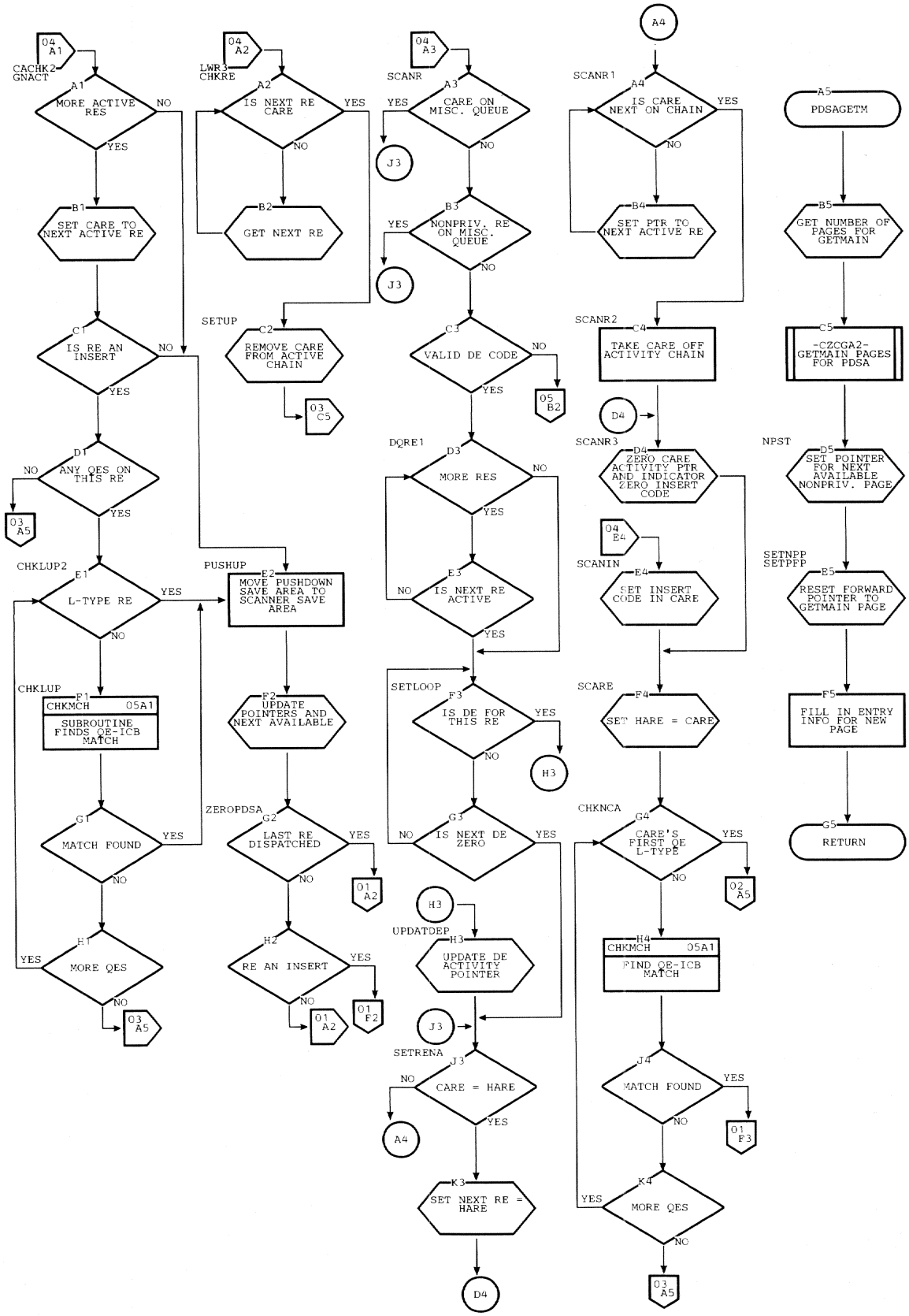


Chart AH. Scanner/Dispatcher (CZCJT) (Part 5 of 5)

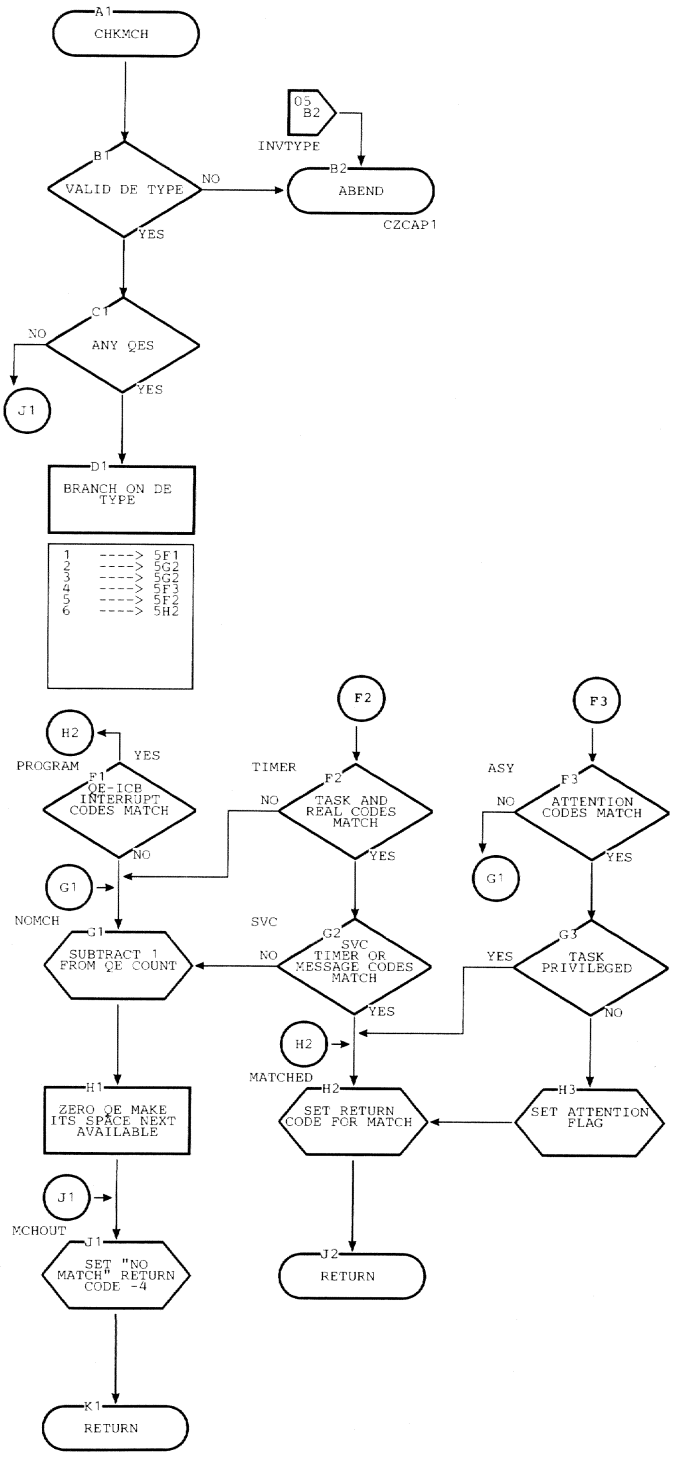


Chart AI. Specify Interruption Routine (CZCJS)

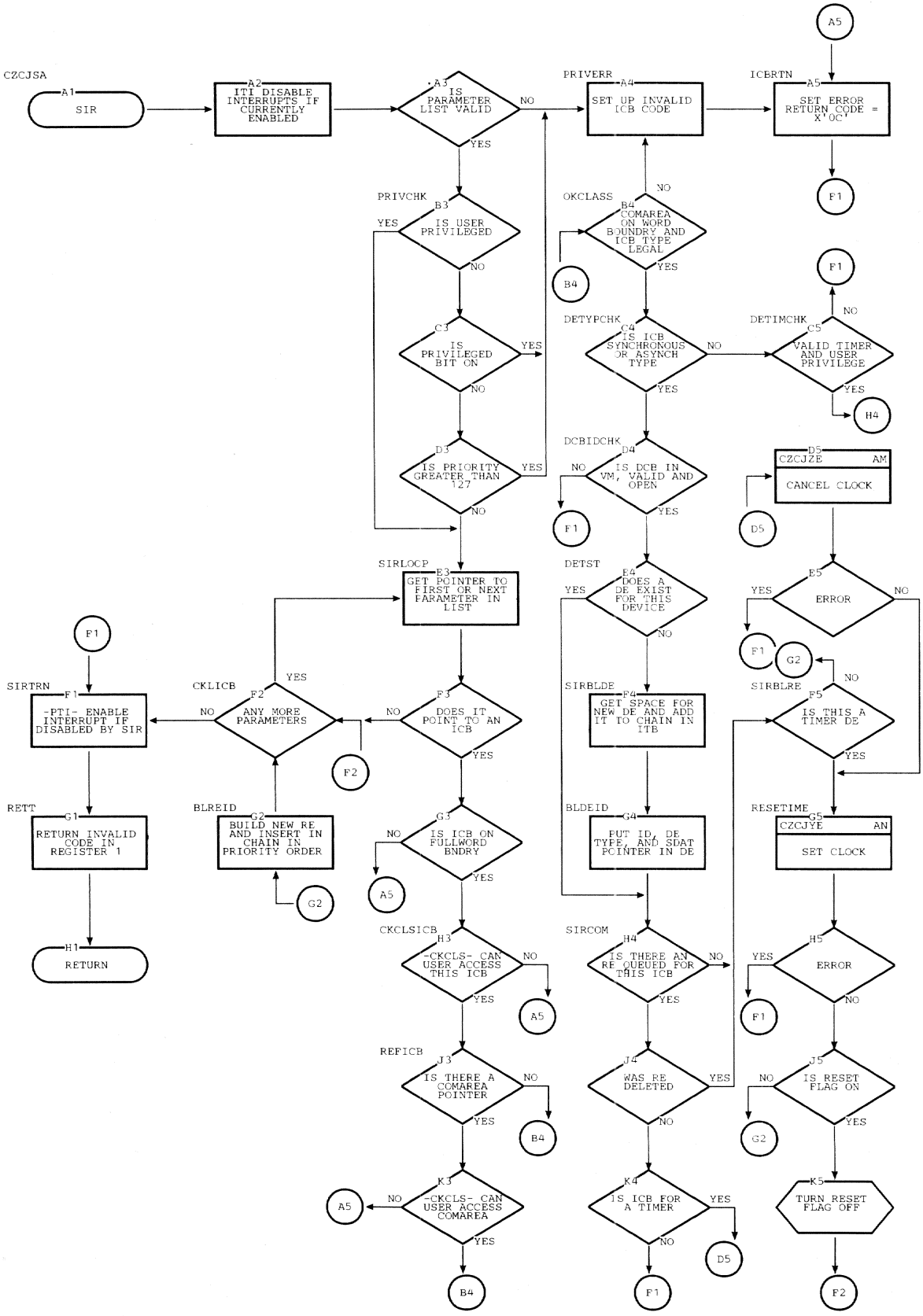


Chart AJ. Delete Interruption Routine (CZCJD) (Part 1 of 2)

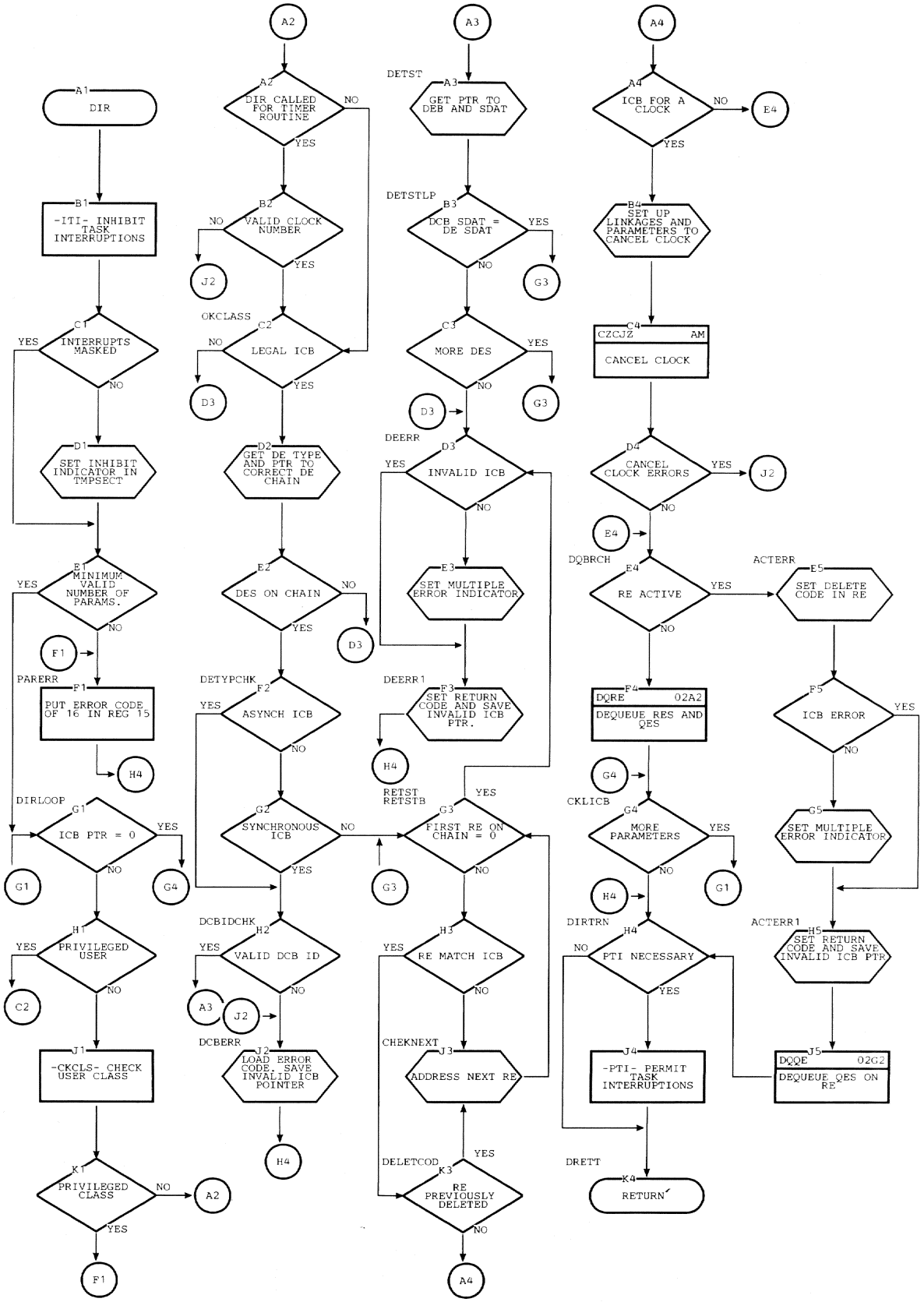


Chart AJ. Delete Interruption Routine (CZCJD) (Part 2 of 2)

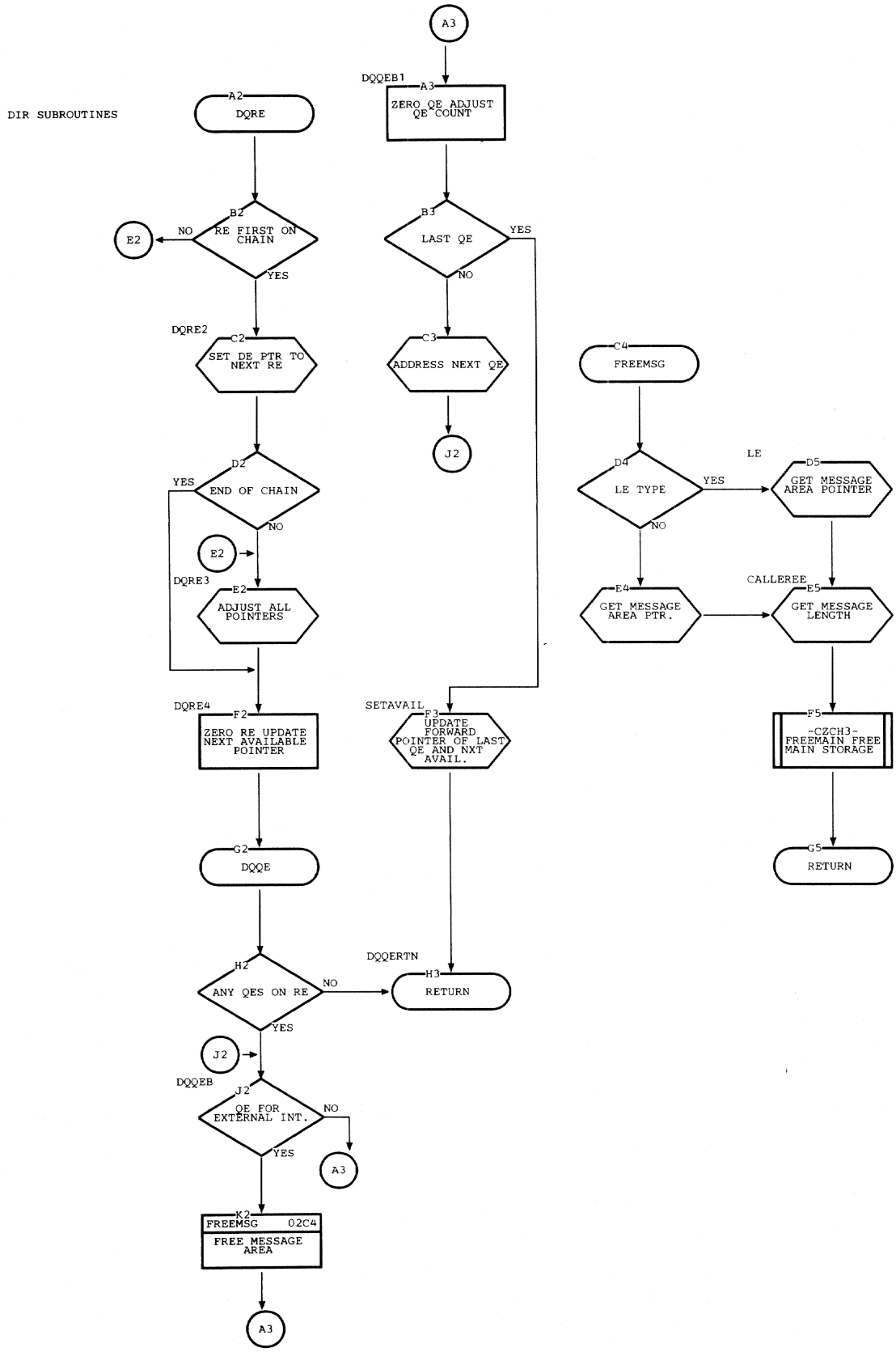


Chart AK. Interruption Inquiry Routine (CZCJI) (Part 1 of 3)

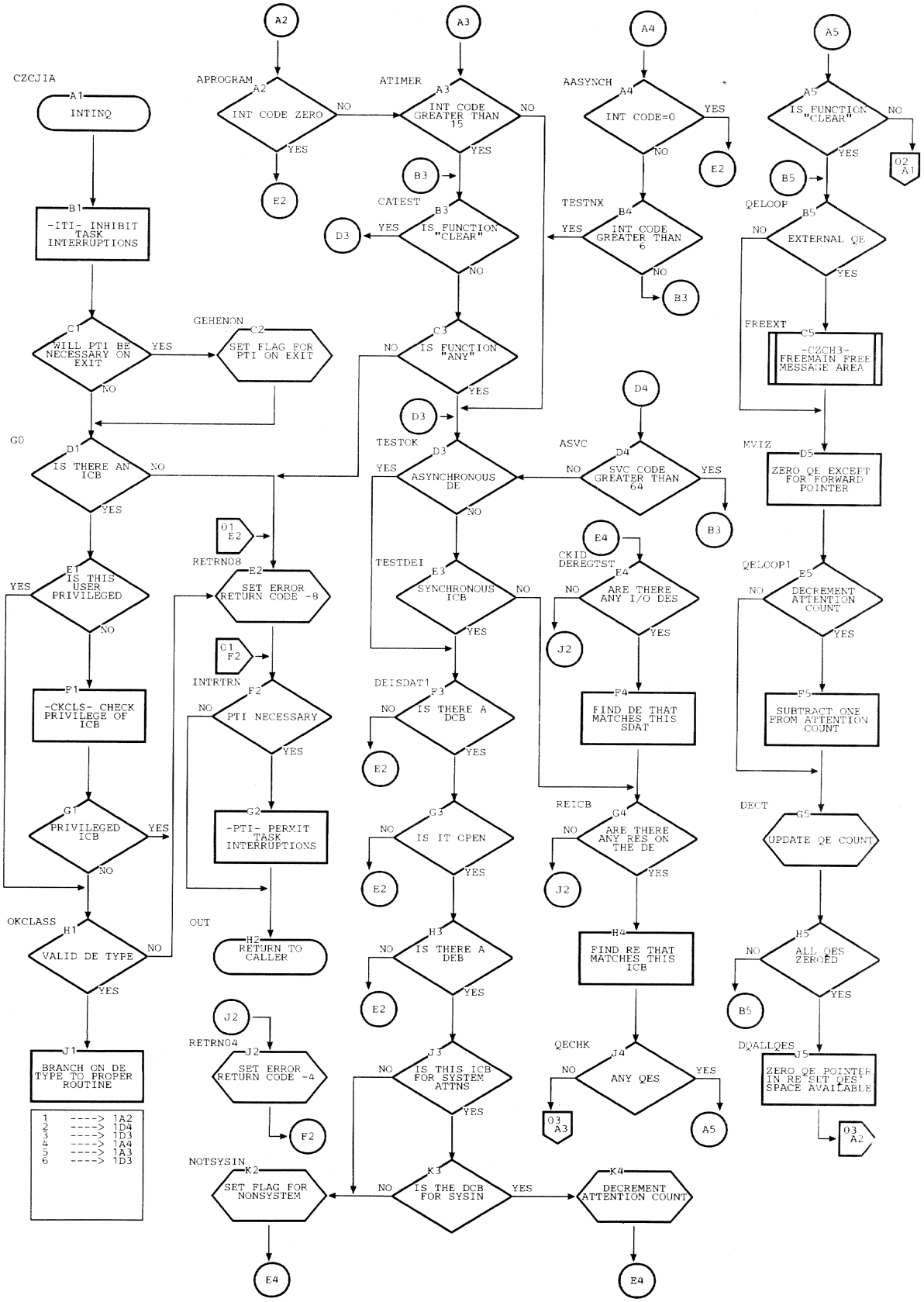


Chart AK. Interruption Inquiry Routine (CZCJI) (Part 2 of 3)

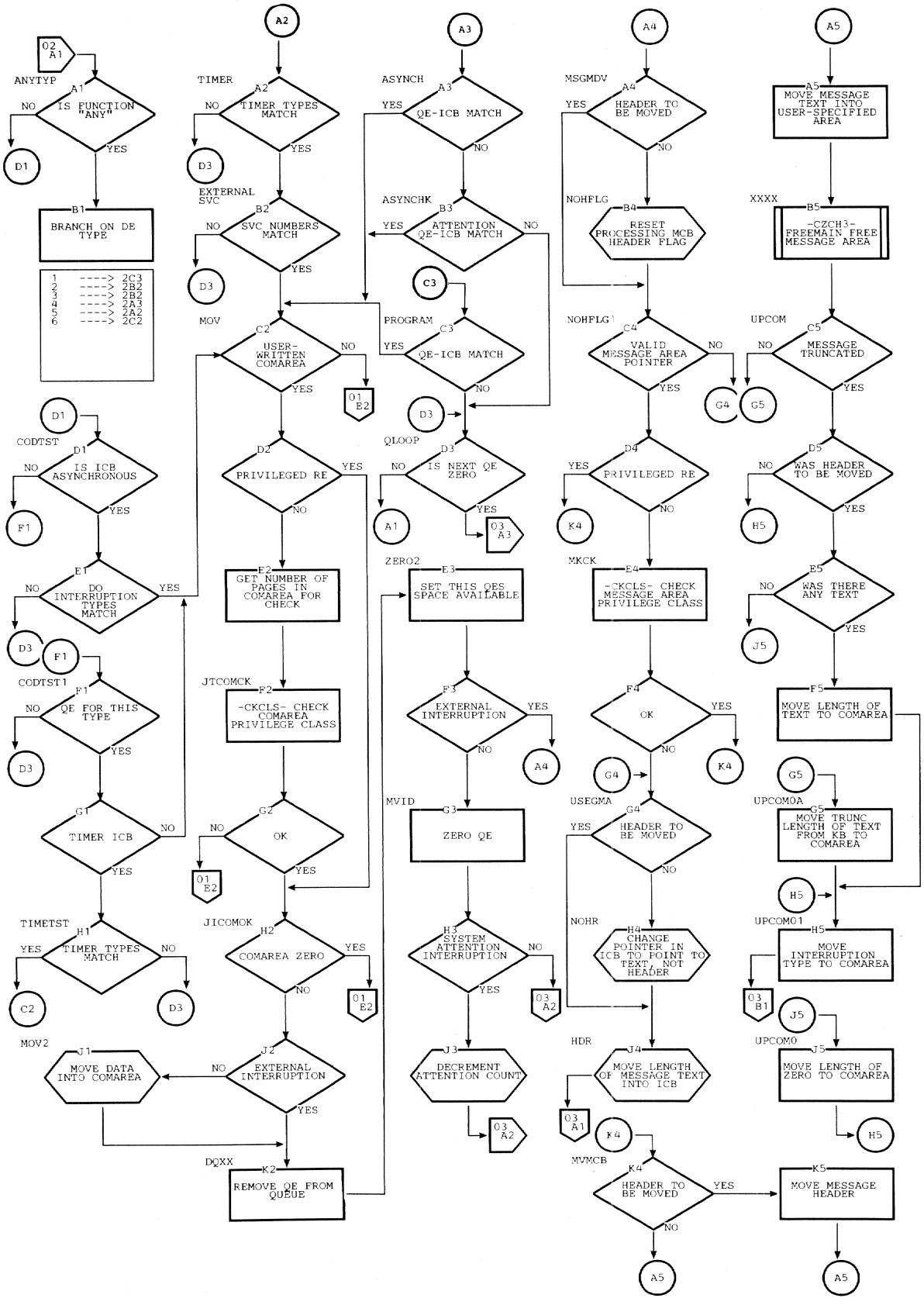


Chart AK. Interruption Inquiry Routine (CZCJI) (Part 3 of 3)

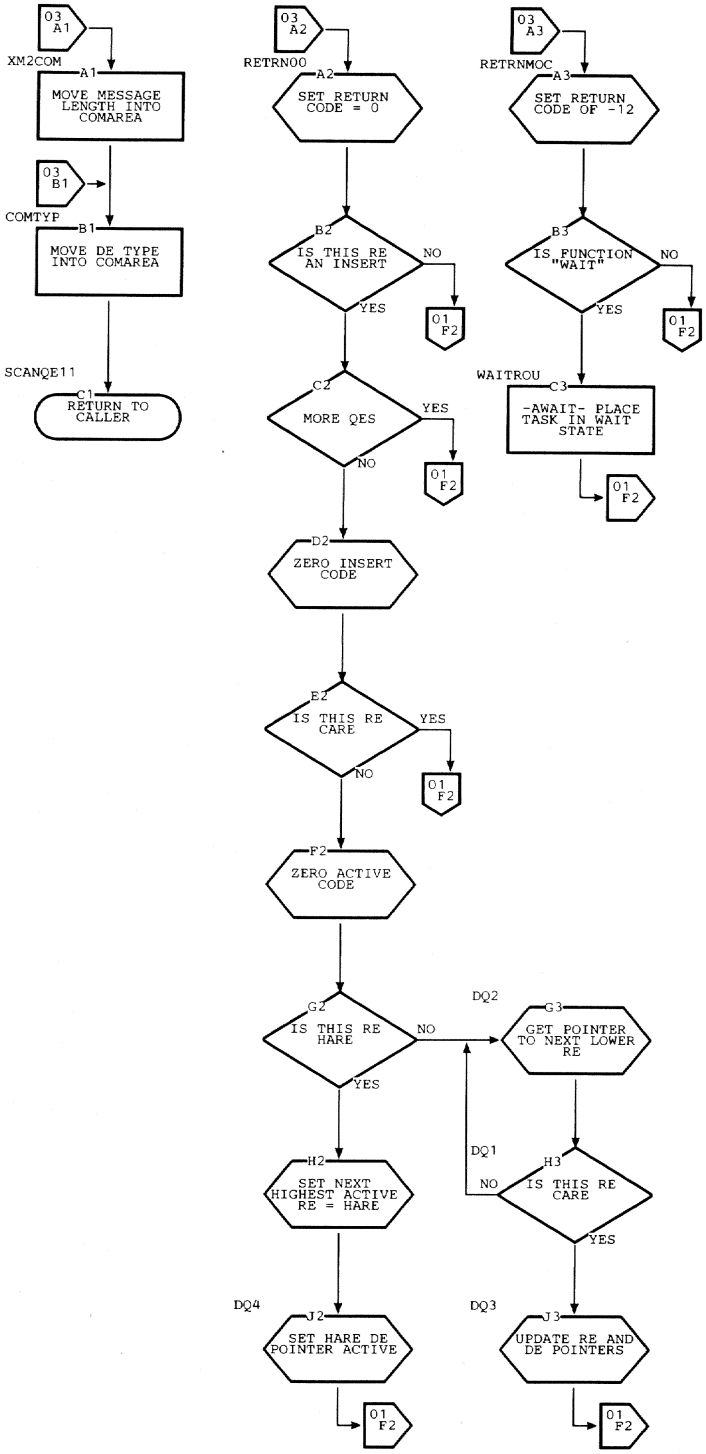


Chart AL. Set Timer/Test Timer (CZCJA)

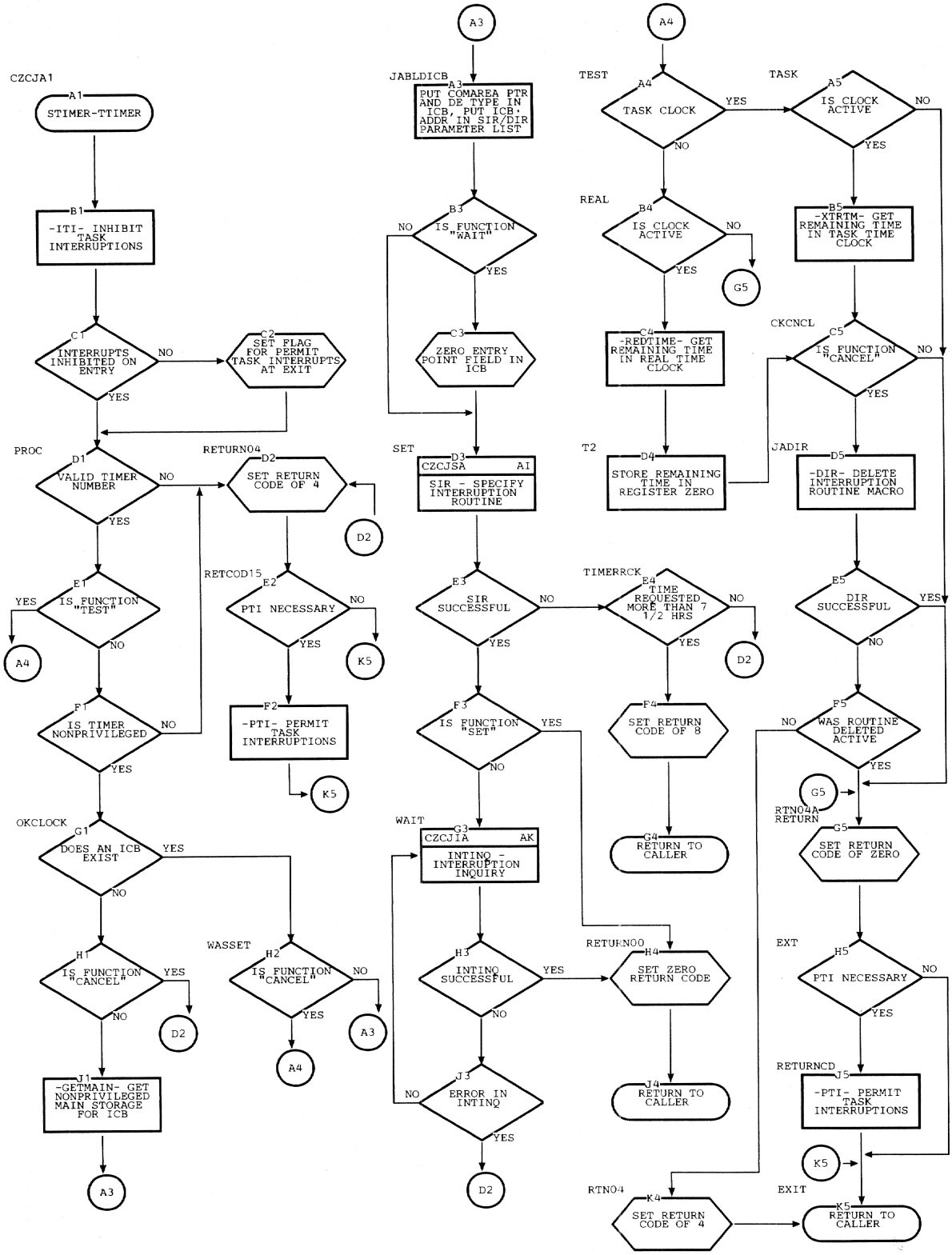


Chart AM. Cancel Clock Routine (CZCJZ)

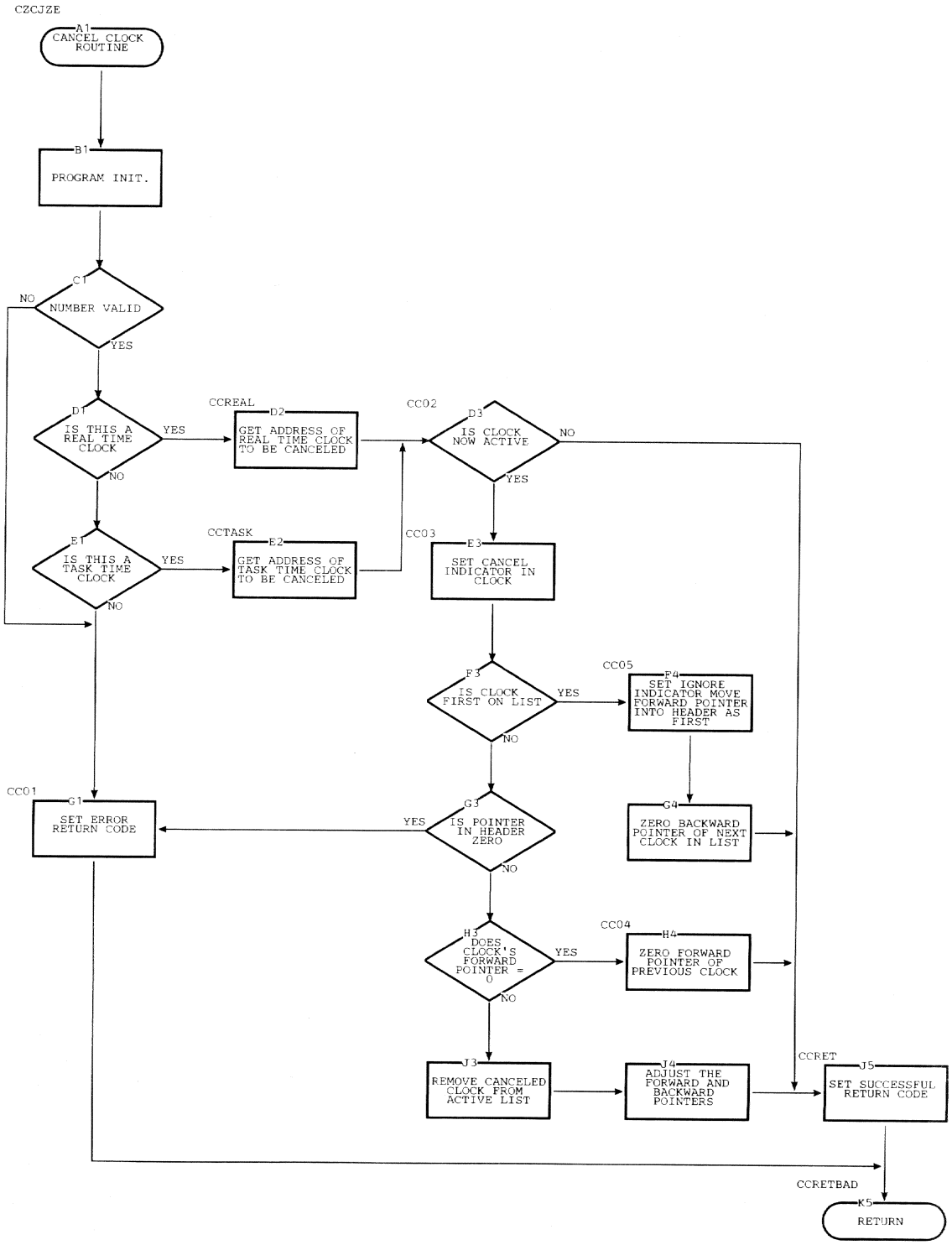


Chart AN. Set Clock Routine (CZCJY)

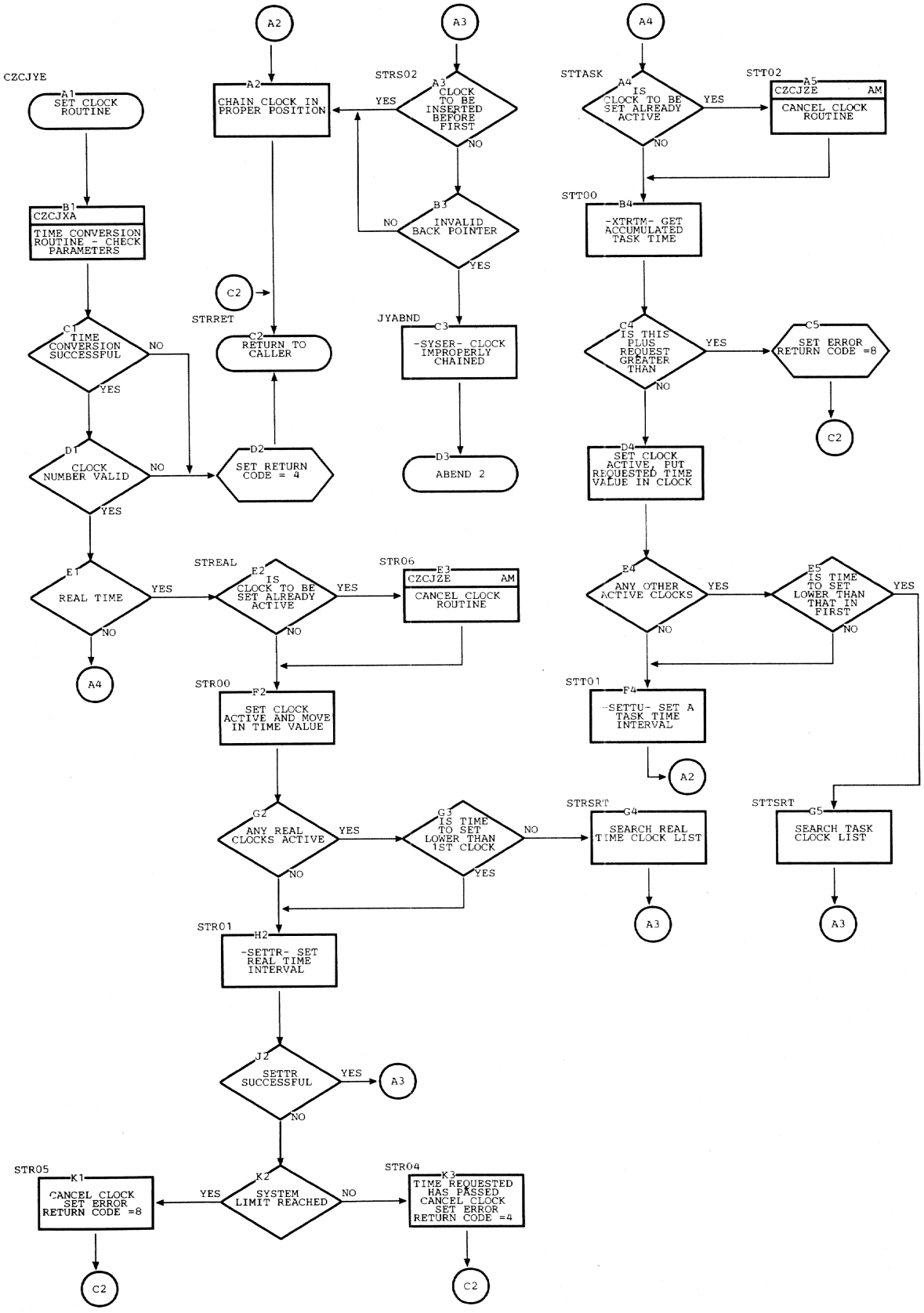


Chart AO. Leave Privilege Routine (CZCJL)

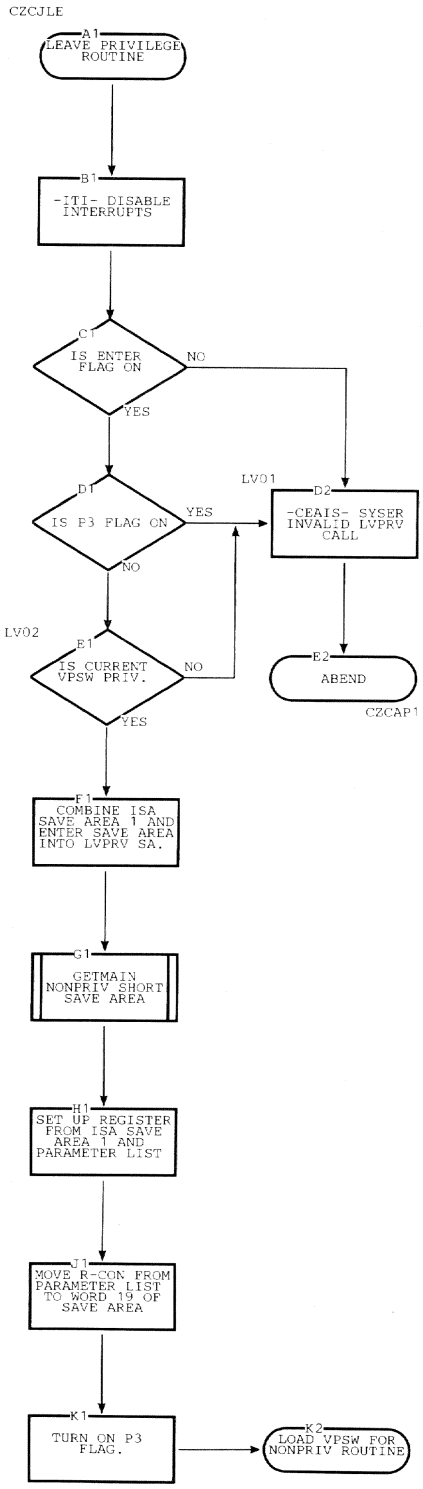
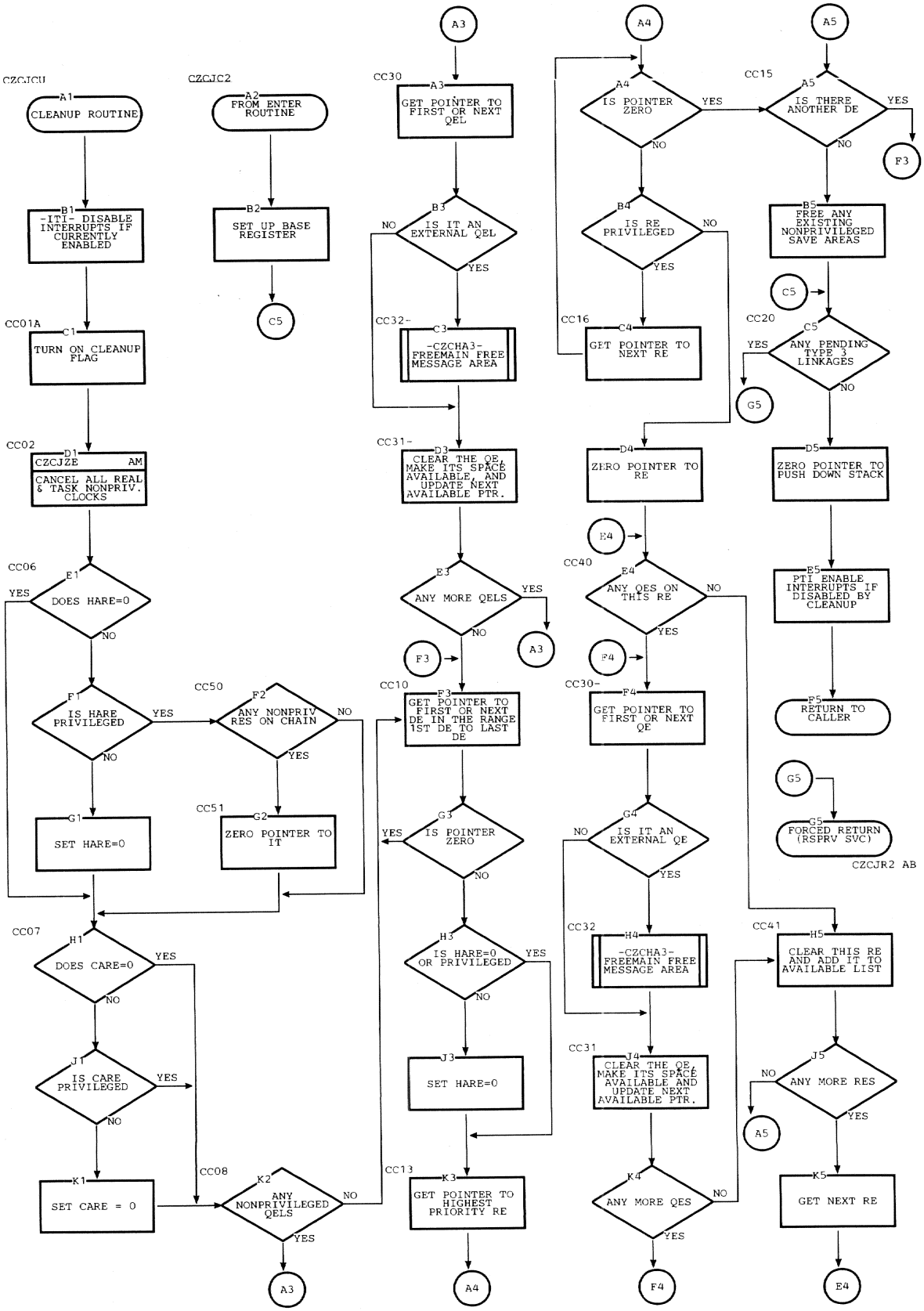


Chart AP. Cleanup Routine (CZCJC)



DATA AREAS

The task monitor uses tables for the storage, recovery, and transfer of information. Each control block is used during particular phases of task monitor operation, depending upon the kind of information required during each phase (see Figure 8). System control blocks reside in virtual storage, and each task has its own copy of those it uses. For the layout and exact contents of each system control block (CHAXXX) mentioned here, see System Control Blocks, GY28-2011.

CHAISA: The interruption storage area holds the status of a particular interruption. It contains the old VPSW and the new PSW used by task interruption control to give control to the correct interruption processor. The interruption processors perform register saves in the ISA, and reference it for information on VSS operation, interruption and SVC codes, privilege state, and certain special conditions.

CHAITB: The task monitor interruption table, shown in Figure 4, contains the queues that allow the task monitor to process interruptions by priority. The ITB is referenced by QLE, which builds QEs and queues them on the appropriate RE for the appropriate DE for that interruption type. The scanner/dispatcher references and dequeues QEs and REs upon dispatch to the specified routine for the interruption. SIR uses information from the interruption control block to create an RE if necessary, or initialize it if it has already been created. DIR dequeues and clears the RE and QE associated with a particular ICB. INTINQ references the ITB in order to determine the status of an interruption, and in "CLEAR" mode, zeros and dequeues all QEs on the RE. Cleanup clears and dequeues all nonprivileged REs and QEs.

CHAIDE: Interruption device entries are chained to each other, and contain pointers to the highest priority and highest priority active RE queued on them.

CHAIRE: The interruption request entries on a DE are chained to one another. Each contains a code for an L-type RE and an activity indicator. They also contain a priority number, a privilege flag, and pointers to the ICB, the first QE, and to the task monitor's pushdown save area.

CHAIQE: The interruption queue entries on a particular RE are chained together. The QE contains a code for L-type, and an interruption type. The rest of the fields contain information peculiar to that type of interruption.

CHAICB: The interruption control block contains the information necessary to interruption handling routines. It is used by QLE, the scanner/dispatcher, SIR, DIR, and INTINQ. It contains a pointer to the COMAREA, and for I/O interruptions, a pointer to the DCB. It also contains the entry point R- and V-cons for the interruption handling routines. There are special ICB fields for each interruption type.

CHADCB: The data control block controls I/O operations and is used only for synchronous or asynchronous I/O interruptions. INTINQ, SIR, DIR, QLE, and the scanner/dispatcher refer to it.

CHADEB: The data extent block is referenced by the same task monitor routines that use the DCB. It too, is an I/O control block, containing information about the data set to be used as input or output, and also about the storage medium for the data set.

CHAIOR: The I/O request control block controls the transfer of data between main storage and virtual memory. It is referenced by the synchronous I/O interruption processor for the address of the appropriate I/O posting routine.

CHAMCB: The message control block is used by the task monitor external interruption processor to implement inter-task communications. It contains a message text, a 'reply expected' flag, and a VSEND SVC.

CHACOM: The communications area, or COMAREA, contains basic interruption information transferred from the QE by the scanner/dispatcher. This information is referenced by INTINQ, DIR, and QLE. It consists of the interruption type and the interruption code within that type. For a program, SVC or timer interruption, the COMAREA also contains the address of the point of interruption. For an external interruption, it contains the address of a message area, and for a synchronous or asynchronous I/O interruption, it contains channel sense information.

CHASDA: The symbolic device allocation table is used to indicate the status of any device which has a DE in the ITB. It is used by QLE.

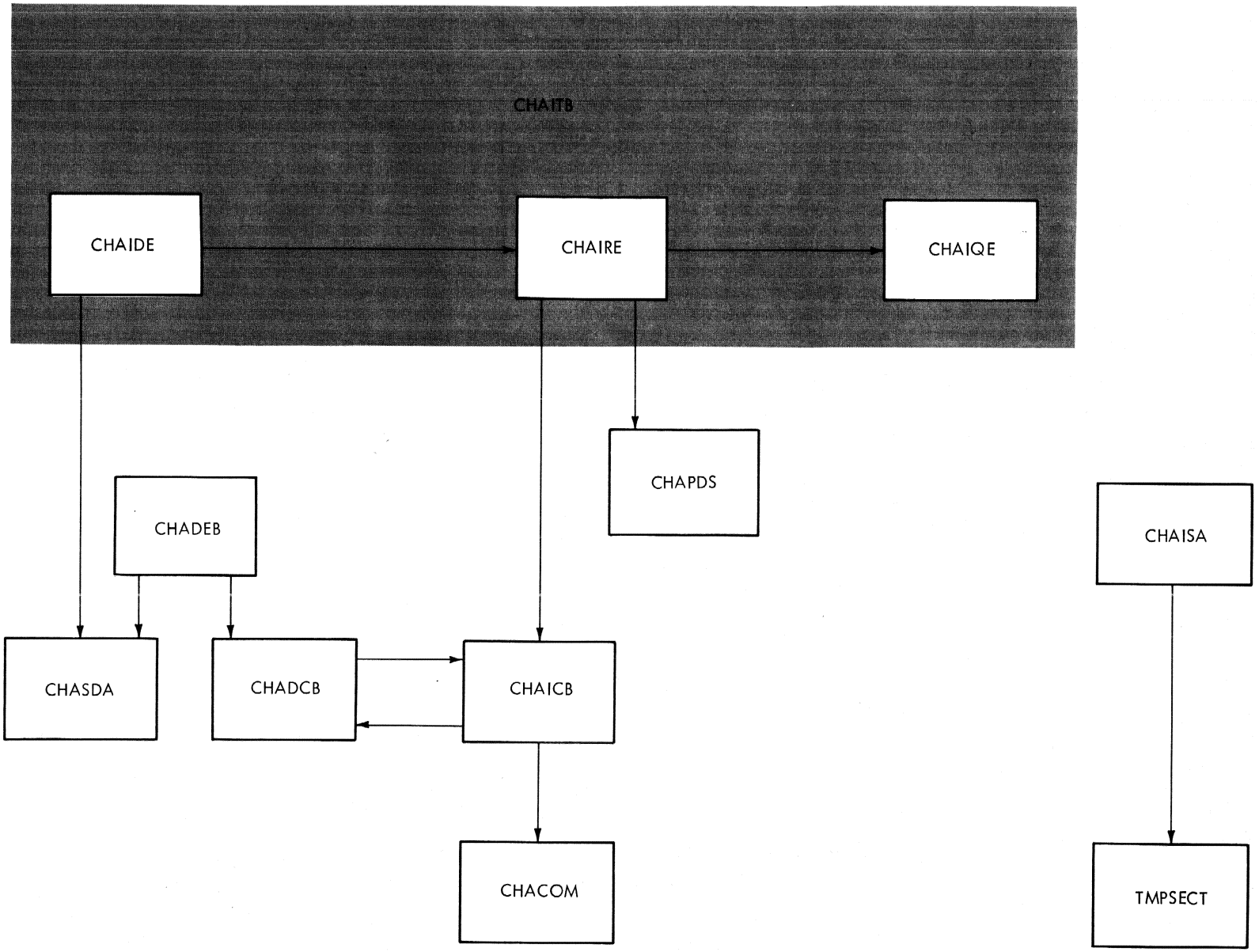
CHAPDS: The task monitor's pushdown save area consists of dynamically allocated storage used by the task monitor to save both task monitor and user status. If processing for one user must be interrupted for initial processing of another interruption, this save area allows resumption of processing for the first user with integrity. The fact that this storage is dynamically allocated makes task monitor code reenterable.

CHACLH and CHACLK: The task monitor clock list header and clocks. The clock list header contains the time value used in the SETTR or SETTU macro, and a pointer to the first active clock. A real or task clock contains an activity indicator, the real or task time already used, and the time requested.

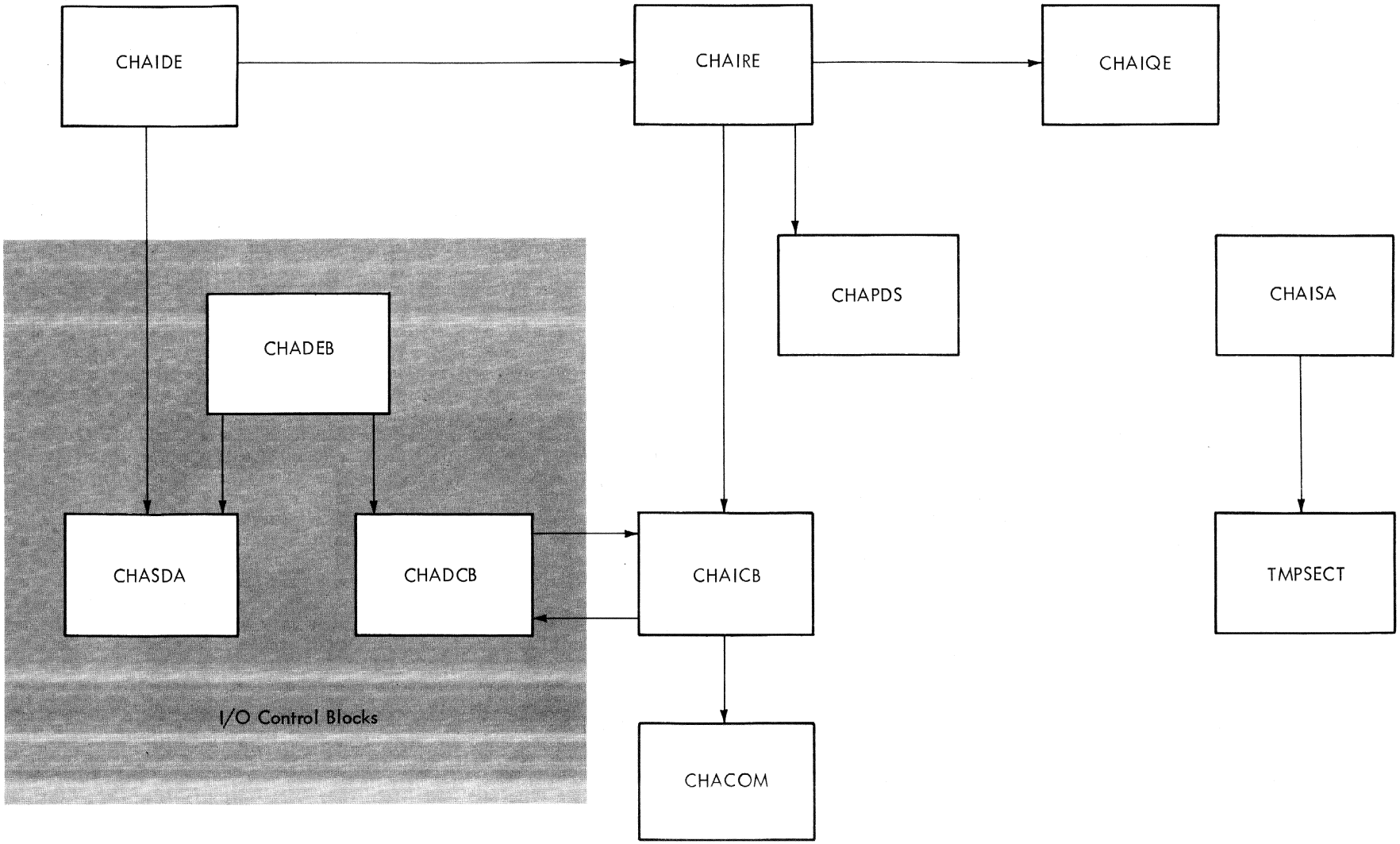
CHAET1 and CHAET2: The enter tables are used by the task monitor ENTER SVC routine. Table 1 points to table 2 and indicates the highest valid enter code. Table 2 contains the V-cons and R-cons necessary to transfer control from one routine to another.

TMPSECT: The task monitor's prototype control section contains several 19-word save areas for routines called or dispatched by the task monitor; for example, QLE, RSPRV, and the dynamic loader. It also contains R- and V-cons for the dynamic loader, virtual storage allocation, DIAGNO, PCS, and parts of the command system. The PSECT contains the 16 task and 16 real clocks and the task monitor's log, which contains a record of the last fifty task interruptions. A more complete description of the TMPSECT fields can be found in the "Diagnostic Aids" section of this book.

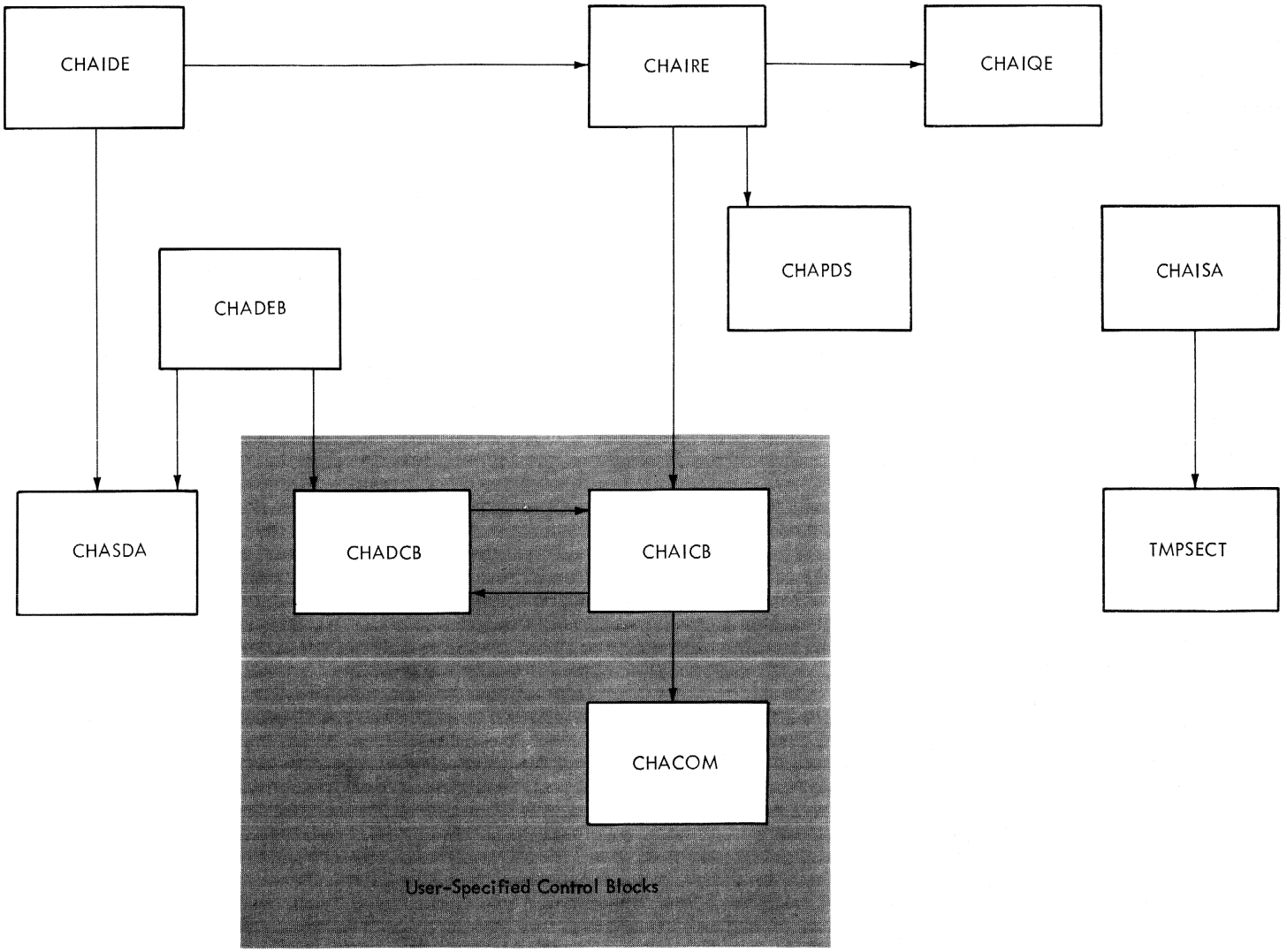
CONTROL BLOCKS REFERENCED BY THE TASK MONITOR (I): THE INTERRUPTION TABLE



CONTROL BLOCKS REFERENCED BY THE TASK MONITOR (II): I/O CONTROL BLOCKS



CONTROL BLOCKS REFERENCED BY THE TASK MONITOR (III): USER-SPECIFIED CONTROL BLOCKS



DIAGNOSTIC AIDS

DATA REFERENCE BY TASK MONITOR MODULES

This table lists the control blocks and work areas that are used by each module of the task monitor during processing.

Table 4. Data Reference by Task Monitor Modules

MODULE / Table, Workarea, etc.	Task Monitor	Scanner/ Dispatcher	QLE	SIR	DIR	INTINQ	STIMER-TTIMER	Time Conv.	Set Clock	Cancel Clock	Cleanup	LVPRV
Interrupt Storage Area (ISA)	C	C	C	C	C	C	--	--	--	--	C	C
I/O Record Control Block (IORCB)	R	--	--	--	--	--	--	--	--	--	--	--
Task Common (TCM)	R	--	--	--	--	--	--	--	--	--	--	--
Enter Table	R	--	--	--	--	--	--	--	--	--	--	--
Message Control Block (MCB)	R	--	R	--	R	--	--	--	--	--	--	--
Interrupt Table (ITB)	--	C	C	C	C	C	--	--	--	--	C	--
Data Extent Block (DEB)	--	--	R	R	R	R	--	--	--	--	--	--
Data Control Block (DCB)	--	--	R	R	R	R	--	--	--	--	--	--
Interrupt Control Block (ICB)	--	C	C	R	R	R	C	--	--	--	C	--
Device Entry (DE)	--	C	C	C	C	C	--	--	--	--	C	--
Request Entry (RE)	--	C	C	C	C	C	--	--	--	--	C	--
Queue Entry (QE)	--	C	C	--	C	C	--	--	--	--	C	--
Communications Area (COM)	--	C	R	--	--	C	R	--	--	--	C	--
Symbolic Device Table (SDAT)	--	--	R	--	--	--	--	--	--	--	--	--
Task Monitor Clocks (CLK)	C	--	--	--	--	--	R	--	C	C	C	--

R = Reference only; C = Change

Table 5. Microfiche Directory

EP Name	Name of Item	Module	CZECT or PSECT	Chart ID
CZCJA1	STIMER/TTIMER routine	CZCJA	CZCJAC	AN
CZCJCU	Cleanup routine	CZCJC	CZCJCC	AP
CZCJC2	Cleanup--EP 2	CZCJC	CZCJCC	AP
CZCJDA	DIR routine	CZCJD	CZCJDC	AJ
CZCJEM	Program interruption processor	CZCJT	CZCJTM	AA
CZCJEN	Last instruction in CSECT	CZCJT	CZCJTM	AG
CZCJER	SVC interruption processor (enter routine)	CZCJT	CZCJTM	AB
CZCJIA	INTINQ routine	CZCJI	CZCJIC	AK
CZCJLE	Leave privilege routine	CZCJL	CZCJLV	AO
CZCJQS	QLE routine	CZCJT	CZCJTM	AG
CZCJR2	SVC interruption processor (restore privilege routine)	CZCJT	CZCTTM	AB
CZCJRC	Real time clocks	CZCJTT	CZCJPS	none
CZCJRH	Real time clock header	CZCJT	CZCJPS	none
CZCJSA	SIR routine	CZCJS	CZCJSC	AI
CZCJTA	Asynchronous interruption processor	CZCJT	CZCJTM	AE
CZCJTC	Task time clocks	CZCJT	CZCJPS	none
CZCJTH	Task time clock header	CZCJT	CZCJPS	none
CZCJTI	Synchronous I/O interruption processor	CZCJT	CZCJTM	AE
CZCJTL	Task log	CZCJT	CZCJPS	none
CZCJTP	Program interruption processor	CZCJT	CZCJTM	AA
CZCJTS	SVC interruption processor	CZCJT	CZCJTM	AB
CZCJTT	Timer interruption processor	CZCJT	CZCJTM	AC
CZCJTV	VSS interruption processor	CZCJT	CZCJTM	AF
CZCJTX	External interruption processor	CZCJT	CZCJTM	AD
CZCJTY	Data set paging interruption processor	CZCJT	CZCJTM	AF
CZCJXA	Time conversion routine	CZCJX	CZCJXX	none
CZCJYE	Set clock routine	CZCJY	CZCJYC	AL
CZCJZE	Cancel clock routine	CZCJZ	CZCJZC	AM
SCAN	Scanner/dispatcher routine	CZCJT	CZCJTM	AH
SOP	Scanner/dispatcher routine (standard optimal procedure subroutine)	CZCJT	CZCJTM	

Table 6. Task Monitor Prototype Control Section (Part 1 of 2)

At	Length	Description
PSNPSA	76B	Nonprivileged save area
PSSA1	76B	Privileged save area 1
PSSA2	76B	Privileged save area 2
PSSAID	76B	Implicit dynamic loader (relocation) save area
PSIDLS	120B	Implicit dynamic loader (relocation) long save area
PSSCLS	120B	Scanner long save area
PSSAQL	76B	Queue linkage entry save area
PSSASC	76B	Scanner save area
PSSARP	76B	Restore privilege (RSPRV) save area
PSSAED	76B	Explicit dynamic linkage save area
PSSADS	76B	Data set paging save area
PSRASA	140B	Short save areas for flags, return codes, pointers, masks, and VPSWS
PSSDS5	32B	Scanner short save area
PSSEP2	44B	Save area for scan at entry point 2
PSSDILE	8B	V- and R-con for DIAGNO
PSDLE	16B	Enter V- and R-con (SVC 116)
PSCDLF	16B	V- and R-con for CLIC (SVC 119)
PSCDLG	16B	V- and R-con for CLIP (SVC 118)
PSCDPL	8B	Queue index and information for privileged LE
PSCDPL1	24B	V- and R-con for RTRN (SVC 122)
PLQLE	16B	V- and R-con and parameter list for GATE routines
PSCMSW	12B	QLE information
GATEPL	28B	Pointers
PSDLA1	16B	V- and R-con and parameter list for the dynamic loader (relocation exception)
PSLCSA	12B	Dynamic loader parameters for DLINK SVC
PSDESA	12B	V- and R-con for DELETE SVC
PSGMBV	24B	GETMAIN and FREEMAIN V-cons, R-con, and constant
PSPCVC	8B	V- and R-con for PCS call
PSAIVC	8B	Logon V- and R-con
PSEXPC	8B	External interruption processor V- and R-con

Table 6. Task Monitor Prototype Control Section (Part 2 of 2)

At	Length	Description
PSACUV	12B	Cleanup V- and R-con
PSAHD3	8B	External interruption subprocessor V- and R-con
PSASCV	8B	Set clock V- and R-con
PSABENDV	8B	ABEND V- and R-con
PSADSV	8B	VMIER V- and R-con
		VSS Entry Points
PSAVPV	8B	CZHNPA--Program interruption processor
PSAVSV	8B	CZHNVB--SVC interruption processor
PSAVXV	8B	CZHNEA--External interruption processor
PSAVAV	8B	CZHNVC--Asynchronous I/O interruption processor
PSAVTV	8B	CZHNVD--Timer interruption processor
PSAVIV	8B	CZHSBB--Synchronous I/O interruption processor
PSAVRV	8B	CZHNVE--Data set paging interruption processor
PSAVSS	8B	CZHNVA--VSS interruption processor
PSLPL	28B	Parameter lists for QLE for each interruption type
TSCPQT	48B	Queue index numbers for QEs and QELs
PSAITB	104B	Task monitor internal adcons--save areas, scan subroutines, QLE subroutines, and tables
PSACPL	19B	Parameter list for setting task clock 15 on attention interruptions with the P3 flag on
MESS251	102B	Messages and message lengths
CZCJTB	16B	Header for real time clocks
CZCJTD	384B	Real time clocks
CZCJTH	16B	Header for task time clocks
CZCJTC	384B	Task time clocks
TMSVC	48B	Entry points within SVC processor
PRIVSAV1	256B	Privileged PDSA
IDLVSW	5B	Three flags and two flag save areas
CZCJTL	162B	Task interruptions log

Table 7. System Enter Codes (Part 1 of 2)

Decimal	Hex	Name	Entry Point	PSECT
		TAM		
0	0	READ/WRITE	CZCYM1	CZCYMP
1	1	BATCH MONITOR	CZABAE	CZABAE
		INTERRUPT HANDLING		
16	10	SIR	CZCJSA	CZCJSP
17	11	DIR	CZCJDA	CZCJDP
18	12	INTINQ	CZCJIA	CZCJIP
19	13	STIMER/TTIMER	CZCJA1	CZCJAR
		SAM		
32	20	READ/WRITE	CZCRAS	CZCRAP
33	21	CHECK	CZCRCS	CZCRCP
34	22	CNTRL	CZCRBS	CZCRBP
36	24	POINT	CZCRMA	CZCRMP
37	25	BSP	CZCRGA	CZCRGP
		VM ALLOCATION		
48	30	GETMAIN (R)	CZCH2	CZCH5
49	31	GETMAIN (PAGE)	CZCG2	CZCG5
50	32	FREEMAIN (R)	CZCH3	CZCH5
51	33	FREEMAIN (PAGE)	CZCG3	CZCG5
		VAM GENERAL SERVICES		
56	38	VDMEP	CZCQK1	CZCAKP
57	39	DUPOPEN	CZCQK1	CZCQKP
58	3A	DUPCLOSE	CZCEY1	CZCEYP
		VISAM		
61	3D	VISAM SETL	CZCPC3	CZCPC3
		VAM		
62	3E	VSAM PUT	CZCOS3	CZCOS3
63	3F	LIBESRCH	CZCDL3	CZCDLP
64	40	READ/WRITE	CZCPE1	CZCPEP
65	41	ESETL	CZCPD1	CZCPIP
66	42	RELEX	CZCPG1	CZCPIP
67	43	DELREC	CZCPH1	CZCPEP
68	44	FIND	CZCOJ1	CZCOJP
69	45	STOW	CZCOK1	CZCOKP
70	46	ADE	CZCPL1	CZCPLP
71	47	GETPAGE	CZCPI1	CZCPIP
72	48	INSPACE	CZCOD1	CECODP
73	49	DELPAGE	CZCOD2	CZCODP
74	4A	VSAM PUT EXTERNAL USER	CZCOS1	CZCOSP
75	4B	VSAM PUT INTERNAL	CZCOS2	CZCOSP
76	4C	MOVEPAGE	CZCOC1	CZCOCP
77	4D	FLUSHBUF	CZCOV1	CZCOVP
78	4E	VISAM GET PAGE INPUT	CZCPI2	CZCPIP
79	4F	VISAM GET PAGE OUTPUT	CZCPI3	CZCPIP
		MACRO COMMAND LANGUAGE		
80	50	GATRD/GATWR	CZATC2	CZATCP
81	51	WTO	CZABQ1	CZABQR
82	52	WTOR	CZABQ1	CZABQR
83	53	ERASE	CZAEJ7	CZAEJR

Table 7. System Enter Codes (Part 2 of 2)

Decimal	Hex	Name	Entry Point	PSECT
		MACRO COMMAND LANGUAGE		
84	54	DATADEF	CZAEA3	CZAEAR
85	55	DDCALL	CZAFS2	CZAFSR
86	56	ABEND	CZACP1	CZACPR
87	57	CARD	CZABD7	CZABDR
88	58	TAPE	CZABD9	CZABDR
89	59	LIST	CZABD3	CZABDR
90	5A	CATALOG	CZAEI2	CZAEIR
91	5B	UNCATLG	CZAEJ5	CZAEJR
92	5C	DSCOPY	CZAFV2	CZAFVR
94	5E	WTL	CZABQ1	CZABQR
95	5F	USATT	CZASA6	CZASAP
96	60	FINDJFCB	CZAEB1	CZAEBR
97	61	CLATT	CZASA7	CZASAP
98	62	RELEASE	CZAFJ2	CZAFJR
99	63	USAGE	CZAGB1	CZAGP
100	64	FINDDS	CZAEC1	CZAECR
101	65	MSGWR	CZAAD3	CZAADR
102	66	UPDTUSER	CZAGC2	CZAGCR
		GENERAL SERVICES		
112	70	IOREQ	CZCSB1	CZCSBR
113	71	MSAM READ/WRITE	CZCMF1	CZCMFP
114	72	MSAM - SET UNIT RECORD	CZCMD1	CZCMDP
115	73	MSAM FINISH	CZCMH1	CZCMHP
128	80	OLTAM - DEV. ALLOC.	CZATG1	CZATGP
129	81	OLTAM - EX. I/O	CZATA1	CZATAP
130	82	OLTAM - POSTING	CZATB1	CZATBP
131	83	OLTAM - TEST COMMAND	CZATS1	CZATSP
144	90	OPEN	CZCLAD	CZCLAB
145	91	CLOSE	CZCLBC	CZCLBP
146	92	FEOV	CZCLDF	CZCLDB
147	93	RFR	CZASD3	CZASDP
148	94	GDV	CZASDX	CZASDP
149	95	AETD	CZASB5	CZASBP
150	96	OBEY	CZASA4	CZASAP
151	97	MCAST	CZATU1	CZATUP
152	98	SYSIN	CZASC7	CZASCP
153	99	LPCINIT	CZASW1	CZAMZP
154	9A	LPCEDIT	CZASW4	CZAMZP
155	9B	PRMPT	CZATJ1	CZATJP
156	9C	ATTN	CZASB2	CZASBP
157	9D	GATE	CZATC2	CZATCP
158	9E	ENTRFR	CZASD5	CZASDP
159	9F	DELENT	CZASD6	CZASDP
160	A0	CSTORE	CZCKZ1	CZCKDP
161	A1	NXTRFR	CZASD4	CZASDP
162	A2	DICTIONARY HANDLER	CZASD2	CZASDP
191 254		RESERVED FOR TSS/360 USERS		

TASK MONITOR ENTER TABLES

Enter Tables 1 and 2 (CHAET1 & CHAET2)

Enter Tables 1 and 2 (ET1 and ET2) are private tables for the use of the task monitor ENTER SVC routine.

ET1 consists of one word for each possible enter code beginning with zero. For assigned enter codes the corresponding word in the table contains a pointer to an entry in ET2; for unassigned enter codes, the corresponding word contains all zeros.

ET2 contains an entry for each assigned enter code and can be accessed only by pointers in ET1.

Figures 12 and 13 illustrate ET1 and ET2, respectively.

Note: When enter code is invalid, the entry word contains all zeros; for a valid enter code, the entry word contains a pointer to ET2.

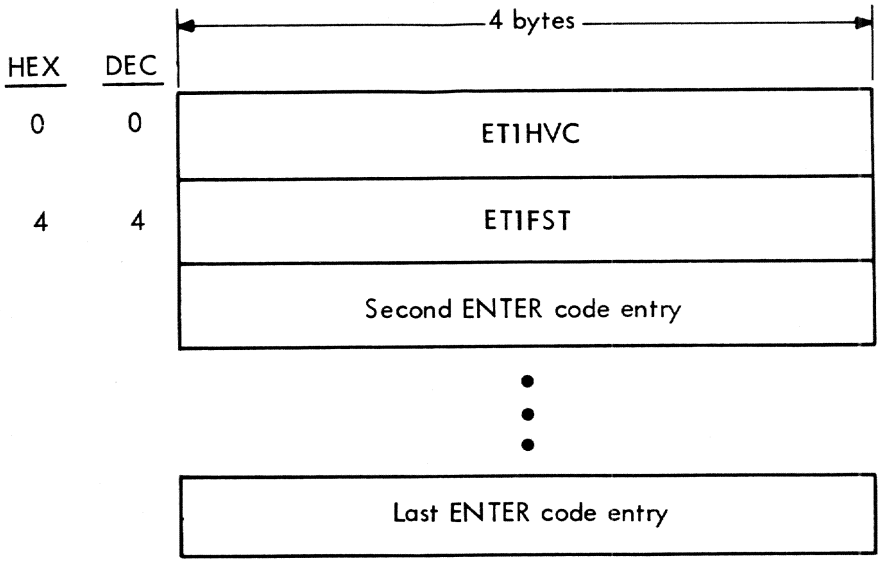


Figure 12. Enter Table 1 (CHAET1)

```
LOCATN  OBJECT CODE    ADDR1 ADDR2 STMNT   SOURCE STATEMENT

                                    0021          COPY CHAET1
                                    0022****                  ***
                                    0023*         HAET1
                                    0024*CHAET1   DSECT
                                    0025******          TASK MONITOR ENTER TABLE 1 --  CONTAINS POINTERS
                                    0026******          TO ENTRIES IN ENTER TABLE 2
000000                              0027*         DS    0F
000000                              0028*ET1HVC   DS    F HIGHEST VALID ENTER CODE
000004                              0029*ET1FST   DS    F ADDRESS OF 1ST ENTER CODE
                                    0030*         CSECT
```

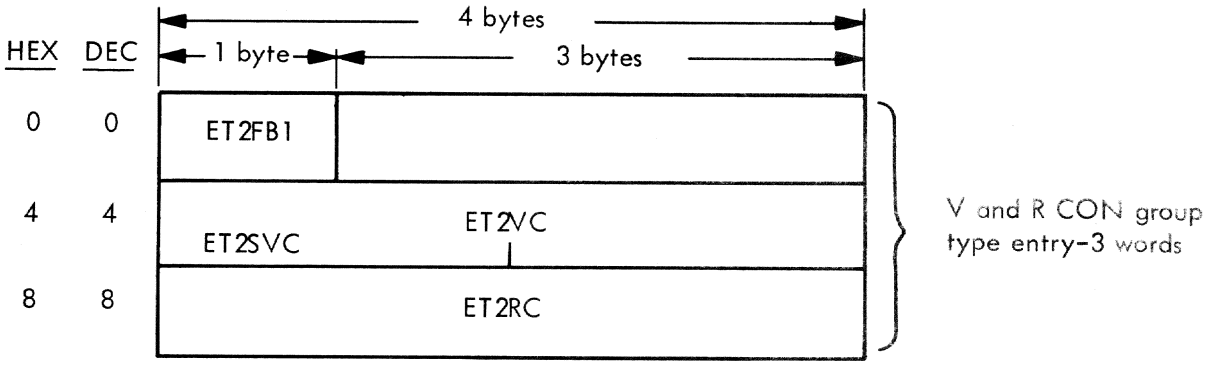


Figure 13. Enter Table 2 (CHAET2)

```
LOCATN  OBJECT CODE    ADDR1 ADDR2 STMNT   SOURCE STATEMENT

                                   0032          COPY CHAET2
                                   0033****                  ***
                                   0034*         HAET2
                                   0035*CHAET2   DSECT
                                   0036******              TASK MONITOR ENTER TABLE 2
000000                             0037*         DS    0F
000000                             0038*ET2FB1   DS    XL1 INDICATOR WORD  1ST BYTE
000000                             0039*ET2TY    EQU   ET2FB1 TYPE 0 - V AND R CON
000080                             0040*ET2TYM   EQU   X'80' 1 - ADCON GROUP
000000                             0041*ET2IS    EQU   ET2FB1 INT. STATUS 0 - NOT INT.
000040                             0042*ET2ISM   EQU   X'40' 1 - INTERRUPTABLE
000000                             0043*ET2P1    EQU   ET2FB1 P1 SETTING  0 - SET P1 OFF
000020                             0044*ET2P1M   EQU   X'20' 1 - SET P1 ON
000001                             0045*         DS    XL3 NOT USED
000004                             0046*ET2VC    DS    F V CON
000008                             0047*ET2RC    DS    F R CON
000004                             0048*         ORG   ET2VC ADCON GROUP
000004                             0049*ET2SVC   DS    H BEGINNING OF ADCON GROUP
                                   0050****      TO ACCESS THE ADCON GROUP, THE ADCOND GROUP IS USED
                                   0051*         CSECT
```

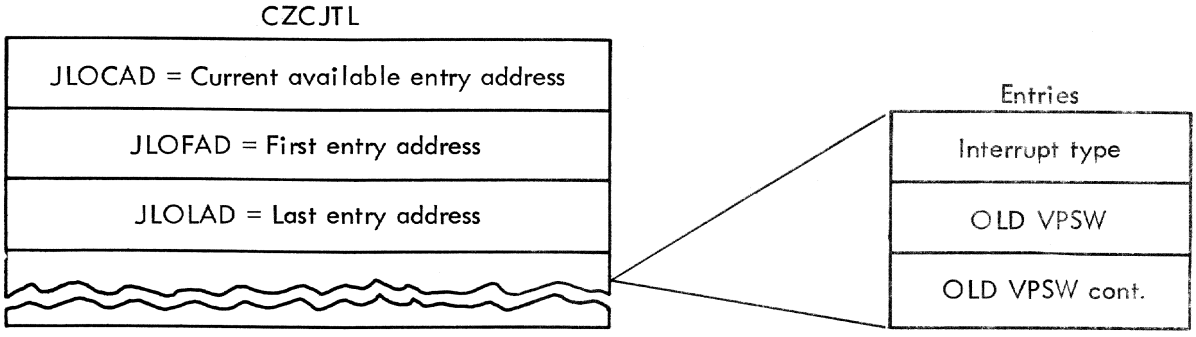


Figure 14. Task Monitor Interruption Log

Where more than one page reference is given, the major reference is first.

Task Monitor Printed in U.S.A. GY28-2041-4

International Business Machines Corporation
Data Processing Division
112 East Post Road, White Plains, N.Y. 10601
(USA Only)

IBM World Trade Corporation
821 United Nations Plaza, New York, New York 10017
(International)