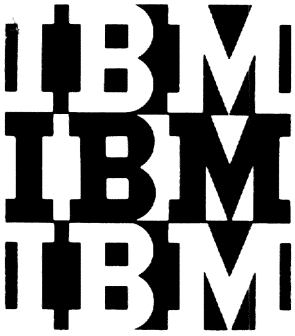


File No. S360-48
Form Y28-2040-0

Program Logic

IBM System/360 Time Sharing System

Support for Time Sharing

Program Logic Manual

This publication discusses operation of the Support for Time Sharing (STS) modules that are included in IBM System/360 Time Sharing System to support and facilitate development of TSS/360.

This program logic manual is directed to the IBM customer engineer who is responsible for program maintenance. It can be used to locate specific areas of the program, and it enables the reader to relate these areas to the corresponding program listings. Because program logic information is not necessary for program operation and use, distribution of this manual is restricted to persons with program-maintenance responsibilities.

PREFACE

This publication is divided into an introduction and a modular description of STS components. It is organized to give the reader a clear understanding of the interfaces between components, as well as their output.

There are two appendixes: "Appendix A: Flowcharts," and "Appendix B: Layout of STS Phases," that depict the composition of the components making up the STS phases.

A related publication, IBM System/360 Time Sharing System: Support for Time Sharing, Form C28-2034, provides a detailed explanation of STS commands and is a prerequisite for this Program Logic Manual.

The reader should also be familiar with

IBM System/360 Principles of Operation, Form A22-6821.

First Edition (October, 1967)

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

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 561, 2651 Strang Blvd., Yorktown Heights, N. Y. 10598

TABLE OF CONTENTS

INTRODUCTION

STS provides a software interface between the 360/67 hardware and the TSS resident supervisor as a serviceability aid for TSS/360. All interrupts, except machine check interrupts, are intercepted by STS and processed or passed on to the TSS resident supervisor. The trapping of interrupts by STS is accomplished by modifying the new PSWs set up by STARTUP to point within the STS supervisor (excluding the machine check PSW).

When the IPL is performed from the system residence pack, STARTUP is invoked and link-loads the TSS Initial virtual memory (IVM) and real core system programs. Real core modules are link-loaded by STARTUP from the TSS*****.RESSUP.G0000V00 data sets along with the STS supervisor (CSFSU) and exerciser (CSFXR). STARTUP reads the first page of the CSFSU module and locates the external new PSW and the address of the PUB table. STARTUP then fills in the addresses of SYSRES, SYSLOG, and SYSLST in the PUB table and saves the external new PSW for subsequent linkage to the STS supervisor.

The first page (page zero) of the STS supervisor CSFSU is the PSA page and is not loaded by STARTUP. It is merely used to locate the pointers to the PUB table and external new PSW. The remainder of the STS supervisor is loaded beginning at core location X'1000'. STARTUP reserves core position X'1000' to X'6FFF' for STS use. The exerciser (CSFXR) is loaded with the remainder of the real core modules in locations above X'6FFF'.

Once STARTUP has link-loaded real core, IVM, and the STS supervisor and exerciser, it transfers control to four bytes less than the address contained in the new external PSW. This activates the initializer functions of the STS supervisor. The initializer sets up a special entry to TSS by modifying the current PSW in the user save area of the STS communication region (REGON). It then fetches the command recognition routine (CRR-CSFCD). STS modules other than CSFSU and CSFXR are contained in a VSAM data set on the system residence pack called TSS*****.SYSSTS.G0000V00. The first page of this data set is a directory used by STS in locating the "phase" to be "fetched". Phase refers to a collection of STS modules link-edited in core-image form (see Appendix A). Fetch refers to the looking up of the location of a module in the directory page and the loading of that module into core from the TSS*****.SYSSTS.G0000V00 data set, with control being passed to the entry point of the loaded module.

Once the initializer phase of the STS supervisor fetches CSFCD, a question mark appears on the console typewriter. The operator should ASN SYSIPT to the card reader and perform an ASN CARDS to set up the PUB table. The PUB table is a table that defines various devices to STS. STS does not use any of the TSS device tables constructed by SYSGEN.

After the ASN CARDS, the operator can invoke other STS functions or simply issue a naked run. This causes STS to restore the machine to its status prior to entry to STS and to perform an LPSW from the PSW saved in REGON previously set by the initializer portion of CSFSU to point to the real core location defined within CSFXR by the entry name START.

START requests STS to fetch LINK (CSFLN), which completes the connection across the STS-TSS interface. START exits to the queue scanner, and TSS is running.

Once TSS is running, a user may reactivate STS by depressing the interrupt key on the CPU console. This sets a switch in STS that tells the STS I/O interrupt processor that the next attention interrupt from the operator's 1052-7 console should be intercepted by STS and cause STS to take control of the system until the next RUN command is issued. When the RUN command is issued, the switch that was set by depressing the interrupt key is reset, and TSS is reactivated and receives future interrupts from the 1052-7 until a subsequent depression of the interrupt key.

Although it appears that TSS is in control, STS remains under the system, intercepting all interrupts subsequently passing them on to TSS/360. All I/O interrupts fielded by STS are maintained in an STS interrupt log which is similar to the TSS/360 interrupt log CEAJIL. This log's location can be determined from a dump or by subtracting 12 bytes from the address pointer in the I/O new PSW. Since STS intercepts all interrupts except machine check, a fundamental rule is, TSS will never modify the new PSWs.

The resident portions of STS are the supervisor (CSFSU), the exerciser (CSFXR), and the command recognition routine (CSFCD). An additional routine, Fetched by CRR on first entry or by ASN CARDS is CSFSA, also becomes resident within the X'6000' to X'6FFF' area of core. This routine provides a dump if a catastrophic error occurs in STS.

All other phases of STS are fetched into an area termed the transient area, as they are required. Usually only one transient phase can be fetched at one time. Upon completion of a requested operation, a transient routine exits to the CSFCD module, which produces a question mark and turns on the PROCEED light on the 1052-7 console.

Only two STS modules are contained within the TSS*****.-RESSUP.G0000V00 data set; however none of the STS modules may be assembled with the TSS assembler. Since STS modules are location dependent and size sensitive, none of the STS modules may be modified in any way.

The remainder of this PLM is concerned with module/phase description of STS and is intended to give the personnel maintaining TSS a feel for the function of each STS source module and how it affects the STS-TSS environment. Details concerning bit settings and fields used must be extracted from the source listing.

MODULE/PHASE DESCRIPTION

SUPPORT SUPERVISOR -- CSFSU

Function: The STS supervisor is divided into two basic phases -- initializer and interrupt processor. The initializer is entered via STARTUP and performs the one-time function necessary to effect the TSS/STS hookup. The interrupt processor section is interrupt driven and fields all interrupts except machine check interrupts and either processes them or passes them on to TSS (Charts AA-AE).

Entry:
SYSEXT1 - Entry to the initializer phase of CSFSU which is invoked by STARTUP
SYSSVC - Supervisor Call Interrupt Processor
SYSIPO - I/O Interrupt Processor
SYSEXTIO - External Interrupt Processor
SYSPRG - Program Interrupt Processor

Exit:
CSFXE - Exit to Exerciser during initialization phase.
LPSW - Interrupt processor exit issuing a LPSW at some points.

External References: None

Operation: The STS supervisor (CSFSU) is initially given control by STARTUP by transferring control to the A (external new PSW) -4 (SYSEXT1), which is the initializer portion of the supervisor. The initializer is subsequently overlaid by transient programs. The initializer saves the TSS PSWs set up by STARTUP and replaces them with STS PSWs that point within CSFSU (except the machine check new PSW). The timer is saved, with the CSW and CAW. The machine configuration is determined from the hardware and is stored in REGON, with the STS identification. The actual SYSRES address, including the CCU (channel control) bit, is determined by performing an SIO in standard mode and fielding the interrupt in extended mode and subsequently storing it in the SYSRES entry of the PUB table. The command recognition routine is fetched and becomes a resident portion of the supervisor, and a load of the external PSW is executed which was previously set to point to the START entry point within the exerciser CSFXR.

The resident portion of CSFSU begins at X'1000' and is entered via interrupts. All interrupts, except machine checks, are intercepted by STS and either processed by STS or passed on to TSS.

Interrupt Processors

The supervisor is divided into four sections, each of which processes a distinct interrupt type. An additional major subdivision of CSFSU is the physical IOCS package, which performs the I/O for CSFSU.

The four interrupt processors are serially reentrant, which implies that distinct interrupts can be processed in parallel. Upon entry to any of the interrupt processors, some common functions are performed. The CPU save area pointer is extracted from the PSA, and registers are saved within this area, which is a part of REGON. A switch in REGON is also tested to determine whether TSS or STS was the user at the time the interrupt occurred. If TSS was the user, the timer must be saved and restored, since STS performs its function in zero time.

SVC Interrupt Processor

The SVC processor decodes the SVC and takes appropriate action. If the SVC number is greater than 25 it is not a valid STS SVC and TSS is given the SVC. There is a table of SVCs in the SVC processor that indicates where to branch to process the SVC and which users can issue the SVC. Three particular users are of interest -- STS, TSS/360 resident monitor, and a TSS virtual memory task. There are currently only 11 SVC entries in this table. If TSS issues an SVC below 25, which is not valid for TSS, it is passed back to the interrupt stacker.

Program Interrupt Stacker

The program interrupt stacker processes program interrupts. Upon entry to this processor it checks whether the interrupt occurred during TSS or STS execution. If it occurred during TSS's execution, the interrupt is passed on to the TSS interrupt stacker; if it occurred during STS's execution, a test is made in REGON to determine if an STS routine has supplied its own program interrupt processor. If so, registers are restored, and an LPSW is issued to transfer control to this program; if a user routine was not specified, control is transferred to CSFSA, which will produce a stand-alone dump and place the system in wait state.

External Interrupt Processor

The interrupt processor fields two types of interrupts -- timer interrupts and interrupts generated by depressing the interrupt key on the CPU console. Timer interrupts are passed to TSS. An external interrupt generated by depressing the interrupt key causes a switch to be set in REGON which informs STS that the next attention interrupt from the 1052-7 is to be processed by STS. If the interrupt key is depressed while STS is executing, the interrupt is discarded.

I/O Interrupt Processor

On entry to the I/O interrupt processor, the interrupt is logged in the STS interrupt log. Pointers to the address of the next entry in the log, the beginning of the log, and the end of the log can be determined from three address constants which immediately precede the I/O interrupt processor. After logging the I/O interrupt, a branch is executed to the physical IOCS processor whose function it is to determine if this is an STS or TSS interrupt by examining I/O interrupt queues built by PIOS for I/O requests. This processor additionally tests for attention interrupts from the 1052-7 and determines if it is valid to process an attention interrupt at the time it occurs. Attention interrupts may not be valid (e.g., if one is received while a transient program is executing), since another transient program will be fetched and the currently executing program would be overlaid. The operator is warned if an attention is received at an inappropriate time. After the interrupt is processed, control is returned to the interrupted program.

Most support routines require that interrupts be enabled while they are executing. There are only two types of TSS interrupts that can occur while STS is executing -- external and I/O. Since TSS interrupts are not stackable within STS, they must be passed on to the interrupt stacker; however, STS requires immediate control from the interrupt stacker. This is accomplished in the following manner:

1. If the old PSW's relocate bit is not on, the old PSW is set to point back to STS so that when the interrupt stacker does an LPSW of the old PSW, control will be returned to STS.

2. If the old PSW's relocate bit is on, the address of the queue scanner in the interrupt stacker is modified to point back to STS so that when the interrupt stacker exits control will return to STS.

   This code is identical for processing both external and I/O TSS interrupts, with some minor exceptions. If the interrupt was generated with TSS the user, it is passed back to TSS via the interrupt stacker immediately.

If STS was entered when TSS was in a disabled state (this can only be effected by placing a DDR or CMP in the interrupt stacker or by a SYSER), and an external or I/O interrupt occurs, a stand-alone dump is taken by CSFSA, and the machine is dropped into wait state. The user must not insert DDR's or CMPs in TSS disabled code.

There is one other situation that causes a wait state condition and that is a simulated machine check. This is caused by the lack of a recovery procedure when certain unrecoverable errors occur and may be detected by depressing STOP and displaying the D register, which will be set to all ones.

STAND-ALONE DUMP -- CSFSA

Function: CSFSA is a stand-alone dump activated by STS only in the event of a catastrophic error within STS. It will output its dump onto a tape or printer (Chart AF).

Entry Point:
ZY2SAD - control is transferred here to dump core.
SYSAD2 - control is transferred here only when module is fetched to set up dump output device assignment.

Exit:
Normal: Fetch of SYSDIS to display PUB, at entry SYSAD2; Send message and AWAIT, at entry SYSAD1.
Error: LPSW 0 -- allows operator to restart with new device.

Modules Called: None

Operation: CSFSA is first entered at ZY2SAD2, via a fetch, to set a flag to indicate whether blocked tapes are to be produced, to set up a physical device address for use at dump time, to save control registers on a Model 67 for use at dump time, and to inform the operator of this action. The actual dump phase is entered at ZY2SAD. Here the wait bit is set on in prog new PSW (meaning a stand-alone card dump is necessary in case of a program interrupt while dumping). All registers are saved, as is low core, and addressability is established via the PSA. A channel program is established (CCWSETUP), and the dump is taken on a tape or printer. Main subroutines are at MAINDUMP (start build reference address), REFCALL (translates address, secures real core storage key), WRITE (calculates output line and BUMP data pointer), SAMELINE (omits printing of identical lines), and KEEPTRY N (starts dump output). Program interrupts due to discontiguous core terminate the dump if core addressability is lost.

CSFSA writes a message to the operator informing him the dump is complete, sets restart PSW to allow reentrance to CSFSA, and enters wait state.

COMMAND RECOGNITION ROUTINE (CRR) -- CSFCD

Function: CSFCD's function is twofold: (1) Examine the commands issued by the STS user and to Fetch the requested service. (2) Issue the proceed signal accompanied by a question mark (Chart AG).

Entry: The entry point to CSFCD is a SYSTRT at core location 3120, Entry is made by a Fetch of SYSTRT by STS Supervisor or via an SVC6 issued by STS modules.

Attributes: CSFCD is a resident routine occupying the area between Supervisor and transient routines.

Exit:

1. Fetch of CSFSA in the event of an error.
2. Fetch of appropriate routine. On exit register 3 will contain a pointer to the 1052-7 terminal input stream.

Modules Called:
CSFRK: to read the input from the 1052-7 terminal buffer.
CSFWK: to print out the diagnostic message for the error condition or the question mark (?) on a proceed condition.
CSFGF: Get the first three characters of input stream.

Operation: Upon entrance into CSFCD the contents of a byte called ERRCODE (located in REGON) is analyzed for a zero (=0) quantity. If the result is positive, the proceed condition will be issued. In response to the '?' the first three characters are concatenated with 'SYS___' and a fetch is performed on the resulting routine with a pointer in register 3 to the input stream. However, if the result is negative)≠0), indicating that the STS supervisor had detected some type of error condition, the contents are examined. If the value is greater than X'C0' a fetch is made of CSFSA for a stand-alone dump. Otherwise, the specific type of error is determined, using the value contained in ERRCODE as an indicator, and a diagnostic message is printed out with the PSW. CSFCD will then await action by the user.

Notes:

1. The first time through CSFCD, a fetch will be made on CSFSA to insure that it will be in core if needed.

2. Before link-editing, the cards INCLUDE SYSRDK, INCLUDE SYSGTK, and INCLUDE SYSGTL must be placed at the end of the object deck in the above order immediately before the entry card.

POOL ROUTINES

These routines are all separate entities. They are included in the same section because of the similarity in their function. They are:

CSFGF	GETFLD
CSFGL	GETLOC
CSFED	EDIT
CSFRK	RDKBD
CSFWK	WTKBD

Function: Although the names of the above routines carry functional descriptions (e.g., EDIT, GETFLD etc.), they are merely used to provide linkage to the routines that actually perform the functions (Chart AH).

Entry Point: The entry points for the POOL routines are as follows:

for CSFGF - GETFLD
CSFGL - GETLOC
CSFED - EDIT
CSFRK - RDKBD
CSFWK - WTKBD

Modules Called: The modules called by the above routines are as follows:

CSFGF calls CSFFG (GETFLD)
CSFGL calls CSFLG (GETLOC)
CSFED calls CSFEM (EDIT)
CSFRK calls READ entry of CSFKR (Read/Write Keyboard)
CSFWK calls WRITE entry of CSFKR

Exits: These routines branch to their associated routines by BR15. No return is provided to the routines.

Operations: Upon entry, the POOL routines pick up one of the five fields of the REGON area of the supervisor and BR15 to them.

SYSRDK - RDKBD (READ Keyboard)
SYSWTK - WRTKBD (WRITE Keyboard)
SYSEDI - EDIT (EDIT MESSAGE)
SYSGTF - GETFLD (GETFIELD)
SYSGTL - GETLOC (GET LOCATION)

Note: CSFRK includes CSFED, hence a user of CSFRK does not have to include CSFED separately.

EDIT Routine -- (CSFEM)

Function: The EDIT routine performs the following functions:

1. Converts all lower-case letters to upper case.

2. Substitutes carriage return characters (X'15') for delimiters in a supplied parameter string (Chart AI).

Entry Point: EDIT is the only entry to CSFEM fetched by the associated POOL routine CSFED. General register 1 contains the pointer to the character string to be edited.

Modules Called: None.

Exit:

Normal: Unconditional branch BR14, with GR1 pointing to the message character string.

Error: GR1 is set to zero before the BR14 is taken.

Operation: Edit assumes that GR1 contains the pointer to the character string to be edited.

It scans the buffer area for delimiters, and replaces them with carriage return characters. Successive blanks will be treated as a single delimiter; i.e., only one X'15' character is substituted for n number of successive blanks. Editing is suppressed, however, if a vertical bar (X'4F') is encountered. Editing is resumed when a second vertical bar is encountered.

Lower-case characters between delimiters will be translated to upper case.

Notes:

1. Applicable to user programs executing in real or virtual core and STS transient routines.

2. The message must end with an EOB.

3. CSFEM makes two passes over the message buffer.

Get Message Subroutine -- (CSFFG)

Function: CSFFG gets a field for the STS user (Chart AJ).

Entry Points: The only entry to CSFFG is GETFLD. CSFFG assumes GR0 contains the field desired by the STS-USER, and GR1 contains the pointer to the beginning of the message buffer.

Modules Called: None.

Exit:

Normal: BR14 with GR1 containing pointer to the beginning of the field. GR0 contains the size of the field.

Exceptional: GR0 is set to zero if a field cannot be found, and then returns with a BR14. This may in fact not be an error condition; e.g., the RUN command might call CSFFG to find out if the second operand exists (i.e., should a naked RUN be performed).

Operation: CSFFG will scan the message buffer for delimiters and increment a counter each time it encounters a delimiter. Two successive delimiters indicate the end of the message. If the field is found, it will place the size and the address of the first byte of the field in GR0 and GR1, respectively.

If the field is not found, the exceptional exit will be taken.

Read/Write Keyboard -- (CSFKR)

Function: The function of this routine is to read from the 1052-7 operator's console, or to integrate to the device assigned by the supervisor for SYSLOG (Chart AK).

Entry Points: There are two entry points to CSFKR: (RDKBD) for the READ operation and (WRTKBD) for the WRITE operation.

Whether READ or WRITE is desired, CSFKR will assume that the contents of GR0 and GR1 are the length of the message and the pointer to the message, respectively.

Modules Called: CSFEM, the EDIT routine is called to compress all the blanks in the message buffer when the READ operation is desired.

Exit: Normal: BR14, with GR1 pointing to the message.

Operation: CSFKR will move the input parameters into the skeleton CCWs for READ or WRITE and EXCP to retrieve or transmit the message.

Get Absolute Core Location -- (CSFGL)

Function: The function of CSFLG is to convert an EBCDIC character string into an absolute core address.

The character string may be one of the following formats:

1. A read core module name
2. A real core module name (±) a hex displacement
3. A real core module name (±) a decimal displacement preceded by an asterisk
4. A hex displacement preceded by a plus sign
5. A decimal displacement preceded by an asterisk (Chart AL)

Entry Point: The only entry to CSFLG is through its associated POOL routine CSFGL.

Modules Called: None.

Exit:

Normal: Unconditional BR14 to the calling routine (not the POOL routine), with GR1 containing the converted address.

Error: If an error condition is encountered, GR1 is set to zero, and control is returned to the calling routine by a BR14.

Operation: Upon entry, CSFLG assumes that GR1 contains a pointer to the beginning of the character string, and that an EOB (X'26') or carriage return character (X'15') indicates the end of the string.

CSFLG will determine which of the five formats the character string is supplied. For format types 1 thru 3, the real core symbol table will be searched to determine the value of the module name or entry point supplied. The error exit will be taken if the module name cannot be found. Distinction is made between hexadecimal and decimal numbers by the presence or absence of the asterisk with the number.

The final address is checked against the range of core. The error exit will be taken if the computed result exceeds the core size of the CPU.

Test Validity of Address -- CSFAD

Function: To determine whether an address is a real core address that exists as part of the current executing CPU; i.e., to insure against the use of an address that is partitioned out or is not part of the system (Chart AM).

Entry Points:

1. ADCOMP is the primary entry point to CSFAD and is entered via a branch and link on registers 14, 15. Register 1 contains the address to be tested.

2. Secondary entry point is ERROR; the routine is entered at this point if a program interrupt occurs while testing the address.

Modules Called: None.

Exits: Return is made via a BR14 with a return code in register 0. A return code of zero indicates that the address was invalid; a return code other than zero ($\neq 0$) signifies a valid address.

Operation: A pointer to the section of coding in CSFAD to handle the case of an unavailable address is moved into a communication area, PGINT, located in REGON. In the event of a program interrupt, control will be transferred by STS supervisor (using the contents of PGINT as a pointer) to the secondary entry point of CSFAD. The address is tested by using the CLC instruction and REG1 in the operand. If the operand of the CLC using register 1 references an unavailable location, the operation will be terminated, causing a program interrupt. If execution of the CLC caused a program interrupt, a zero return code is set in register zero; otherwise, register zero will be set to contain a nonzero value. The original contents of PGINT is restored and return is made to caller.

Assign Device Address Command -- CSFAS

Function: The assign command (ASN) will allow temporary changes to be made to the STS PUB table (Chart AN).

Entry Points: Entry is made into CSFAS at entry point SYSASN. General register 3 contains a pointer to the input message.

Modules Called:

CSFAN - performs actual assignment.

CSFGF - obtains the requested fields from message.

CSFWK - prints the message on the 1052-7 for an error or completion.

CSFEM - edits buffer message.

Exit:

SVC 6 - When the input stream was from a 1052-7 terminal buffer, or when an error has occurred on input from cards.

Fetch SYSSA - When the input was on cards and assignment was completed.

Operation: It is determined whether the input will be from cards or terminal keyboard. If from cards, the card is read in from SYSIPT. In either case, general register 3 will contain the pointer to the input stream. At this point, the second field is obtained, and the symbolic device is presented in the XXX portion of SYSXXX. The XXX portion is moved into an 11-byte parameter list. The next three fields, containing DEVICE ADDRESS, TYPE, and MODE, are similarly obtained and placed into the parameter list. A BAL is made to CSFAN as the result of an expansion of the ASSIGN macro, with register 14 pointing to the parameter list. On returning, a test of the return code is made to determine if assignment was successfully completed or not. If negative, a diagnostic message is printed out on the 1052-7 terminal, and return is made to CRR via an SVC 6. If positive, and input was from the 1052-7 terminal, an OK message is printed, and exit is made to CRR. However, if the input was on cards, the next card is read, and the above is repeated until //END card was encountered, at which time exit is made to SYSSA.

Assign Device Address Subroutine -- CSFAN

Function: CSFAN assigns (in the physical unit block (PUB) table) a symbolic device to a physical device address (Chart AO).

Entry Points: Entry points are ASSGN and CTPUB. Entry is made by the issuance of an ASSIGN macro from the ASN command routine. The operand is the address of an 11-byte field (whose address is contained in register 14) containing four subfields.

1. Symbolic device (three bytes EBCDIC) RDR, LST, IPT, OPT, LOG, or 000-255.

2. A value from 0000-3FFF (4 bytes EBCDIC) representing:

   Channel Controller - 0-3
   Channel - 0-F
   Control Unit - 0-F
   Device Unit - 0-F

3. MODE (two bytes EBCDIC)

   a. For 7-track tape T1.
   b. Ignored if none possible.

4. TYPE (2 bytes EBCDIC) indicating device type.

Modules Called: None.

Exit: Exit is made by a B 4(14).

Operation: The location of the parameter string is contained in register 14. The XXX portion of SYSXXX is compared to RES; if equal, the relative address (relative to PUB) is obtained. If not equal, the address of the PUB and the particular entry is obtained. The address, mode, and type are converted to entry form, and the PUB is set up.

Note: For all MODE bytes, two blanks or zeros are accepted as indicating that no mode settings are required.

Display PUB Routine -- CSFPB

Function: Display entire PUB table (series of devices) or a particular entry in the table (Chart AP).

Entry Points: DISPB1 is the single entry point to CSFPB. Register 3, on entrance, contains a pointer to the 1052-7 terminal message buffer.

Modules Called:

CSFWK - print out the PUB entries.

CSFGF - obtain the fields of the input stream.

CSFGL - resolve absolute core location.

Exit: Exit is made via an SVC 6 to CRR.

Operation: The address and number of entries of the PUB table is obtained from REGON. It is then determined whether there is a third field in the input stream. If the third field exists, a particular entry within the table is desired, and the table is searched for the entry and printed out via a BALR to CSFWK. If no third field is present, a printout of the entire PUB table is wanted. Each full word entry is tested for zeros; if the entry contains all zeros, it will not be printed out. This testing is conducted for the entire table until the end is reached, at which time exit is made by issuing an SVC 6.

Execute User Program Routine -- CSFRU

Function: The RUN command activates the execution of a user program phase or restarts execution if processing was stopped for a specific reason (Chart AQ).

Entry Point: SYSRUN is the only entry point. General register 3 points to the input stream.

Modules Called:

CSFWK - writes messages on the 1052-7.

CSFFG - finds a particular field in parameter list.

CSFGL - computes absolute core location of specified field.

Exits:

With an SVC 7 to the user program after successful operation.

With an SVC 6 to the command recognition routine when the requested starting location is below STS's supervisor end, and the user program is in problem state.

Operation: CSFRU is invoked by the command recognition routine. Upon entry, R3 is pointing to the input buffer. Parameters are set up, GETFLD is called, and returns with a pointer in R1 to the second field of the RUN command. The second field specifies the starting address and may be a symbol or an absolute address. If the second field is defaulted, the machine status, as saved on entry to STS, is restored. The old PSW in REGON is checked to see whether the task was in problem state or supervisor state. If the requested starting location is in the supervisor and the interrupted program was in supervisor state, a warning message is issued, and processing continues. If the starting location is in the supervisor and the interrupted program was in problem state, a message is printed, and control returns to the command recognition routine.

LINK -- CSFLN

Function: CSFLN informs STS that TSS is the user and resolves an entry point within the interrupt stacker that contains the address of the queue scanner (Chart AR).

Entry Point: link1 -- control is transferred here when phase CSFLN is FRASUed by the exercisor (CSFXR).

Exit:

Normal: SVC 7 restore user program--in manual.
Fetch CSFXR--in automatic mode.

Error: Appropriate error messages, followed by SVC 6.

Modules Called: CSFGL (GETLOC) is invoked to obtain an address with the Interrupt Stacker.

Operation: CSFLN is invoked via a fetch and performs initialization functions for STS. Basically, it informs STS that TSS is the user by setting the MODTSS bit in the MODESW. Since TSS/360 currently operates only on a Model 67, we call CSFGL to translate the address in the interrupt stacker (CEAJIA), which contains the address of the queue scanner. This address is modified at times by STS when it receives a TSS interrupt that it cannot stack but which requires control returned to it. Upon completion of this function, exit is made in normal mode via an SVC 7, which restores the status of the machine prior to the FRASU state, which returns to the exercisor.

Exercisor -- CSFXR

Function: To provide initialization at the START entry point and to set up for a PAUSE (user requested task stop) at the EXER2 entry point (Chart AS).

Entry Points:

EXER	-	called from START to complete initialization.
EXER2	-	entered when a task is about to get control to permit PAUSing of the task.
START	-	initial entry to CSFXR by the initializer phase of CSFSU after STARTUP.
TIMECON	-	an external definition to hold the timer value set up by STARTUP.

Exit: CEAJQS - queue scanner (TSS module).

Modules Called:

1. CEAJQS - Queue Scanner.
2. CEAL01 - Supervisor Core Allocation.

Operation: The initializer phase of CSFSU sets up the current PSW in the STS communication region (REGON) to point to the entry point START in CSFXR. START immediately FRASUs CSFLN to perform some housekeeping functions, which return to the point in CSFXR following the FRASU. The timer value set up by STARTUP and the save in TIMECON by the initializer of the supervisor are restored, and the initial entry point (EXER) of the exercisor (CSFXR) is invoked. This results in an immediate exit to the queue scanner (CEAJQS).

Entry point EXER2 is invoked by the dispatcher just prior to activating a task. If the TSI pointed to by 188 does not have the pause bit on, an immediate return to the dispatcher is effected. If the TSI pointed to by 188 does have the pause bit on, the page pointed to by the PSW contained in the XTSI is tested to determine if it is in core. If the page is not in core, the routine exits to the dispatcher; if the page is in core and the pause set bit is on, exit is also made to the dispatcher, since

we have already completed the exercisor's pause procedure.

Assuming that the page pointed to by the PSW in the XTSI is in core and the pause set bit is not on, supervisor core allocation (CEAL01) is invoked to obtain a save area for the task registers 14, 15, 0, and 1. The task registers in the XTSI are then modified so that GR1 contains a pointer to the core allocated by CEAL01; GR14 and 15 contain the character C'PAUSEbbb'; and GR0 contains the two bytes of the first instruction to be executed by the task. These two bytes which were saved are overlaid by a FRASU SVC 10, the pause set bit is set on, and control is returned to the dispatcher.

Virtual Memory Processor -- CSFVM

Function: Designed to process two types of requests (from programs operating in virtual memory, only).

1. TCOM - transmits and receives messages to and from the 1052-7 console.

2. TDUMP - permits dumps of real or virtual core memory areas or combinations of both (Chart AT).

Entry Point: Entry is at SYSUMP10; it is the result of a macro expansion into an SVC 5. A parameter list generated in a CSECT or PSECT follows the SVC 5.

Modules Called:

CSFRK - obtains information from the 1052-7 when an input message is desired.

CSFWK - prints out TCOM information on the 1052-7.

CSFPG - obtains the address of the virtual memory page.

Exits: There are two types of exits made from CSFVM:

1. SVC 7 - issued on entry via a TDUMP or TCOM to print a message.

2. SVC 6 - issued when entered through TCOM when message being printed is a question mark (?).

Operation: The location of the parameter string is calculated, and from the first byte configuration it is determined whether the macro was TCOM or TDUMP. At this point, a branch is made within CSFVB to perform the desired operation. The type of output for TDUMP is further determined whether a real core or virtual memory dump is wanted; whereas for TCOM it is determined if the request is for receiving or transmitting messages.

TDUMP: After obtaining the type, the parameter list is further examined for the from and to addresses. The dump is taken, and CSFVM refers back to the parameter list to check for a chained parameter list. If the list is chained, the above is repeated; if not, an SVC 7 is issued.

TCOM: It is determined whether CSFVM is to receive or transmit messages:

1. Receive -- if CSFVM is not expecting a response, it proceeds to check for a chained parameter list and reacts as it did in TDUMP. If a response is expected, CSFVM awaits the response and inquires into a chained list upon receipt.

2. Transmit -- determines the type of message. If it is a question mark (?), an SVC 6 is issued; otherwise, the message is printed out on the 1052-7, and a check is made for a chained parameter list, as before.

Find Page -- CSFFP

Function: CSFFP locates segment, auxiliary segment, page, and external page table entries. A subroutine will locate the proper RSPI entry location in RC for any SPT number or will locate the next available entry in the RSPI (Chart AU).

Entry Points:

FNDPAG - standard linkage.

Input - R3 = location of XTSI.
R4 = VM address.

Exits:

Normal: Standard linkage - return to calling routine.
R9 = number of remaining page table entries.
R10 = A (Segment Table Entry).
R11 = A (Auxiliary Segment Table Entry).
R12 = A (Page Table Entry).
R13 = A (External Page Table Entry).

Error:	Error in search RSPI table	- CC = 1
	Segment not assigned	- CC = 2
	Segment or page table length invoked	- CC = 3
	Segment length exceeded	- R13 = 0
	Page length exceeded	- R13 = 1

Modules Called: None.

Operation: CSFFP first checks the length of the segment table and computes the address of the segment table entry and the auxiliary segment table entry. If the segment is shared, an internal subroutine locates the RSPI pointer and the shared page table number and returns the address of the pointer to the page table and external page table. From XTSI the segment table is located first, then the auxiliary segment table, the page tables, and the external page table. These addresses are passed back in R9-12, respectively. If errors are detected, a condition code is set, with a more specific code in R13, and CSFFP returns to the calling program by a B on R14. Normal return is also a B on R14. CSFFP is called only by CSFPL.

Convert a Symbolic Virtual Memory Address -- CSFTD

Function: CSFTD converts the EBCDIC representation of a hexadecimal number into actual hexadecimal notation (Chart AV).

Entry Point: The entry point is TDYADD. Upon entry, CSFTD assumes that GR0 contains the number of hexadecimal characters to be unpacked, and GR1 contains the pointer to the hexadecimal string.

Modules Called: None.

Exit: Exit is made by a BR14 with converted quantity in general register 1.

Operation: By a series of executes in MVC, TR, and PACK, the VM address is converted into actual hexadecimal notation.

Note: CSFTD assumes that the maximum number of hexadecimal characters to be converted is eight, and that the unpacked result will be able to fit into a general register (GR1).

Locate Virtual Memory Page -- CSFPL

Function: CSFPL is a subroutine called by VM routines to get a physical memory address for a given VM address or an external address (TSS symbolic address) (Chart AW).

Entry Points:

PAGLOC - standard linkage.

R15 = V(PAGLOC)

R0 = VM address

Exits:

Normal: B 4(14)

Error: BR 14
R1 = TSS symbolic address if CC=0
R1 = Error code if CC≠0

Error Code	Meaning
0	Page is not assigned external storage address.
1	Segment specified is too large.
2	Segment specified is not assigned.
3	Page specified is too large.
4	Page table specification error.
5	Error in locating system table.
6	NO active TSI.
7	XTSI not in core.
8	Control register 0 error.
9	Error encountered searching RSPI table.

Modules Called: Find page (CSFFP) to locate segment and page tables for VM address.

Operation: After initial housekeeping, CSFPL calls CSFFP to get the segment and page table addresses, which are returned in R9-R12. Assuming that CSFFP detected no errors, a check is made of the page table to see if the page is in core or if it has been paged out (last byte of entry = 8). If it is in core, CSFPL returns to the calling program with an address in R1. If the page is marked out of core, the block entry is checked, and if forward and backward core block pointers = 0, it is still in core, and the core address is returned.

If core block pointers are ≠ 0, the page is out of core, and the symbolic device address associated with it is used from the external page table. The page table entry is zeroed out so that the next time it is paged in the entry it will be filled out by passing the reclaim page function. Normal exit is B 4(14); error exit is B14, with condition code set to nonzero and error code in R1 or CC=0, with symbolic device address in R1.

Get Virtual Memory Page -- CSFPG

Function: CSFPG is called to obtain the pointer in core to the VM page requested and, if necessary, to bring this page into core (Chart AX).

Entry Point:

PAGGET - Standard linkage.
R1 - Address of VM page.

Exits:

Normal: B 4(R14) standard linkage.
R1 = Core address of page.
R0 = CCB address.

Error: B (R14)
CC≠0 R1 = Error code (returned from pageloc).
CC=0 R1 = Error code -- error detected by CSFPG.

Modules Called:

CSFPL - obtains the physical address of the VM page address.

CSFGL - converts EBCDIC character to absolute core locations.

Operation: CSFPG first calls pageloc to obtain the physical address of the VM page address. If page core finds the page in core it returns the physical address in R1, and CSFPG returns. If the page is not in core, either the page is external or an error occurred. CSFPG analyzes the return code, and if no errors occurred it takes from the external page entry the symbolic device address of the page. In the physical/actual table, CSFPG locates the device group table printer and constructs eight possible addresses. It then calls reverse pathfinding until it returns a symbolic device address matching that of the device address in the external address passed from PAGLOC (CSFPL). Now CSFPG has access to the physical address for paging operation. The specific location on disk or drum is read by PAGGET, and it returns to the calling program with the address of the page brought in R1, R0=CCB address. Return is made by B 4(R14). In detection of error conditions, CSFPG does a BR 14 with an error code in R1.

Buffered Page Get -- CSFBP

CSFBP is identical in operation to CSFPG (Chart AY).

CSFBP is called in the process of dumping virtual memory, and it includes the dump block (CSFDB).

CSFDB contains instructions from the VM processor that are executed only once. After execution, control is passed to CSFPB, which overlays the CSFDB code with its I/O buffer. Subsequent calls to GET PAGE call CSFPG.

Dump Block (VM) -- CSFDB

Function: CSFDB processes parameters for a VM dump, checking for validity in both form and logic. It then prints task identification user supplied parameter information, registers, the TSI, XTSI and shared page tables. After establishing dump format, dump address parameters are computed and CSFDB returns to calling program (Chart AZ).

Entry Point: DMBVDM - standard linkage.

Exits:

Normal - standard linkage.
R1 = VM map indicator.
R7 = from VM address
R0 = then VM address

Error - following errors result in an appropriate message to operator and SVC 6 ('?'):

1. No active tasks.
2. Error in parameter.

Modules Called: CSFDO - at entry points:

DUMP - to print TSI, XTSI, and page tables.

SETUP - establish print format.

IOCCW - set print spacing and page ejection.

LABEL - set labels for print of registers.

LOOPC8 - print registers.

IO - print heading and skip lines.

LINE - obtain dump line count (for print).

HEADING - store ID information for print.

CSFTD - convert VM addresses.

Operation: After saving the SYSVDMs, return register CSFDB checks if there are any active VM tasks, and puts task ID information from the TSI and input parameters into an external buffer area in CSFDO entry point HEADING.

A request to "dump" VM map or "VM all" will result in a subsequent fetch of SYSPVM. Parameters of specific VM addresses are validated by branching to CSFTD. A branch to CSFD at entry point SETUP formats the floating point dump, and floating point registers are printed by a branch to CSFDO at entry point LOOPC8. Control and general registers are obtained from REGON or the XTSI, formatted in hexadecimal by SETUP, and printed. A pointer to the TSI is passed to the CSFDO routine for printing. This procedure is repeated for the XTSI and shared page tables. Internal branching to ADJLINE determines line spacing and page ejection for printing. Dump format is established by a branch to SETUP with specified input parameter. The "from" and "thru" VM addresses are loaded in registers 7 and 0, respectively. A VM map indicator is set in register 1, return registers are restored, and CSFDB returns to calling routine. If map is the only parameter, SYSPVM is immediately fetched.

Patch Real Core Processor -- CSFPA

Function: CSFPA is used to patch real core, general registers, control registers, communication points (CMPS), dynamic dump requests (DDRS), and to accept card input (in specified format) from SYSIPT (PAT CARDS). It additionally provides linkage to other patch routines, depending on the specification of the first operand (Chart BA).

Entry Points: CSFPA is fetched by CSFCD when the first three letters in the first field typed by the user are PAT. CSFCD passes a pointer to the input message buffer in general register 3. The parameters following the PAT in the input message specify the type of patch and contents.

The sole entry point is SYSPAT1.

Exit: CSFPA exits with an SVC6 to invoke CSFCD. If the input is from cards a "patend" precedes the SVC6 call.

"DONE" is printed upon completion of a patch requested from the 1052-7.

Modules Called:

CSFVP - pat VM processor.
CSFD1 - pat disk processor.
CSFGL - used to get absolute core location if a symbol is specified for patching.
CSFGF - used to write diagnostic messages to the 1052-7 terminal.
CSFED - used to edit card input. (CSFRK edits output from the 1052-7.
CSFAD - used to insure CSFPA that the user is patching a valid core address.

Operation: Upon entry CSFPA analyzes the first operand to determine what is to be patched. If the first operand is VM, DA, or DISK, CSFVP or CSFD1 is fetched; otherwise, processing is internal to this routine. If patching general or control registers are specified, the patching is done in the user register save area (REGSTR, SCTLREG, respectively) in REGON. This is because the supervisor will load the hardware registers from these areas when the user resumes running.

When patching real core, CSFPA first makes certain that the user may access the core he requests be patched, via a call to CSFAD.

When a CMP or DDR is patched; CSFPA first checks to see that the location specified for the CMP or DDR contains a "replaceable instruction" (i.e., a branch instruction may not be overlaid for a CMP or DDR).

If the instruction is replaceable, its core address is stored in the CMP or DDR table, with the first two bytes of the instruction. The DDR or CMP count is upped by one. A maximum of 50 DDRs and 50 CMPs may be requested.

Virtual Memory Patch -- CSFVP

Function: CSFVP provides the capability to modify virtual memory core locations, registers, and PSWs (Chart BB).

Entry Point: Entry into CSFVP is SYSUPT. A pointer to the parameter input stream is contained in register 3.

Modules Called:

CSFPG - locate page in core or on external device.

CSFWK - print out error message if any errors encountered.

CSFFG - scan parameter string for necessary data.

CSFLG - interpret number from EBCDIC to Hexadecimal.

Exit: Return is made to CRR via an SVC 6.

Operation: A BALR is made to CSFFG to obtain the contents of the third field of input stream. The area to be patched is determined, and the proper branch is made. If a register or PSW is to be patched, their locations are determined according to the status of the system (i.e., locations may be in either REGON or XTSI), and the modification is performed. If, however, the patch is in virtual memory, a BALR to CSFPG is made to obtain the location of the page in question. On returning, register 1 will contain a pointer to a page which in turn will contain the area to be patched. Register 0 will contain either one of two values:

1. Zero Quantity: This return code signifies that the page was in core, and modification can be made.

2. Non zero Quantity: A return code other than zero indicates that the page was external and had to be brought into core. Upon completing modification, an EXCP is issued on a CCB constructed by CSFPG, to restore status of page. In both

cases, after modification is completed, return is made via an SVC 6 to CRR.

Note:

1. If page was external, the modification will be permanent, as opposed to a temporary patch when it is in core.

2. Patches may not cross page boundaries.

Snapshot Processor -- CSFDR

Function: CSFDR is used to process dynamic dump requests and communication points (Chart BC).

Entry Point: CSFDR is invoked by an SVC 5 which was implanted in core when a CMP or DDR was requested.

Modules Called: CSFDO - used by CSFDR to perform the dumping requested onto the device assigned to SYS001.

Exits: CSFDR exits by an SVC 7 to allow the user program to continue processing.

Operation: CSFDR is entered when an SVC 5 is encountered during execution. CSFDR first checks the DDRTABLE in the REGON area to determine whether any (more than one may be specified) DDRS were requested at this location. If there are DDRs requested, they are serviced by means of using CSFDO to do the dumping to SYS001. At the beginning of each dump, a symbol table is printed. This is followed by the control registers and general registers. If the DDR ID is between 50 and 99, the symbol table will not be printed.

When all dynamic dump requests are serviced, an attempt is made to execute the instruction previously overlaid by the SVC 5 to cause the dynamic dump or the communication point. The machine status, as saved prior to invoking the snap processor, must be restored before executing the instruction. This is done by modifying the PSWs to point to the instruction followed by an SVC 7. This instruction is reconstructed, placed in the PSA, and will then be the subject of an EXECUTE instruction. If a program interrupt occurs, the supervisor checks the field PGINT in REGON (the transient program interrupt processor). If there is an address in PGINT, control is transferred to that address. Otherwise, the supervisor will handle the interrupt in its usual fashion. If no interrupt occurs, execution follows

the logical sequence of code that will result in another SVC 5. This will again save the user's condition and will cause a second invocation of the snap processor. A switch, previously set, indicates that the snap processor was called the second time. Then the PSW will be set up to point to the instruction that would have normally been executed. A test is made as to whether there was a CMP. If there was none at this location, the machine status as saved upon entry to the snap processor is restored, and an SVC 7 returns control to the user. If there was a CMP request, CSFWK puts out the CMP message and ID, followed by an SVC 6 to return to the command recognition routine.

DELETE Routine -- CSFDE

Function: To delete dynamic dump requests (DDRs or DRs) and real core communication points (CMPS) (Chart BD).

Entry Point: The only entry point is SYSDEL. CSFDE is fetched by the command recognition routine (CSFCD), which passes CSFDE a pointer to the input message buffer in general register 3.

Modules Called:

CSFGL - used to get the core location of the CMP or DDR to be deleted.

CSFGF - to access the fields typed in by the user.

CSFWK - to write diagnostic messages to the 1052-7 terminal.

Exits: If the CMP or DDR delete request is valid, a message, "DONE", is printed after the entry in the CMP or DDR table has been deleted. If the request is invalid (i.e., no DDR or CMP is found at the specified address) a message. "NO", is printed. In either case, after the message has been printed, CSFDE issues an SVC 6 to call CSFCD.

Operation: Upon entry, CSFDE examines the message buffer to determine whether

- DEL ALL
- DEL CMP [at address]
- DEL DDR (or DR) [at address]

has been specified. If ALL has been specified, all CMPs and all DDRs are deleted; otherwise, the CMP or DDR at the address

specified is deleted.

The deletion is accomplished in the following way. The symbols DDRTABLE and CMPTABLE, found in the REGON DSECT, point to the table desired. The byte immediately preceding these tables is a count of the CMPs or DDRs in the respective tables. Each table may have a maximum of fifty entries.

After locating the desired table, CSFDE obtains the at address specified by the user and converts it to an absolute core location. The appropriate table is then scanned for a matching absolute core location. If a match is found, the SVC 5 at the core location is overlaid by the two bytes stored in the table when the request was made. The count is decreased by one and the DONE message is printed.

If the at address, when converted, does not match any core location in the table, "NO" is printed.

Dump Out Processor -- CSFDO

Function: CSFDO is a subroutine used to dump information onto the device assigned to SYS001 (printer or tape). If the dump is to tape, and an EOF is detected, the dump will continue to SYS010 if it is similarly assigned (Chart BE).

Modules Called: None

Entry Points: Entry points to CSFDO include:

SETUP - to set up dump format.

HEADING - user may move in a 112 character header into this area.

DUMP - enter here to do actual dumping. FROM and TO addresses in R8 and R9 respectively.

I/O - entry point to have CSFDO execute the CCW at I/O CCW.

CONVX - convert input (pointed to be GR1) to hexadecimal format and place it in output buffer (pointed to by GR2).

CONVF - convert input to full-word output - GR1 and GR2 used.

CONVH - convert input to half-word output - as in CONVX.

IOCCW - move in the CCW to be executed here.

LABEL - label for dump may be moved in here.

Exits: CSFDO is a closed subroutine and exits with an unconditional branch to general register 14.

Operation: A user generally enters CSFDO initially at an entry point called SETUP. Here the user is able to set up the format he wishes for the dump (hexadecimal characters, etc.), and also the blocking of the data (halfword, fullword, etc.). Before entering SETUP, the user may move into an externally defined area called HEADING, a 112-character heading which will be printed at the top of each page of the dump. In addition, a user may move a 10-byte "id" into an externally defined area called LABEL. This is printed on the same line with the heading.

After SETUP, the user enters at DUMP to perform the actual dumping, in the format specified at SETUP time. The from and to address for the dump is specified in general registers 8 and 9, respectively.

The I/O required for this dumping is performed internally. All translation is also performed internally by subroutines and CSFDO may be called for the sole purpose of translation. The parameters passed are pointers (1) in general register 1 to what is to be translated, and (2) in general register 2, a pointer to the drop area. Four bytes are translated before the called routine exits with a BR 14.

Additionally, CSFDO may be called to execute a CCW specified by the user. The CCW is moved into the area specified by the external symbol IOCCW. The user then enters at the ENTRY point IO to execute the CCW.

Dump Command Routine -- CSFDU

Function: The DUMP command (Chart BF provides the user with:

1. dumps of real core.
2. linkage to other dump routines which dump:
   a. TSS System status information (abridged table dump);
   b. the contents of direct access devices;
   c. a sorted real core symbol table, or
   d. the current active task virtual memory.

Entry Point: SYSDUM is the only entry point. Upon entry, GR 3 is pointing to the input buffer.

Modules Called:

CSFDO - prints dumps on SYS001 and SYS010.

CSFGL - to compute absolute values.

CSFFG - to locate a field.

CSFWK - to write on terminal.

CSFPF - to get address of prefix area.

CSFAD - to determine whether the specified storage element belongs to the dumping CPU.

Exit: If the operand is DA, VM, ATD, or MAP, the module affiliated with these parameters is fetched. In any other case, it exits with an SVC 6 to the command recognition routine upon completion of processing.

Operation: CSFDU is invoked by the command recognition routine. Upon entry, GR 3 points to the input buffer. When CSFDU is entered, the interrupt log is transferred from the standard area to the space between the end of SYSDUM and SYSEND (i.e., available space in the transient area), since dumping destroys the user log progressively as it dumps. If the log is longer than the available space, the log is truncated, losing the oldest interrupts.

Saving starts with the last known interrupt and moves back as much as the space allows. If the operand is DA (direct access), VM (virtual memory), ATD (abridged table dump) or MAP (sorted symbol table map), the module affiliated with these parameters is invoked. If the operand is ALL, the FROM and TO addresses are computed, using a pointer in REGON which gives the size and configuration of the machine. If it is not any of the above, it is sent to GETLOC. If the operand is a symbol, GETLOC will return its address. If the operand is an absolute decimal address, it must be preceded by an asterisk.

The PSWs, control registers and general registers are dumped, and the portion of real core that was requested is dumped by DMPOUT. A symbol table map is also dumped with

with the real core dump. The map is set up by CSFSS. The output goes on SYS001 or SYS010.

If the to address is out of range, no dumping takes place, and a message is sent to the terminal. Holes within the from and to addresses are deleted, and dumping takes place.

Numeric Symbol Sort Processor -- CSFSS

Function: The sole function of CSFSS is to output a numerically sorted version of the real core symbol table to the device assigned to SYS001. (Chart BG).

Entry Point: The only entry point to CSFSS is SYSNSS. CSFSS is fetched by CSFDU when the user types "dum map" on the 1052-7 terminal. CSFCD fetches CSFDU, which, in turn, fetches CSFSS.

Modules Called:

CSFIO - an I/O package used by CSFSS to dump the map to SYS001.

CSFWK - used to send diagnostic messages to the 1052-7 terminal.

Exits: CSFSS exits by issuing an SVC 6 to invoke CSFCD.

Operation: When entered, CSFSS gets the contents of PCTADR, a symbol in the REGON DSECT. PCTADR points to the halfword immediately following the real core symbol table that contains a count of the number of 11-byte entries in the table.

The count is multiplied by 11. By subtracting this product from the end of the table, we can begin our scan. The scan is accomplished by searching for the low entry in the table, printing it, and setting a flag to exclude this entry from subsequent scans. When all entries have been printed, the flags are turned off.

If PCTADR is zero, a message is printed indicating this.

Virtual Memory Dump -- CSFVU

Function: This routine dumps virtual memory in one of four formats (Chart BH).

Entry Point: CSFVU has a single entry point at SYSVDM. A pointer in register 3 will point to 1052-7 terminal buffer

message and will be the input to CSFVU.

Modules Called:

CSFPG - locates virtual memory page.

CSFDO - dumps indicated area.

CSFWK - used to print out any error messages encountered.

CSFDB - prints out header, general registers, control registers, floating point registers, PSWs, TSI page, XTSI pages, and shared tables.

Exit: Exit is made via an SVC 6 to DRR. A DONE message is printed on the keyboard upon completion of the dump.

Operation: A BALR is made to CSFDB to have the header, general registers, etc., printed out. Upon returning, the from and to addresses are obtained from the input stream and passed to CSFPG (CSFPG is called for each page to be dumped). After obtaining proper pages, a branch and link is made to CSFDO (as parameters in registers 8 and 9) passing the from and to addresses and the format to dump out page. After the entire area has been dumped, return is made to CRR.

Note:

1. Area is dumped according to the assignment of SYS001.

2. Will manufacture a to address to the end of core if the parameter is omitted from input.

Dump VM Map -- CSFMB

Function: CSFMB accesses the task dictionary table for a particular virtual memory task and prints a listing of all control sections referenced, with additional information, such as VM address, attributes, length, and VM address and separation of all external references (Chart BI).

Entry Point: SYSPVM; no parameter list.

Exits:

Normal: SVC 6 - "7" at console.

SVC 7 - restore system at point of FRASU.

Error: return code from called routines printed in error message, followed by "?", SVC 6.

Modules Called:

CSFPL - finds VM page.

CSFPG - access as VM pages of TDY.

Operation: CSFMB accesses the ISA for the address of the TDY header and, using subsequent calls to CSFPG, accesses a program module dictionary group, computing relocation adjustment and real core addresses of last PMD in the group. Accessing this PMD preface again through CSFPG, the PMD body is finally accessed. If it is determined that any portion of the PMD group may extend to another VM page, an execute instruction is modified to branch to a subroutine that checks the VM address in the next instruction. If it exceeds page boundaries, CSFPG is called to access that VM page, and the real core addresses and relocation adjustment are modified. For each control section, CSFMB prints name, sequence numbers, VM address, separation, and attributes, as well as a list of all entry points and external references with VM addresses and separation. A subroutine at Print $ modifies and executes printer CCW, while routines at taprtn modify tape CCW. Any error in VM discovered by CSFPG results in appropriate error message. Normal return is resumption of frasud task or '?' at consoles.

Display Real Core -- CSFDI

Function: CSFDI provides a means of displaying the contents of real core storage, general and floating point registers, PSWs, CAWs, and CSWs (Chart BJ).

Entry Point: SYSDIS is the only entry point. Upon entry, GR3 points to input buffer.

Modules Called:

CSFLG - to compute absolute values.

CSFFG - to locate a field.

CSFWK - to write to 1052-7 terminal keyboard.

Exits: If the operand is DISK, DA, STATUS, or PUB, the modules affiliated with these parameters are fetched. Otherwise, it exits with an SVC 6 to the command recognition routine.

Operation: CSFDI is invoked by the command recognition routine. When the routine is entered, R3 is pointing to the input buffer. Parameters are set up, GETFLD is called, and returns with a pointer in R1 to the second field of the DISPLAY command. If the operand is DISK, DA, STATUS, or PUB the modules affiliated with these parameters are fetched. All other display requests are processed within SYSDIS. Pointers to the fields to be displayed are picked up in REGON. The display request forces full-word alignment.

Display Virtual Memory Processor -- CSFVD

Function: To display virtual memory locations and affiliated registers and PSWs (Chart BK).

Entry Point: SYSVDS is the entry point to CSFVD. Upon entry, register 3 contains the address of the parameter input stream.

Modules Included:

CSFPL - get a physical memory address from a given virtual memory address.

CSFPG - obtain location of virtual memory page and insure that it is in core.

CSFFG - obtain parameter of input stream.

CSFLG - convert EBDIC characters in hexadecimal notation.

CSFWK - print error messages and complete message.

Exit: Exit is to CRR via an SVC 6. A DONE message is printed out on the 1052-7 upon completion of display.

Operation: The third field is obtain via a BALR to CSFFG. Upon returning, the type display requested is determined from the contents of the field (registers, PSWs, core). The from and to addresses are determined from the input stream, and any necessary conversions are made via a BALR to CSFLG. The area of interest is printed out, and an SVC 6 is issued.

Set Control Function Status -- CSFSE

Function: The SET command activates or deactivates control functions related to STS/TSS operations.

Its function is to pause virtual memory execution (Chart BL).

Entry Point: SYSSTT is the only entry point. SYSSTT is entered with R3 pointing to the input buffer.

Modules Called:

CSFGF - finds a particular field in parameter list.

CSFWK - writes messages on 1052-7.

Exits:

1. With a fetch to the routine requested by the operator.

2. With an SVC 6 upon error detection in the parameter list of the SET command.

Operation: CSFSE is invoked by the command recognition routine. Upon entry, R3 is pointing to the input buffer. CSFGF is called and returns with a pointer in R1 to the second field of the SET command. The second field is examined, and when a match is found the appropriate routine is fetched to analyze the remaining parameters of the SET command.

If no match is found, or the second parameter is missing, the operator will be made aware of the error via a message on the 1052-7 terminal. Control will be given to the operator, so that he may reenter the SET command.

Set Pause Bit Processor -- CSFSP

Function: This command is used to set the pause mechanism in the TSI for a particular task on or off (Chart BM).

Entry Point: SYSPAU is the only entry point. At input general register 3 is pointing to the 1052 terminal message buffer.

Modules Called:

CSFRK - reads messages from the 1052-7.

CSFWK - writes messages on the 1052-7.

CSFGF - finds a particular field in the parameter list.

CSFGL - computes absolute core location of specified field.

Exits: With an SVC 6 whether or not the operation is successful.

Operation: CSFSP is invoked by the SET command. Upon entry, R3 is pointing to the input buffer.

CSFGF examines the third field for the ON or OFF condition. If the third field is missing, the user is asked via a message on the 1052-7 console, to specify ON or OFF.

The user is asked to identify the task with either the task-id, conversational user-id, batch user-id, or batch sequence number. The task-id must be entered as a 4-character number, the BSN must be entered as a 3-character number. If it is a user-id it must be preceded by the word BATCH or CONV.

CSFSP scans the active and inactive lists to determine whether the specified task exists. If the task is not found in either of these lists, a diagnostic message is issued, and control returns to the command recognition routine. If the task does exist, the pause bit in the TSI is set to ON or OFF, as requested by the user. If the task is in the middle of a pause, the operator will be notified with a message on the 1052-7 terminal, and control returns to the command recognition routine. Scanning of the active list is accomplished by picking up a pointer to the first task in the first word of the DSECT CHBSYS (systems table). From there on each TSI has a pointer to the next TSI in the chain until a match is found or the forward pointer is zero; either case terminates the scan. If the task is not found in the active scan, CSFSP uses SYSFIT (a field in CHBSYS) to get a pointer to the first inactive task and continues in the same way as the inactive scan.

Pause Subroutine -- CSFPZ

Function: The pause subroutine prints the userid, the deviceid and taskid of the current user task on SYSLST or SYSLOG. It additionally resets the task status as it was prior to the exercisor (CSFXR) modification of the task (Chart BN).

Entry Point: Pausel - control is transferred here when phase CSFPZ is fetched via a FRASU (SVC10) (executed in the task virtual memory) that was set up by the exercisor.

Exit:

Normal: SVC 7 - resume user task (in automatic mode).

SVC 6 - '?' enter CRR (in manual mode).

Error: None

Modules Called:

CSFGL - obtains address CEAL2.

CEAL2 - releases supervisor core allocated area; PAUSE is invoked via a FRASU that is set up by the exercisor at entry point EXER2.

Operation: After obtaining TSI and XTSI addressability, CSFPZ obtains the system PSW (SVC9) to save the current system mask.

Interrupts are disabled, using an SSM, and pointers are set to the old PSW and the registers saved in either the XTSI (if TSS interrupt) or in the STS save area. The old PSW instruction counter is bumped back to the SVC, which is modified in real core to reflect the true instruction that was saved in user register 0 by the exercisor. The address of supervisor core allocated area (previously obtained by CSFXR) is obtained from user register 1 and used to restore user registers 0, 1, 14, and 15. CSFGL is called to obtain the address of the supervisor core release routine (CEAL2), and CEAL2 is called to release the supervisor core allocated area (64 bytes). Interrupts are enabled using the saved current system mask, and the pause set bit is turned off in the TSI to indicate that a pause has been completed. The pause bit, used to indicate that a pause is desired for this task, is reset.

The taskid and userid are translated and printed on SYSLST if in automatic mode, or on SYSLOG if in manual mode. If the task is in automatic mode, an SVC 7 resumes the user; if in manual mode, an SVC 6 ('?') returns control to CSFCD.

Patch, Dump, Display Disk -- CSFD1/CSFD2

Function: This routine provides the STS user with the capability to patch, dump, and display direct access service contents. (Chart BO and BP).

Entry Point: SYSDK0 is the entry point to CSFD1, and SYSDK1 the entry point for CSFD2. Upon entry, R3 is pointing to the input buffer.

Modules Called:

CSFGF - to locate a field.

CSFGL - to compute absolute values.

CSFWK - to write on terminal,

CSFRK - to read from terminal.

Exit: When an EOB is sensed and the input stream has been processed, the routine exits with an SVC 6 to the command recognition routine.

Operation: Phase I (CSFD1) provides communication between Phase I and Phase II (CSFD2) and preserves all information prepared by Phase I and used by Phase II.

This routine is invoked by standard dump, display, or patch commands, upon encountering either DUM DA, DIS DA, OR PAT DA. CSFD1 is fetched to perform the desired operation. CSFD1 will respond on the terminal with one of the following messages:

```
UNIT CYL TRK REC FROM TO FORMAT

UNIT CYL TRK REC AT FORMAT TEXT
```

The first message is for dumping or displaying; the second one for patching.

Upon return from CSFWK, a switch is set to bypass the printing of this message when CSFD1 is refetched at the end of CSFD2. The user enters the desired values for each field.

Each of these fields is sent to CSFGL. Upon return, the cylinder, track, and record are validated to ascertain that they are all in the ranges of the device. Once that is assured, a decision is made whether this was a dump, display, or patch request; then the appropriate CCWs are constructed.

If it is a display request, the area requested is displayed on the terminal.

If it is a dump request, the dump is sent to the printer, in the format specified in the request. There are two formats for displaying or dump; the characters C or D for characters, X or a blank for hexadecimal.

If it is a patch request, the operator types in the patch as soon as the proceed light on the 1052-7 comes on.

When the requested operation is completed, CSFD1 is refetched. The test bit, previously set, directs the flow of execution to rereading the input message. If an EOB is found, control returns to the command recognition routine; otherwise, it waits for another input message.

Abridged Table Dump -- CSFAT

Function: The function of CSFAT is to output to SYS001 information about the current active task in an easily readable format (Chart BQ).

Entry Point: SYSATD is the sole entry point in CSFAT. CSFAT is fetched by CSFDU when the user enters DUM ATD.

Modules Called:

CSFGL - used to get the core locations of tables used by CSFAT (i.e., CHBSCN).

CSFWK - used to write diagnostic messages to the 1052-7 terminal.

Exits: CSFAT exits by issuing an SVC 6 to invoke CSFCD.

Operation: CSFAT operates by means of a series of BALS to internal subroutines. CSFAT initially determines whether SYS001 is assigned to a tape or printer and sets switches to perform the required I/O. If SYS001 is a tape, the output is in 133-byte records, the first of which is a machine control character for the printer. After printing a header, CSFAT scans the TSI and outputs the pertinent flags that are on (i.e., conv., in the wall, etc.).

If the XTSI for the current active task in in core, the current PSW, interrupt code, general registers, and control registers are taken from the XTSI and printed. This step is skipped if the XTSI is not in core.

CSFAT checks the current TSI and XTSI for other pertinent data (i.e., I/O op pending, pging op pending, current and last T/S value). Information from the TSI is always printed; information from the XTSI is printed if the XTSI is in core. If not, the entry is followed by the XTSI-out message (i.e., CURNT T/S VAL - XTSI OUT).

CSFAT uses the segment table to print out the page locations in real core and whether they are available. At the side of this page table a task interrupts pending table is printed. This table indicates whether any I/O, timer, async, external, program or SVC interrupts are pending. If any are pending, the number of such interrupts is printed, with a pointer to the first GQE for the specific type of interrupt.

Those entries in the core block table having the same userid as the current active task are marked user owned (X'BC') and have their block numbers, segment numbers and page numbers printed.

Finally, CSFAT dumps the scan table. The device number and the first and last GQE for each device is noted.

PSA Access Routine -- CSFPF

Function: Locate the prefix storage area (PSA) for the CPU requested (Chart BR).

Entry Point: Entry point to CSFPF is GETPFA. On entry general register 1 contains CPU# whose PSA it desires.

Exit: Return is made via a BR14, also passing as return codes:

GR 0 Low-Order-Byte Contents	Meaning
X'00'	GR1 contains address of requested prefix area.
X'01'	GR1 contains 0 since no CPU status table exists.
X'02'	GR1 contains FFFFFFFF, since GR1 contained an invalid CPU number.
X'04'	GR1 contains FFFFFFFF, since CPU is not available in this installation.
X'08'	GR1 contains prefix address, but CPU is not available due to malfunction.
X'10'	GR1 contains FFFFFFFF, since CPU is partitioned out of TSS domain.
X'20'	GR1 contains FFFFFFFF, since CPU is not entered in status table.
X'40'	GR1 contains FFFFFFFF, since status table indicates neither prefix area is active.

Modules Called: None

Operation: It is determined whether CPU1 or CPU2's PSA is requested. The corresponding ID is searched for in the CST (CPU status table) until a match is obtained, then the address of the active PSA (i.e., the prime or alternate) is placed into register 1, and an appropriate return code is placed in register 0.

APPENDIX A: Flowcharts

The charts in this section are identified in alphabetic sequence and appear in the same order in which they are discussed in the text.

Chart AA. Supervisor - Initialization (CSFSU) (Part 1 of 5)

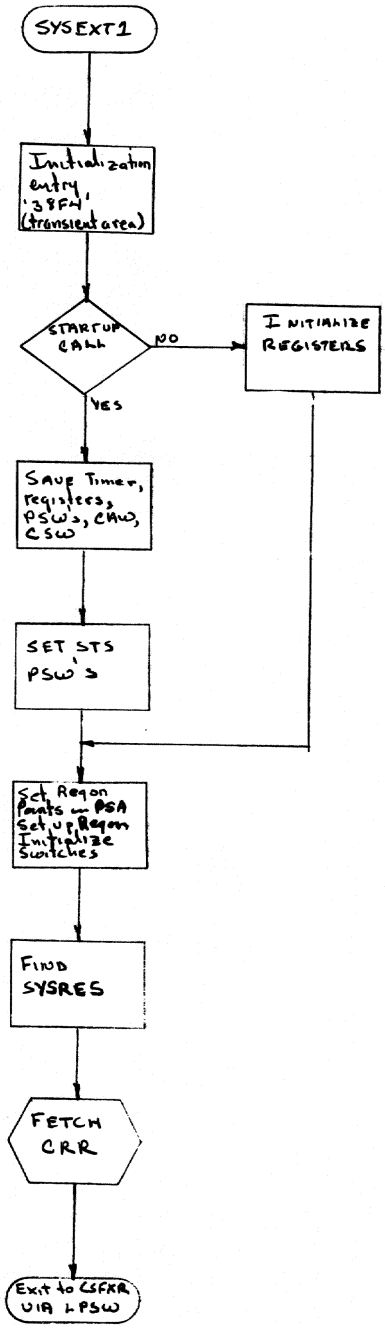


Chart AB. Supervisor-Program Interrupt (CSFSU) (Part 2 of 5)

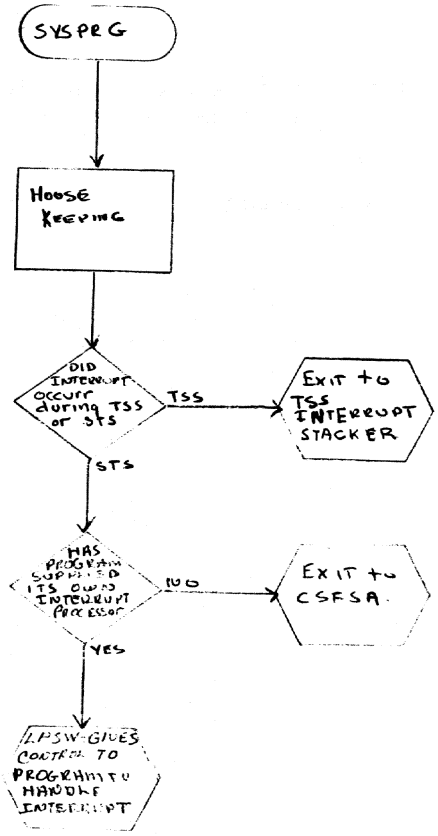


Chart AC. Supervisor-Extrnl. Inter. Proc. (CSFSU) (Part 3 of 5)

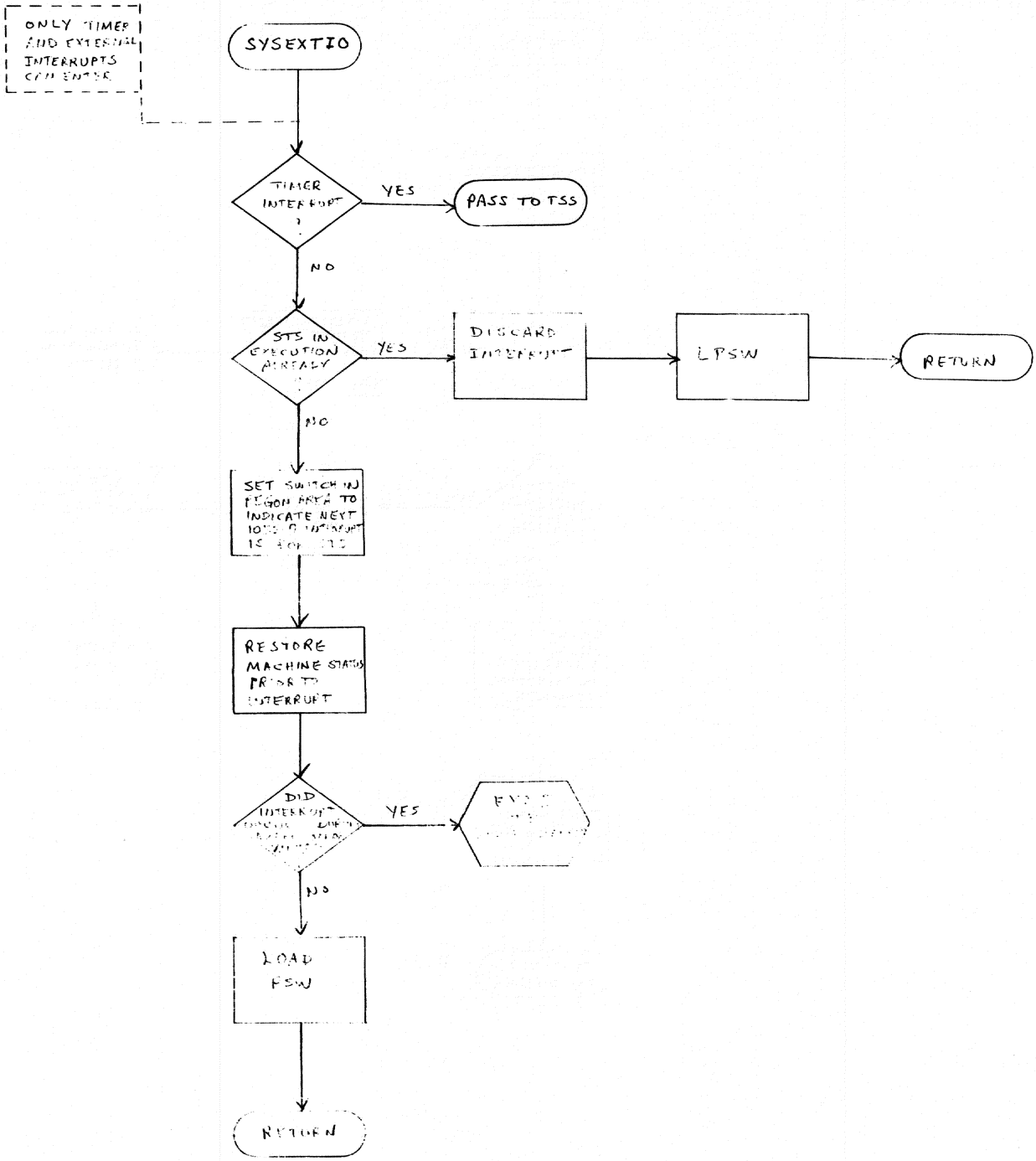


Chart AD. Supervisor-I/O Inter. Proc. (CSFSU) (Part 4 of 5)

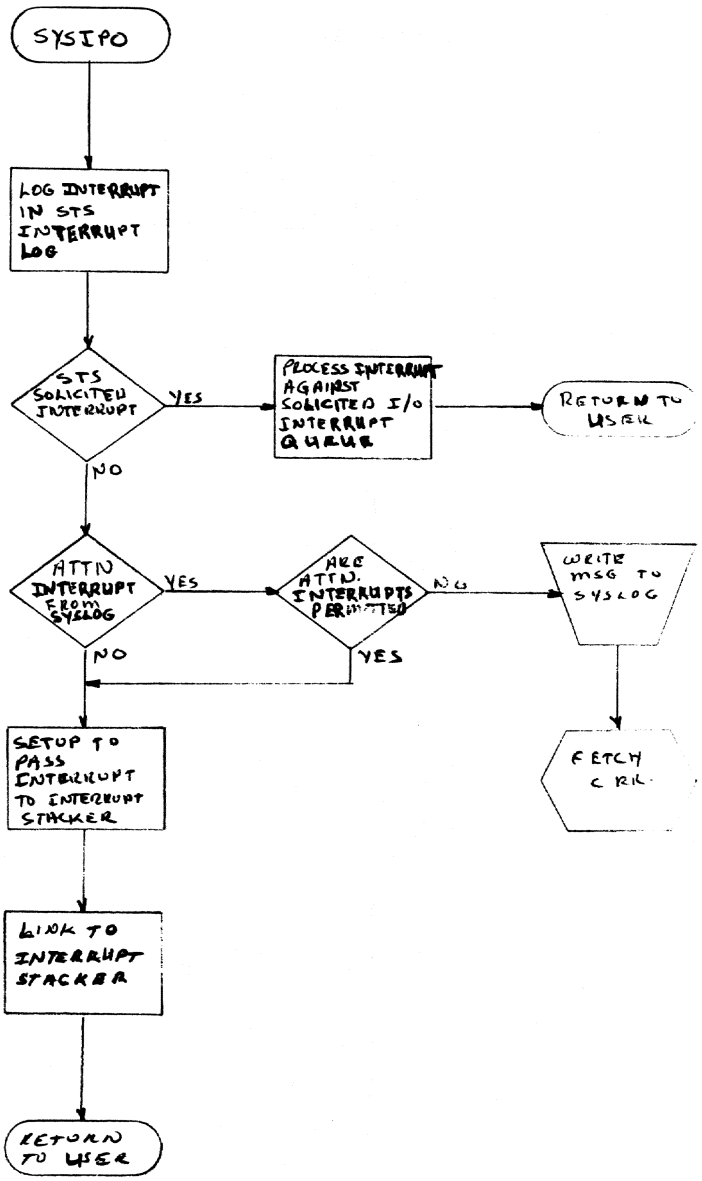


Chart AE. Supervisor-SVC Inter. Proc. (CSFSU) (Part 5 of 5)

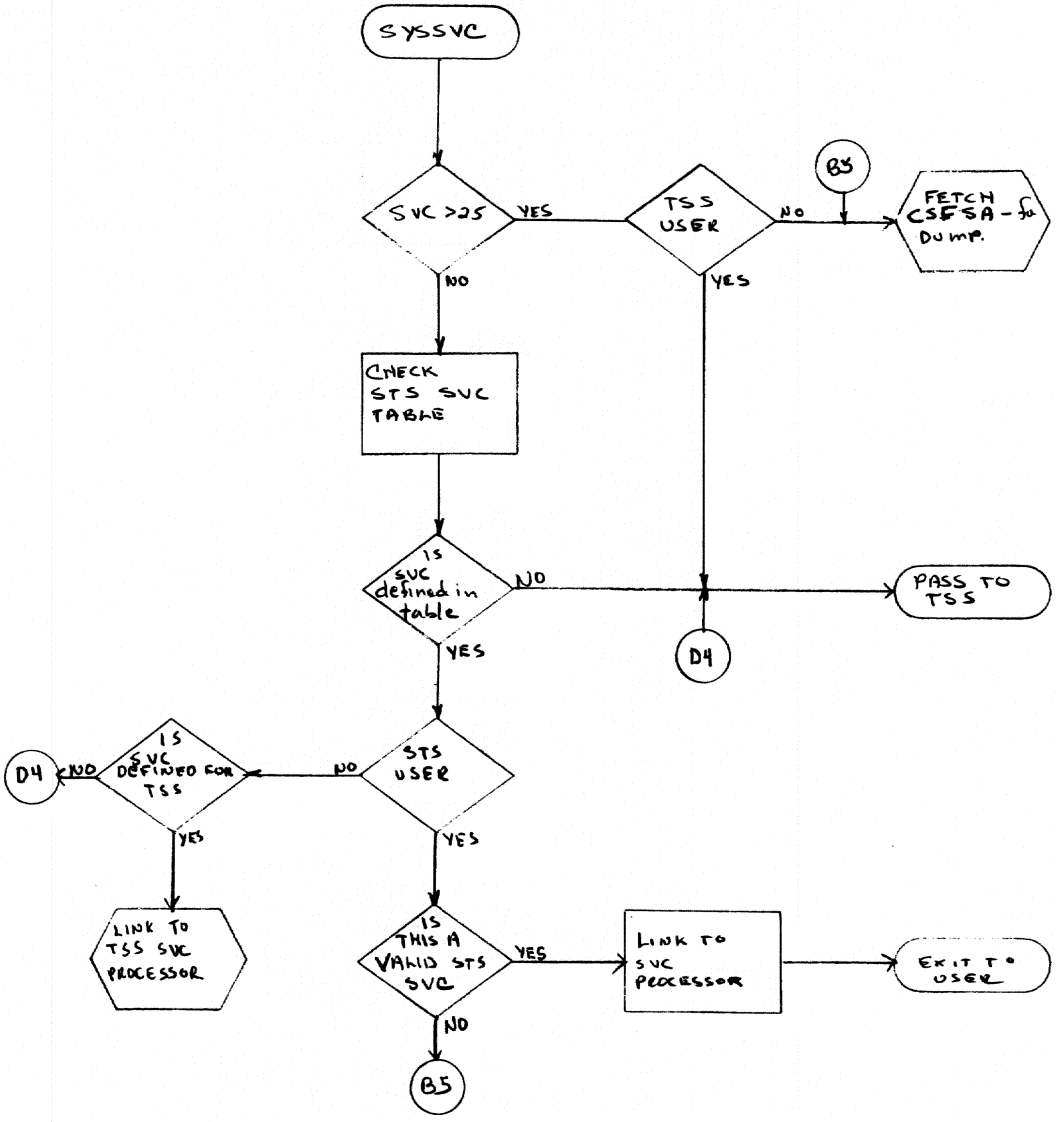


Chart AF. Stand Alone Dump (CSFSA)

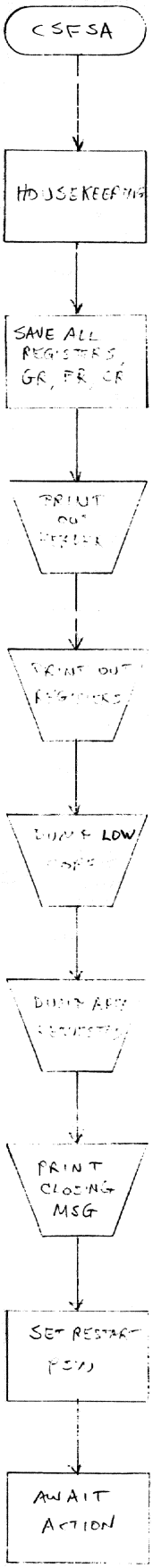


Chart AG. Command Recognition Routine (CSFCD)

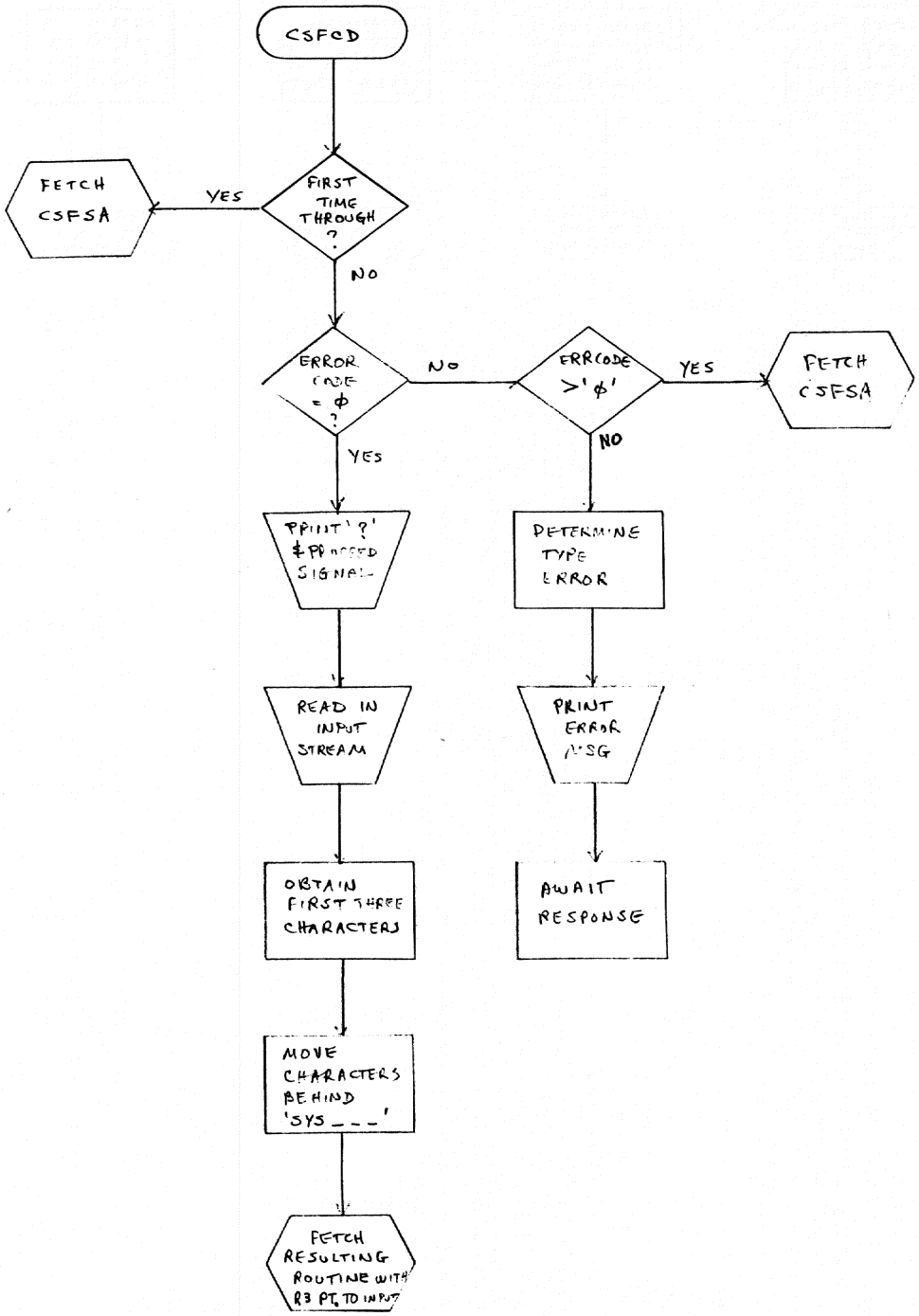


Chart AH. Pool Routines

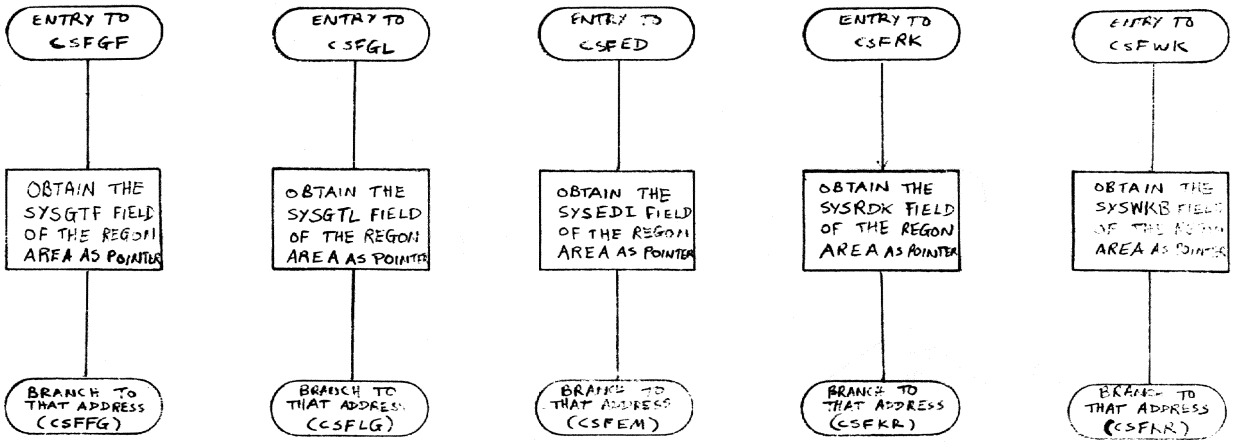


Chart AI. Edit Routine (CSFEM)

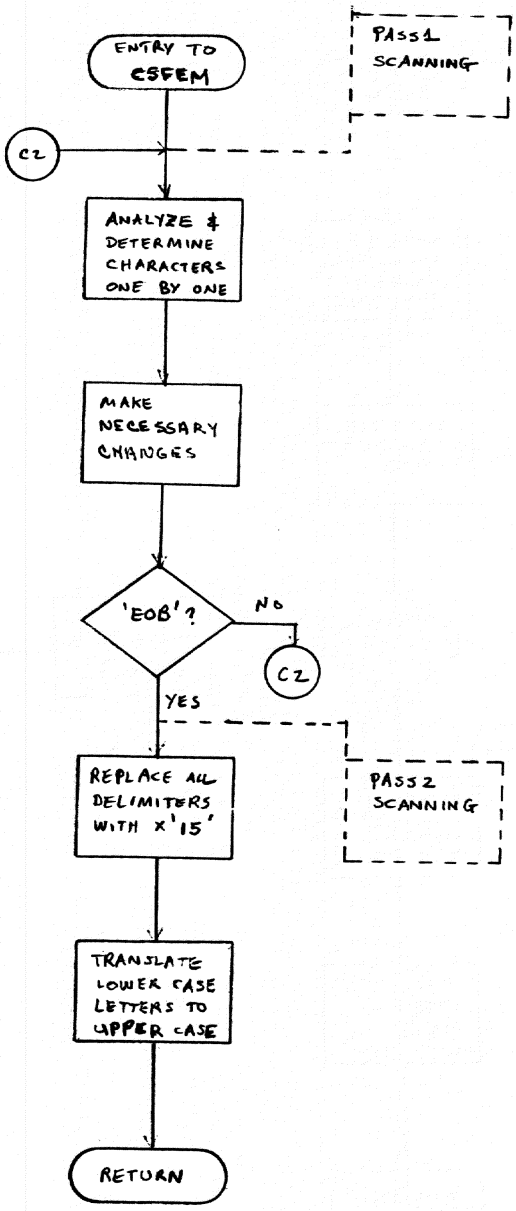


Chart AJ. GET Message Subroutine (CSFFG)

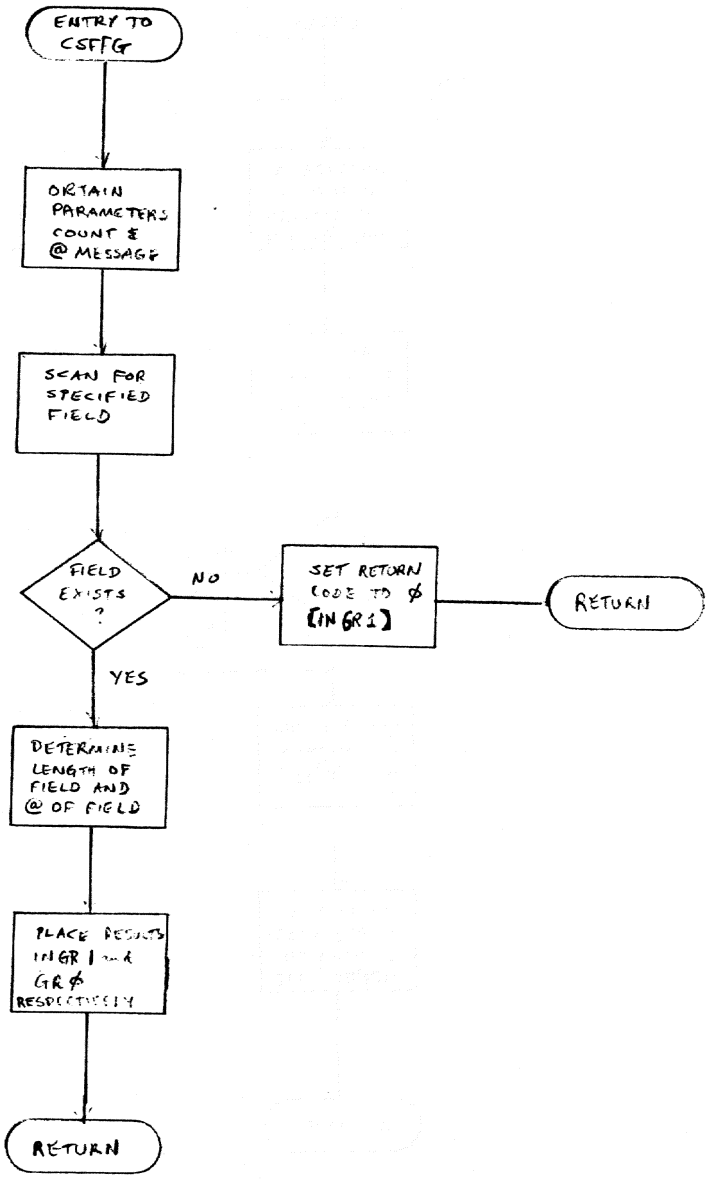


Chart AK. READ/WRITE Keyboard (CSFKR)

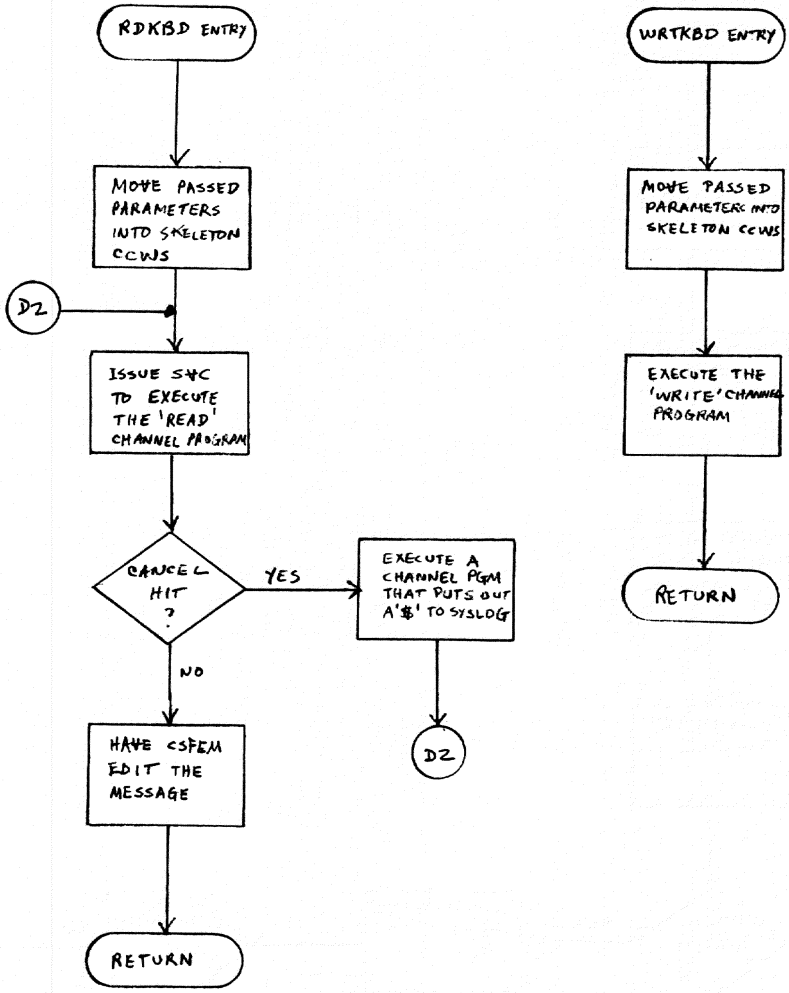


Chart AL. GET Absolute Core Location (CSFLG)

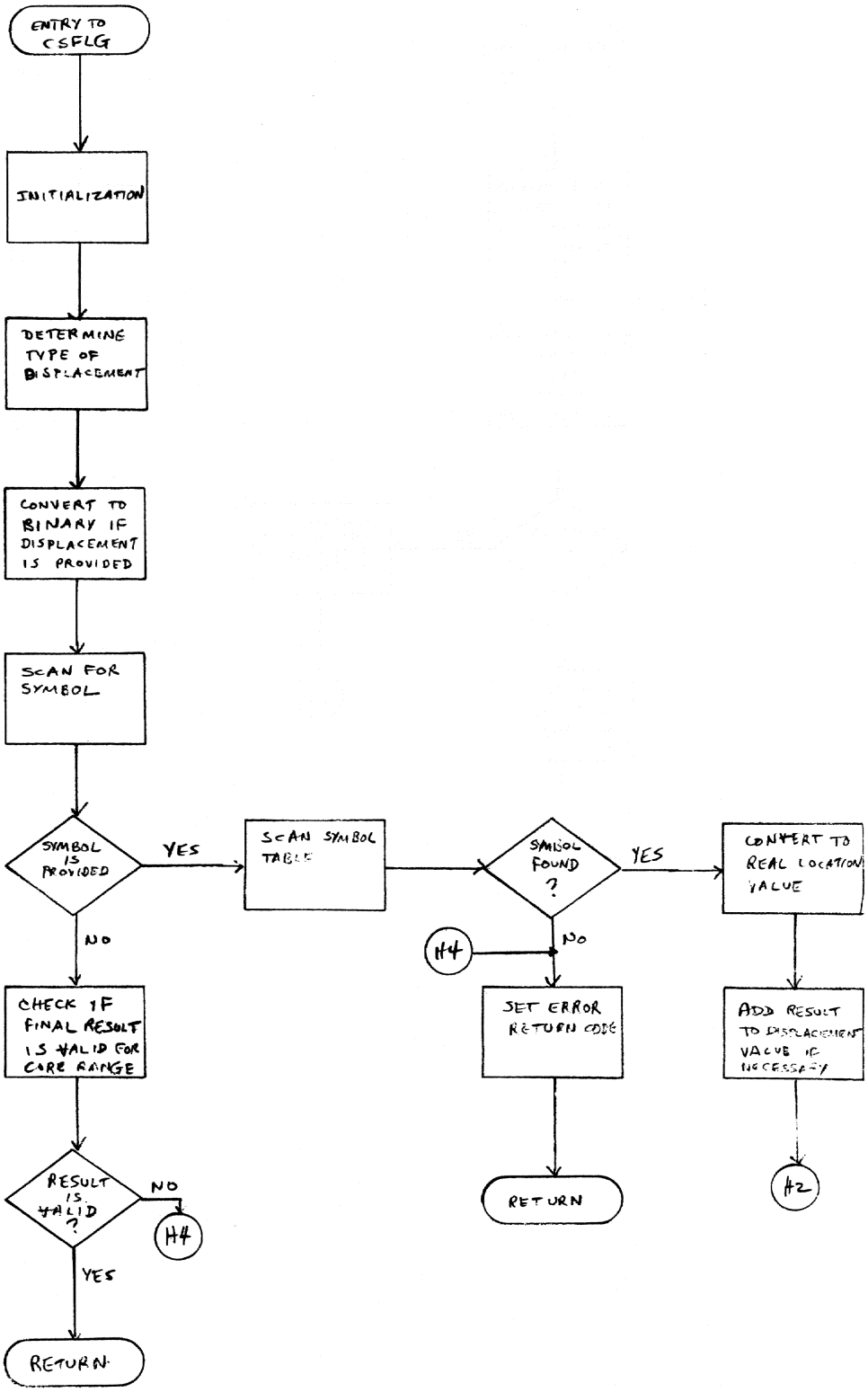


Chart AM. Test Validity of Address (CSFAD)

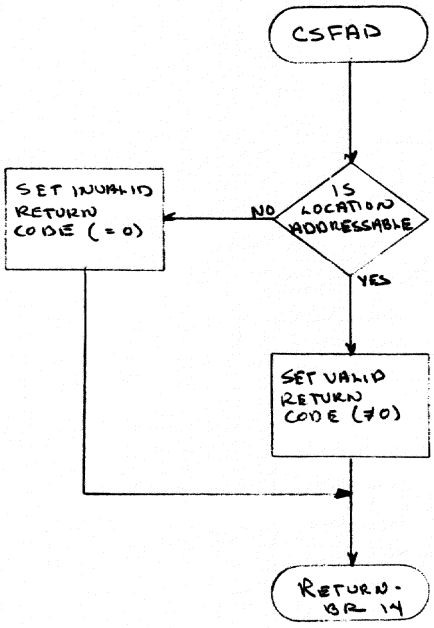


Chart AN. Assign Device Address (CSFAS)

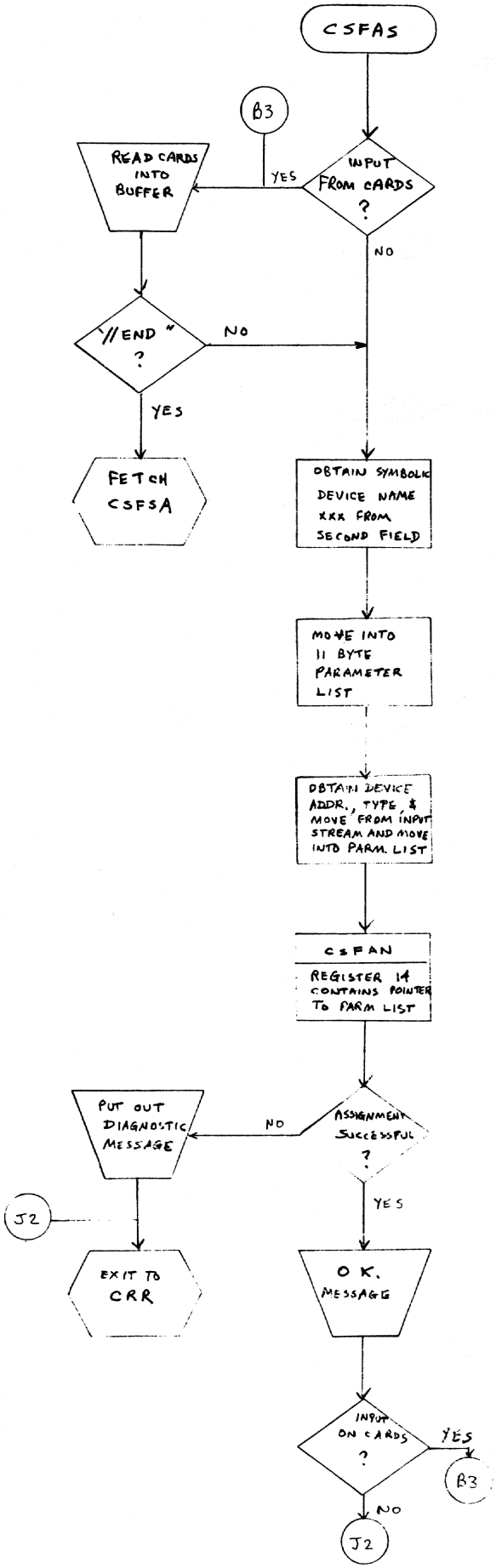


Chart AO. Assign Device Address Subr. (CSFAN)

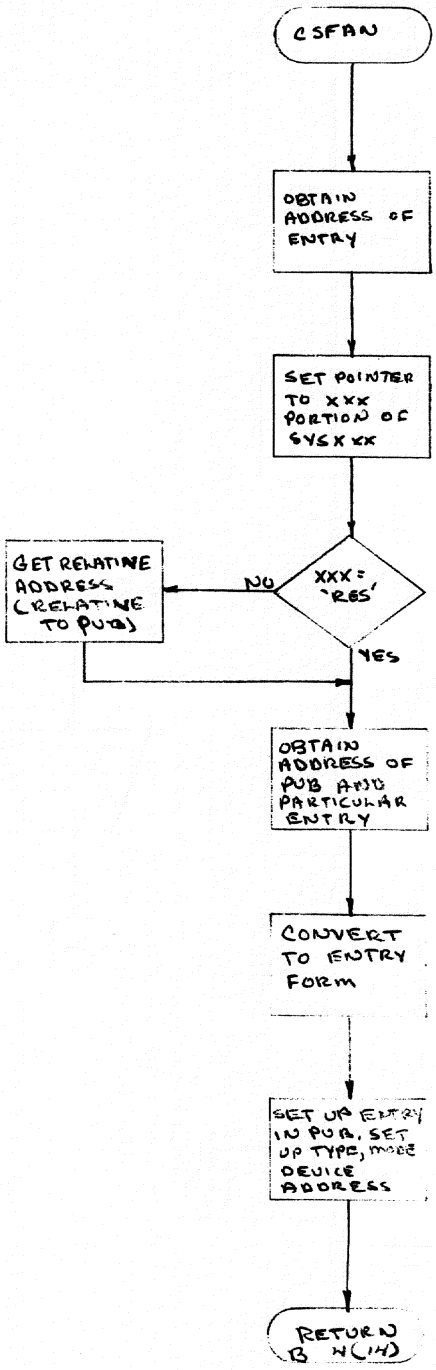


Chart AP. Display Pub. Routine (CSFPB)

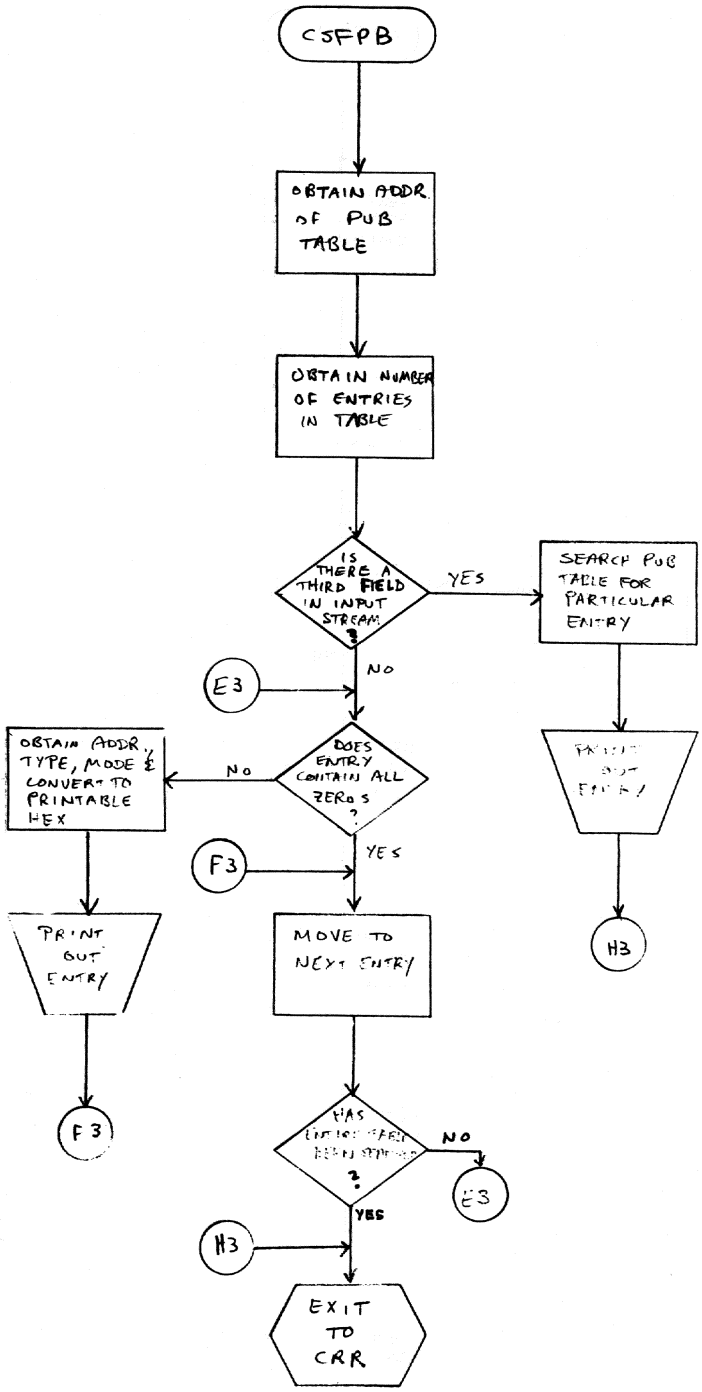


Chart AQ. RUN Command (CSFRU)

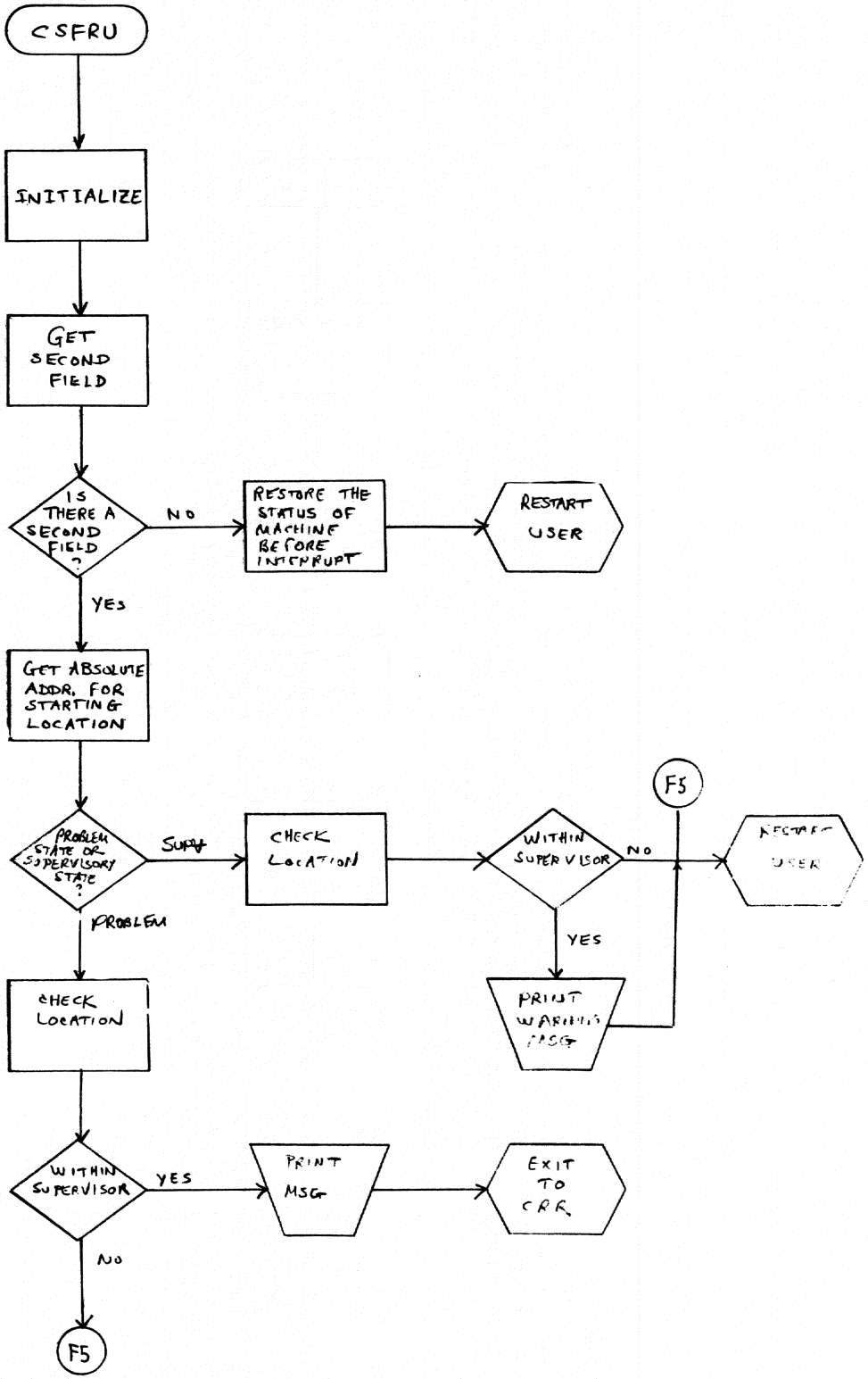


Chart AR. LINK (CSFLN)

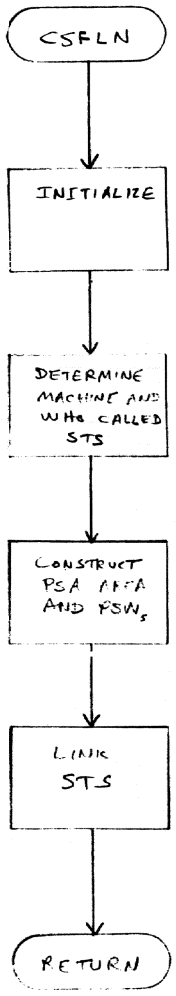


Chart AS. Exerciser (CSFXR)

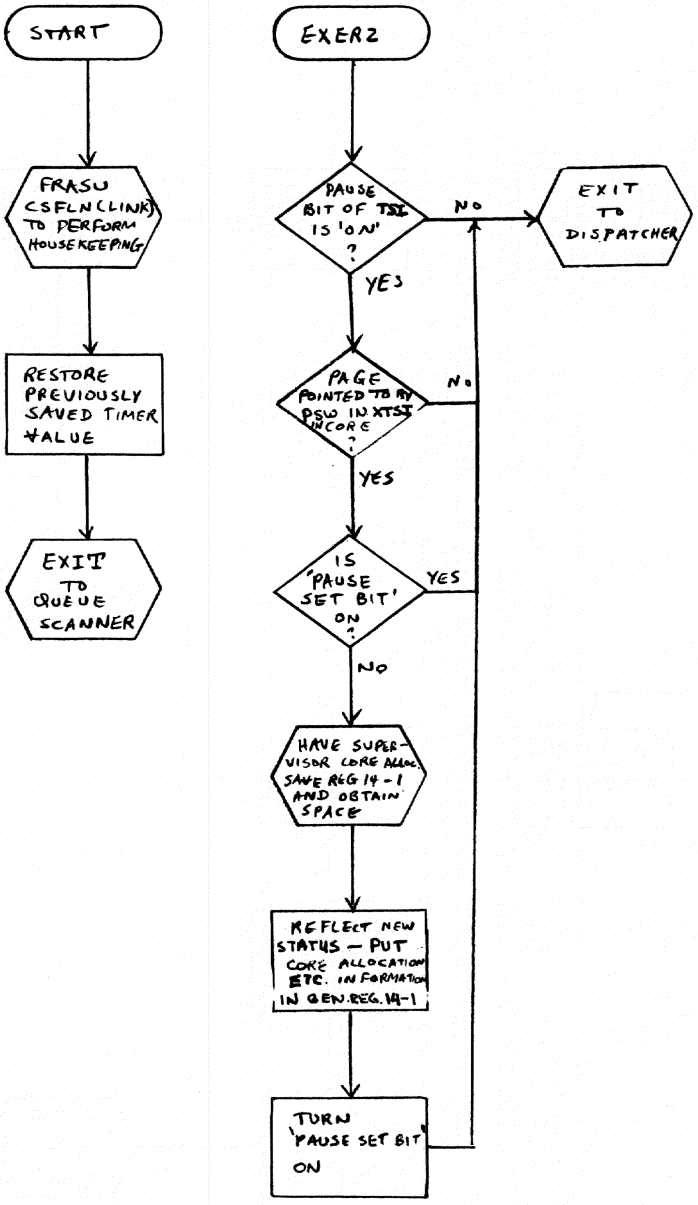


Chart AT. Virtual Memory Processor (CSFVM)

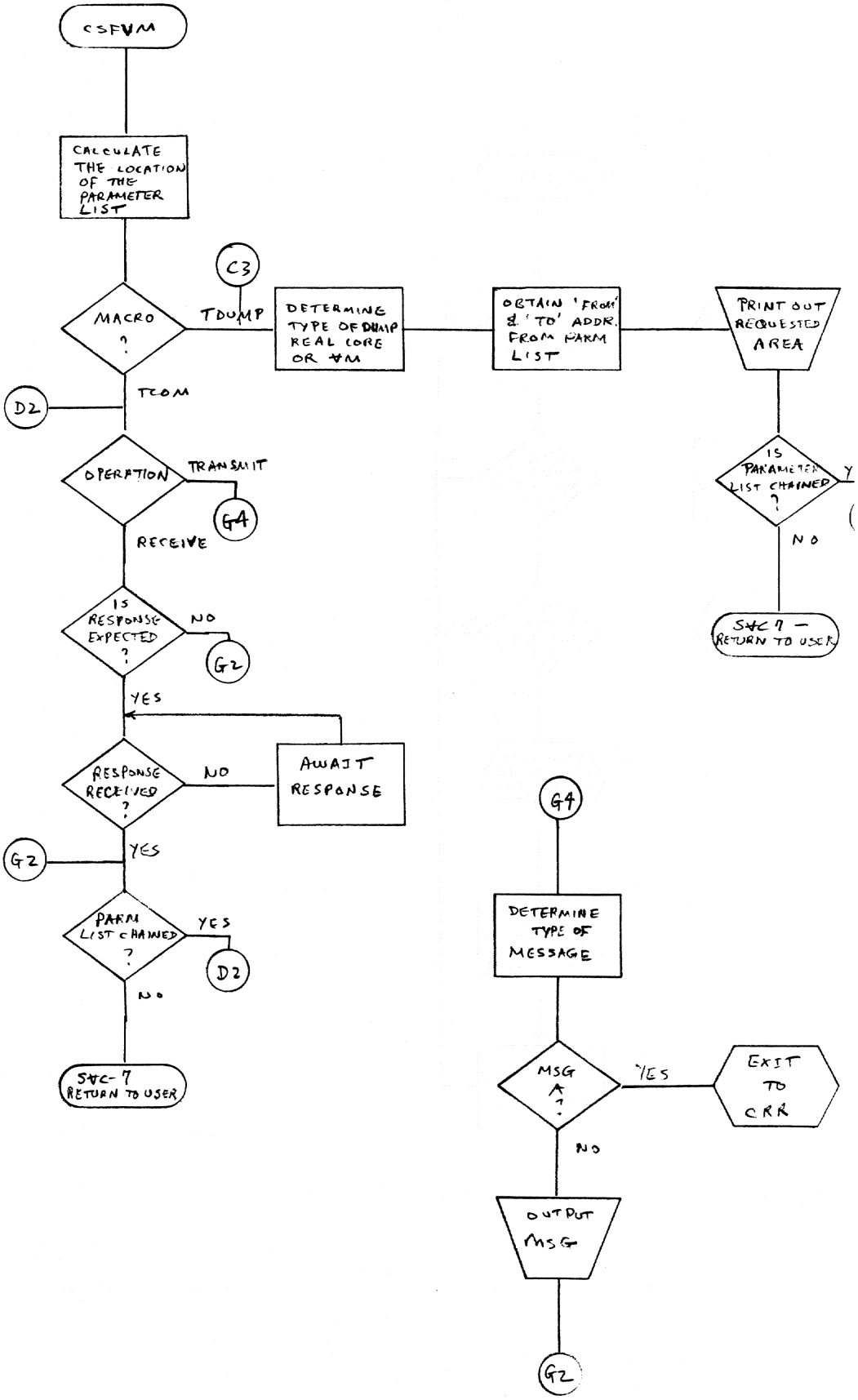


Chart AU. FIND Page (CSFFP)

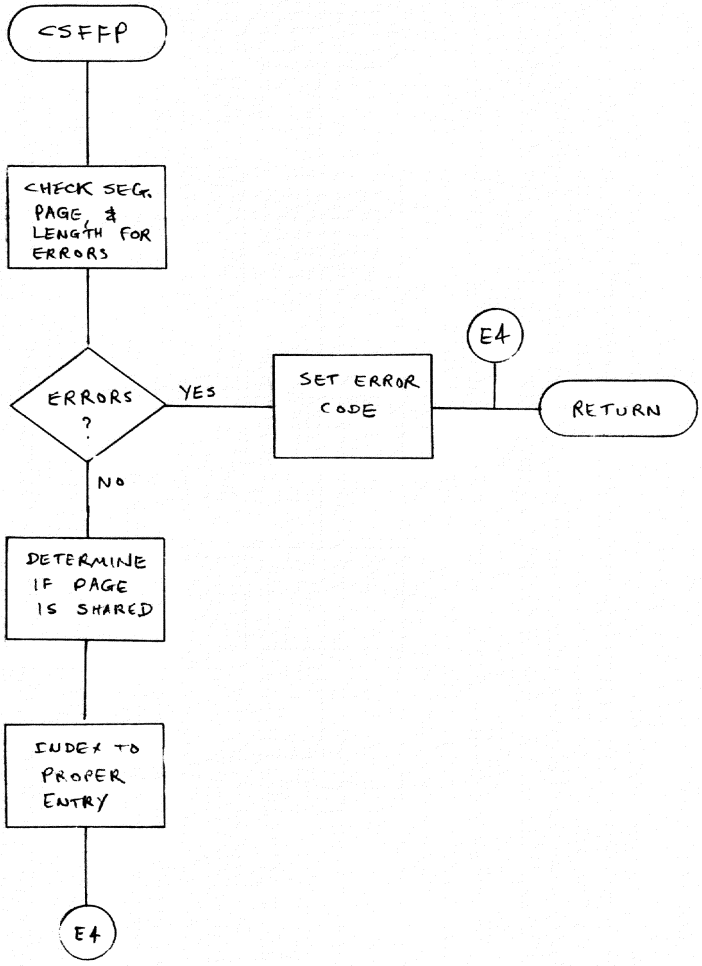


Chart AV. Convert A Symbolic VM Addr. (CSFTD)

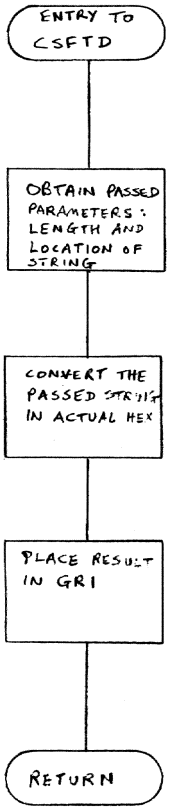


Chart AW. Locate VM Page (CSFPL)

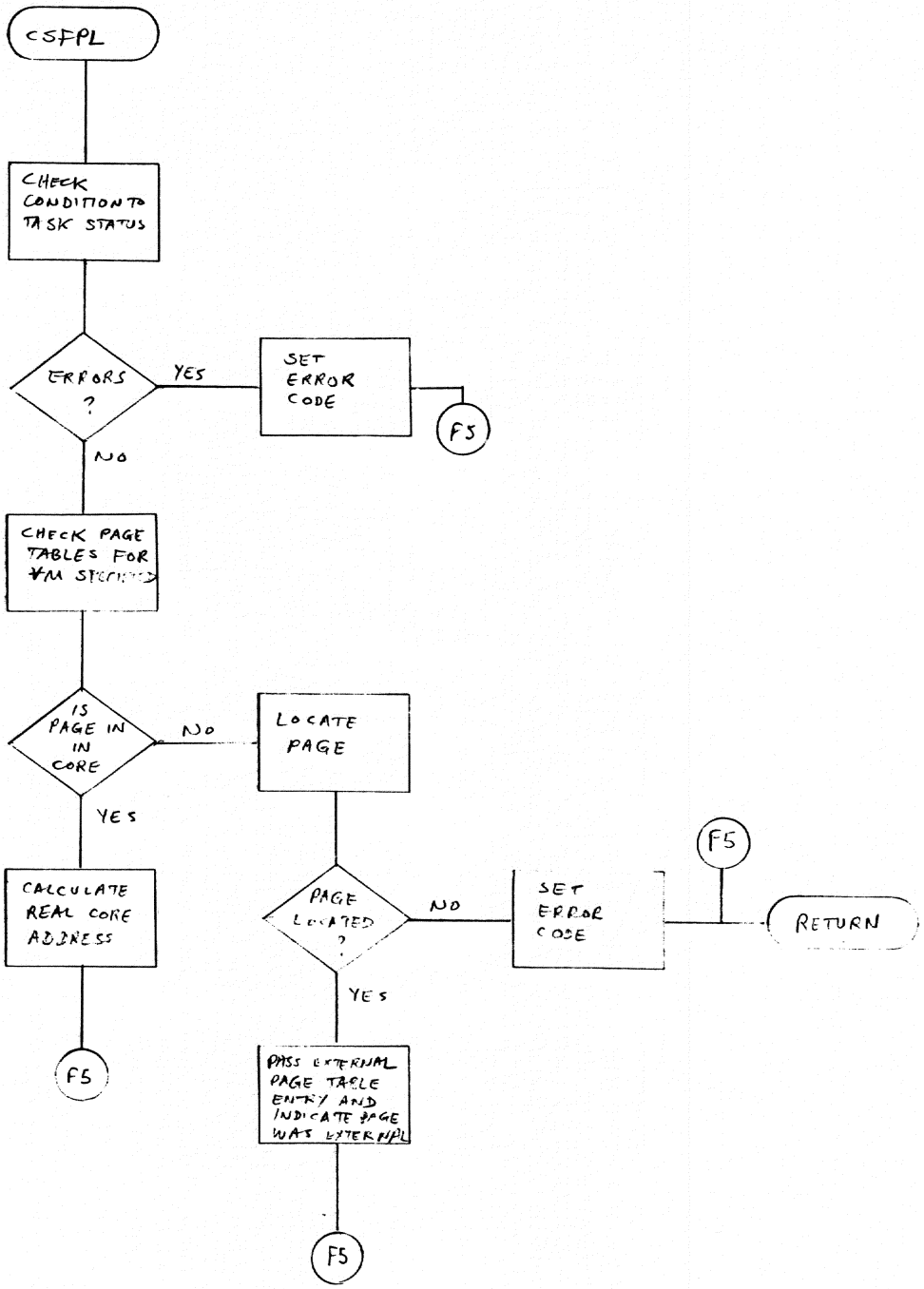


Chart AX. Get VM Page (CSFPG)

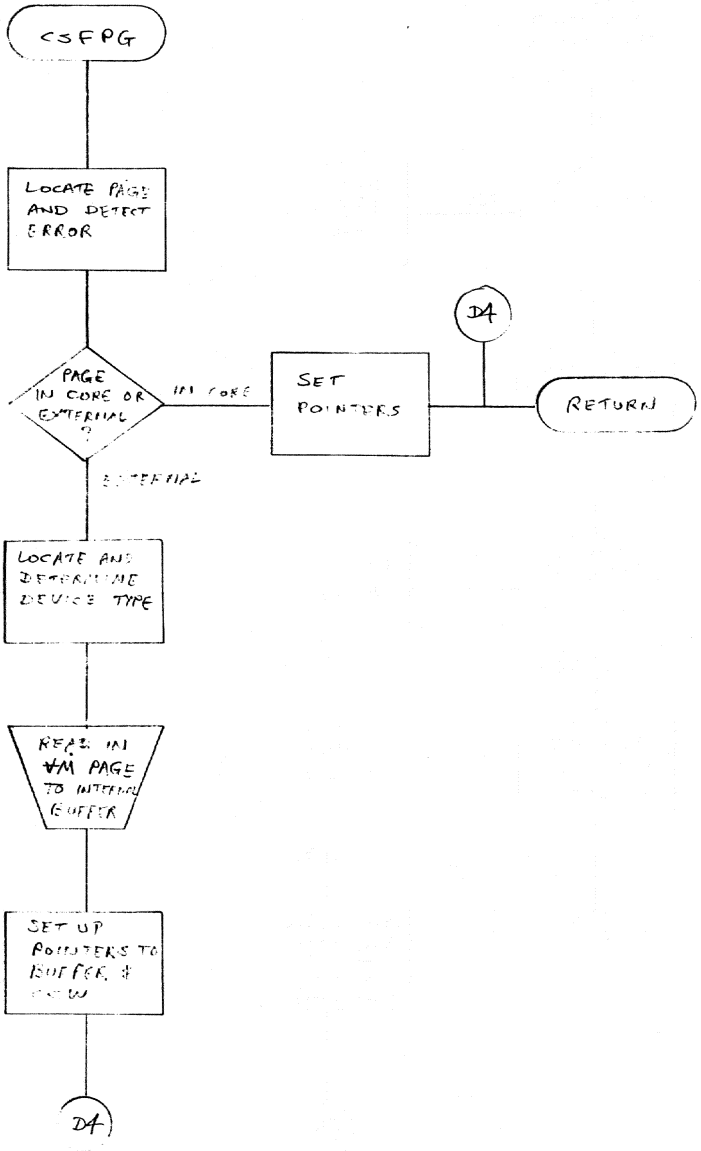


Chart AY. Buffered Page GET (CSFBP)

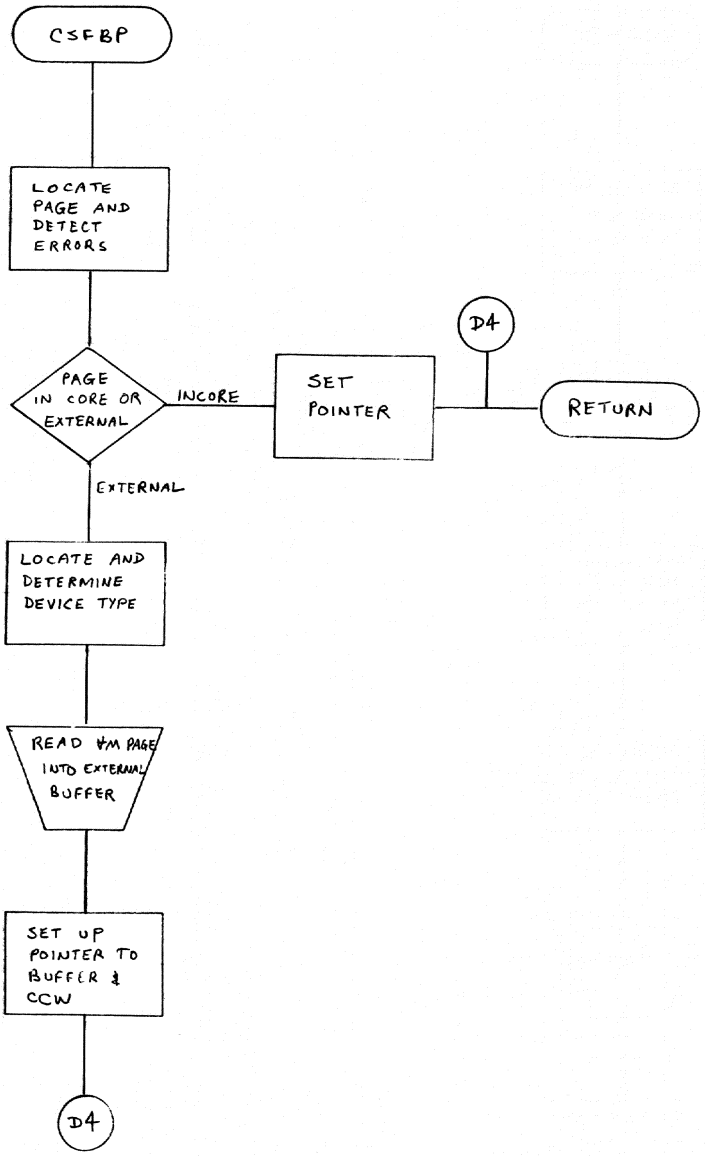


Chart AZ. DUMP Block (CSFDB)

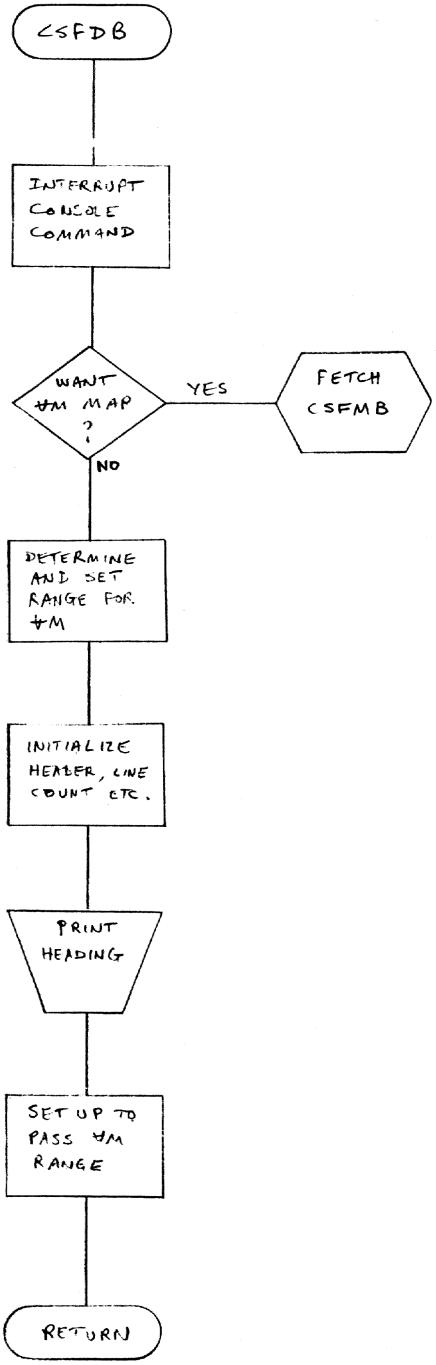


Chart BA. Patch Real Core Processor (CSFPA)

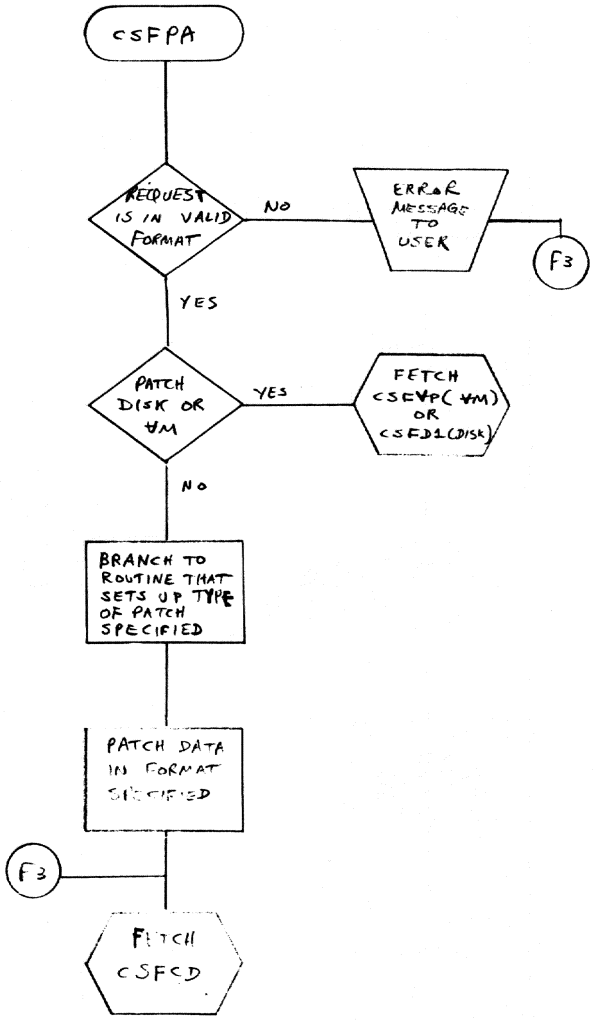


Chart BB. Patch Virtual Memory Proc. (CSFVP)

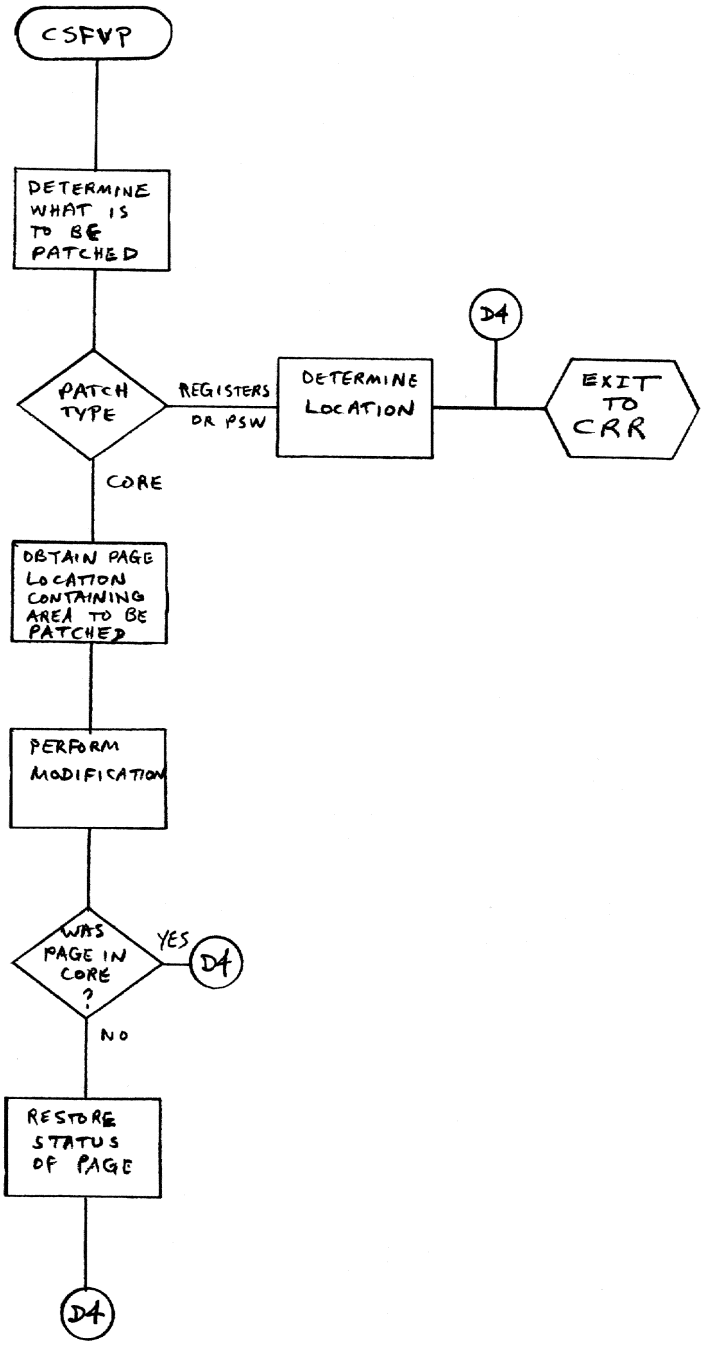


Chart BC. Snapshot Processor (CSFDR)

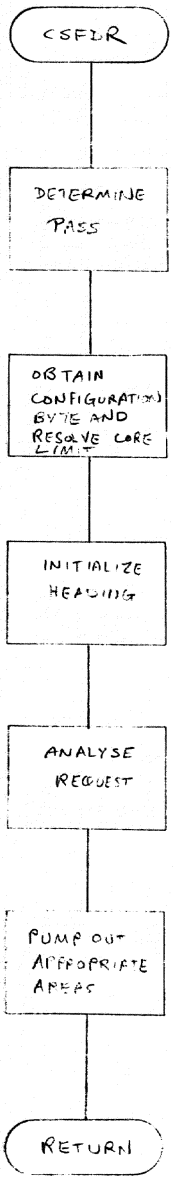


Chart BD. DELETE (CSFDE)

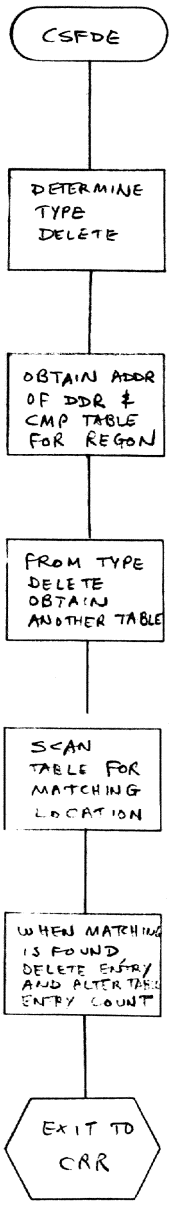


Chart BE. Dump Out Processor (CSFDO)

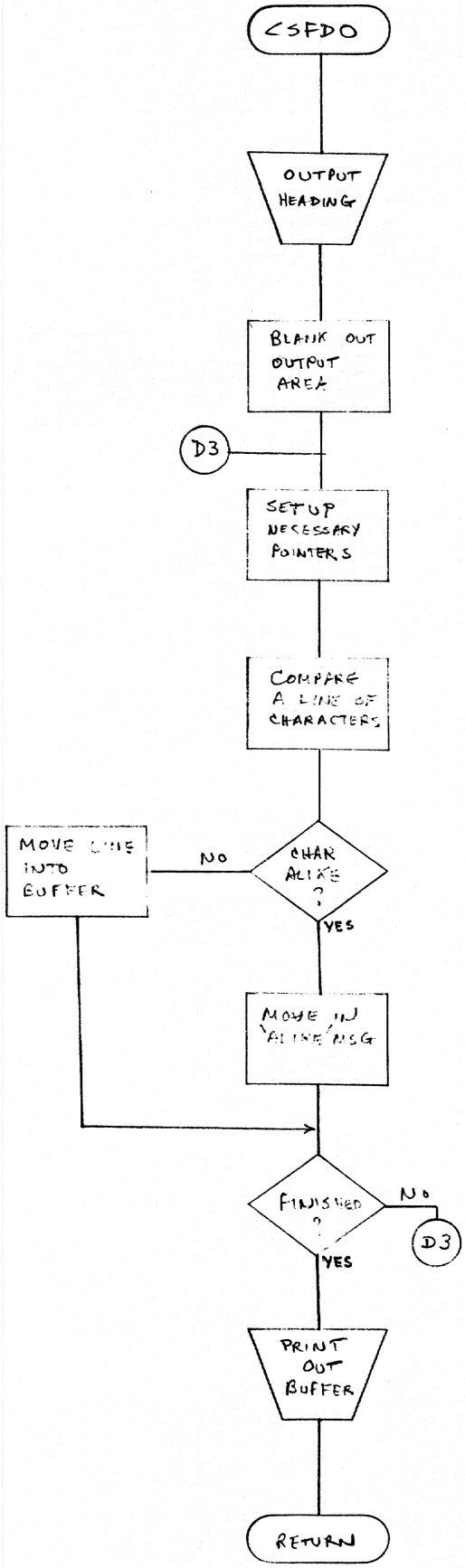


Chart BF. Dump Command Routine (CSFDU)

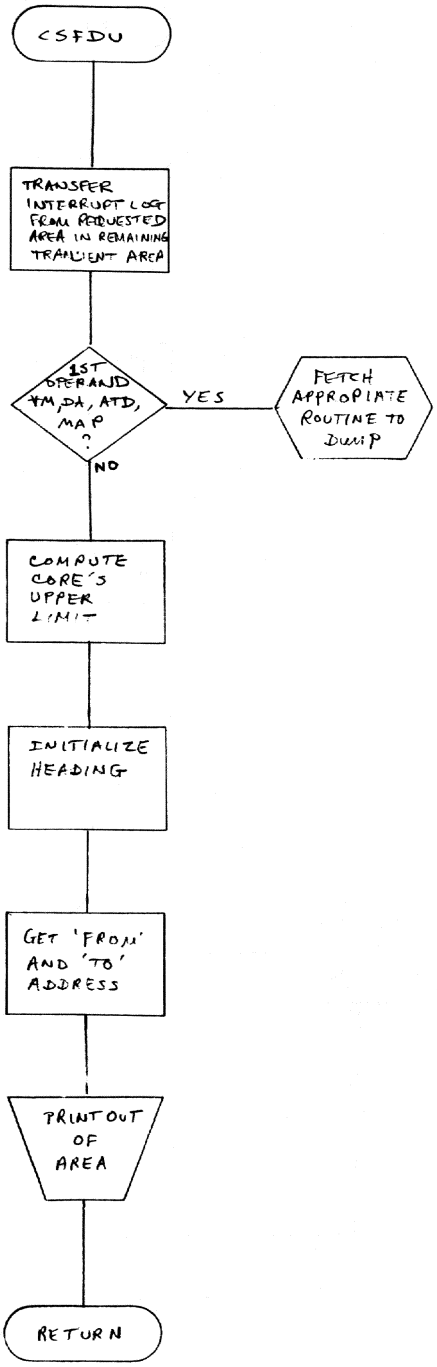


Chart BG. Numeric Symbol Sort Proc. (CSFSS)

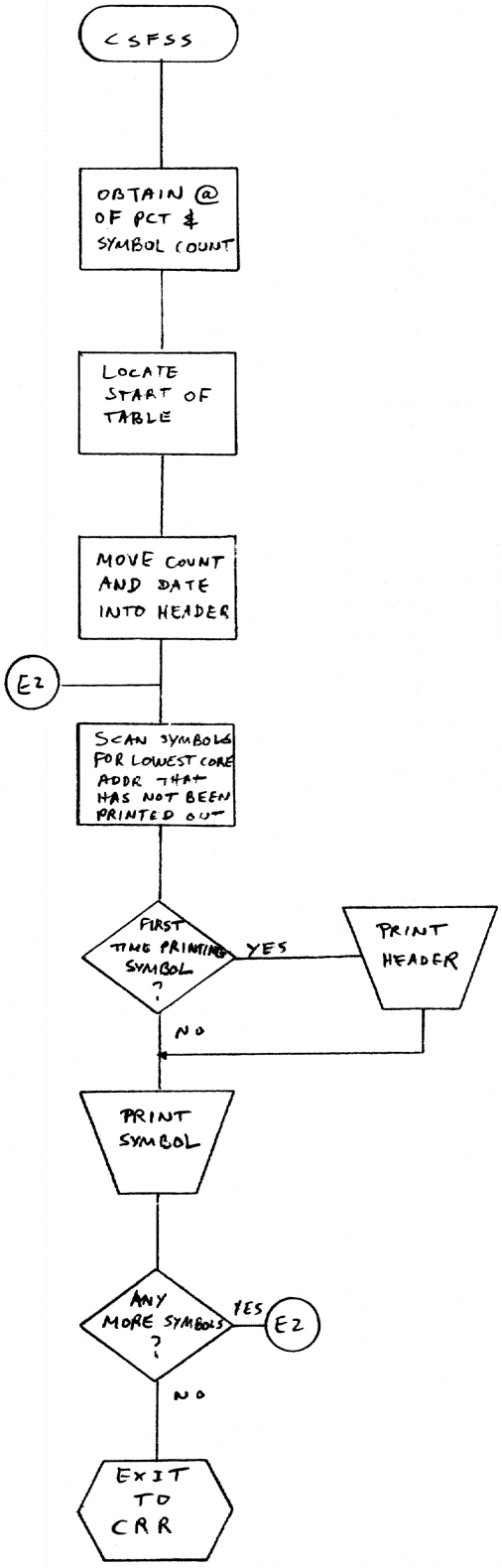


Chart BH. Virtual Memory Dump (CSFVU)

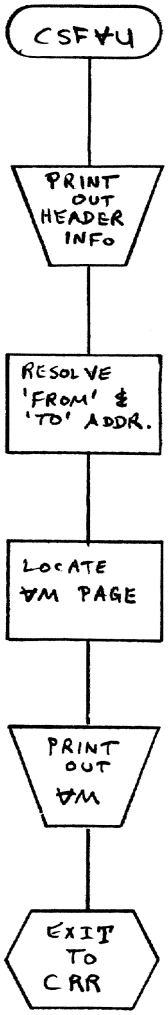


Chart BI. Dump VM Map (CSFMB)

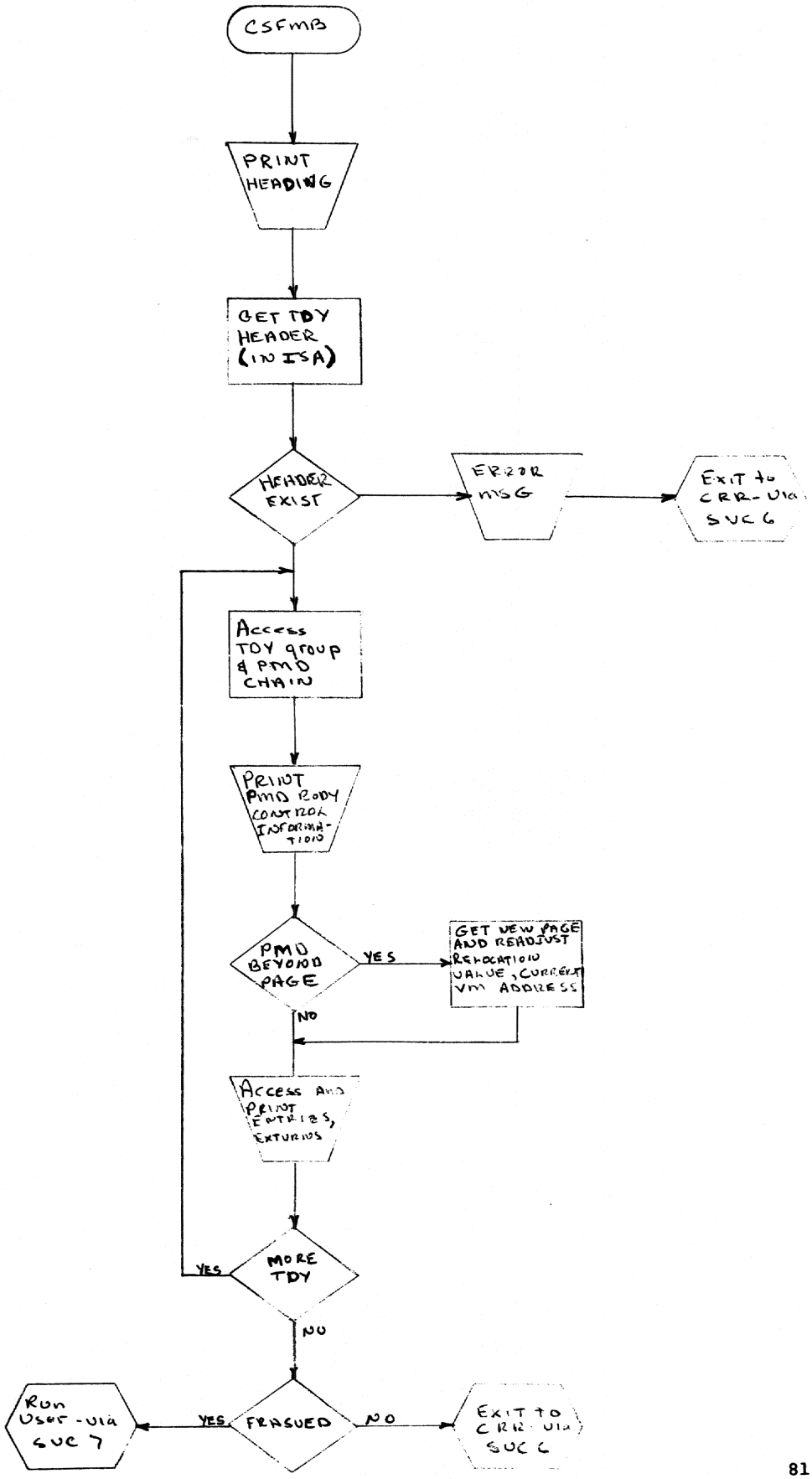


Chart BJ. Display Real Core (CSFDI)

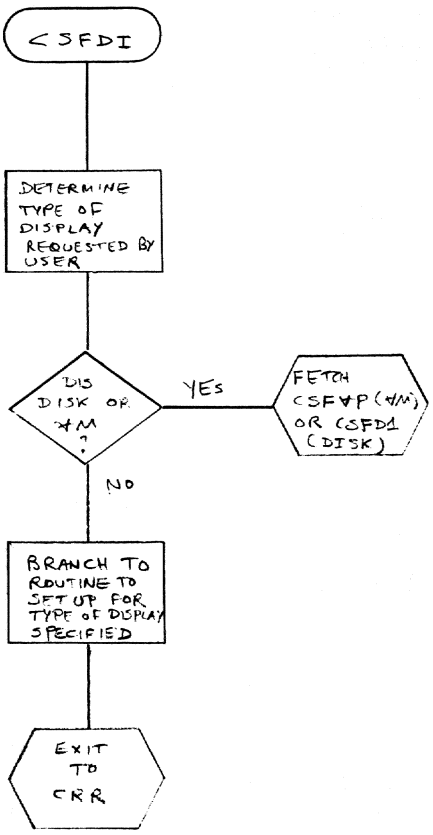


Chart BK. Display Virtual Memory Proc. (CSFVD)

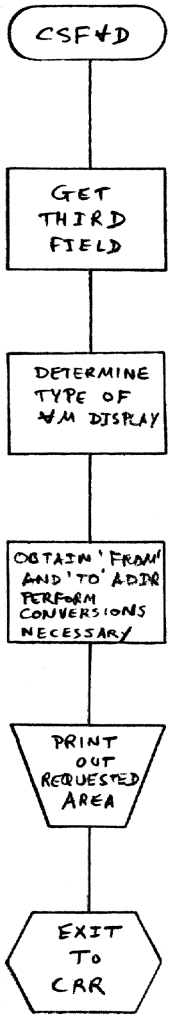


Chart BL. Set Control Function Status (CSFSE)

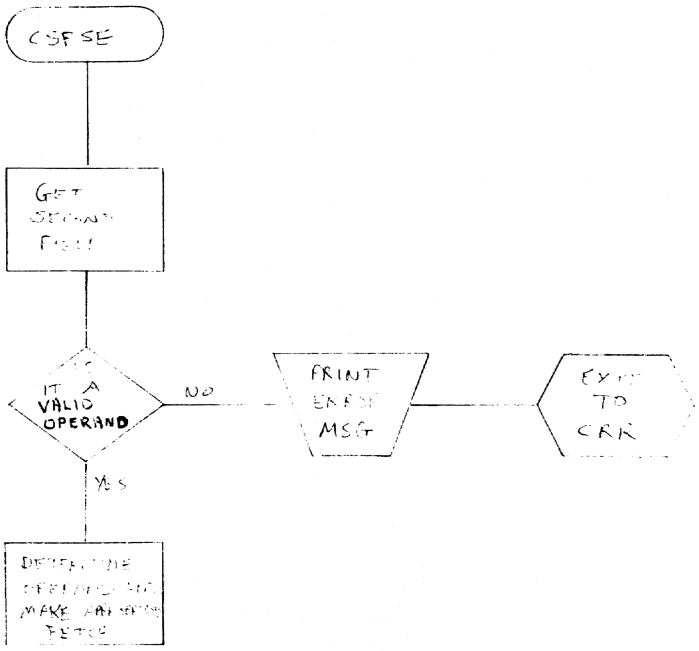


Chart BM. Set Pause Bit Processor (CSFSP)

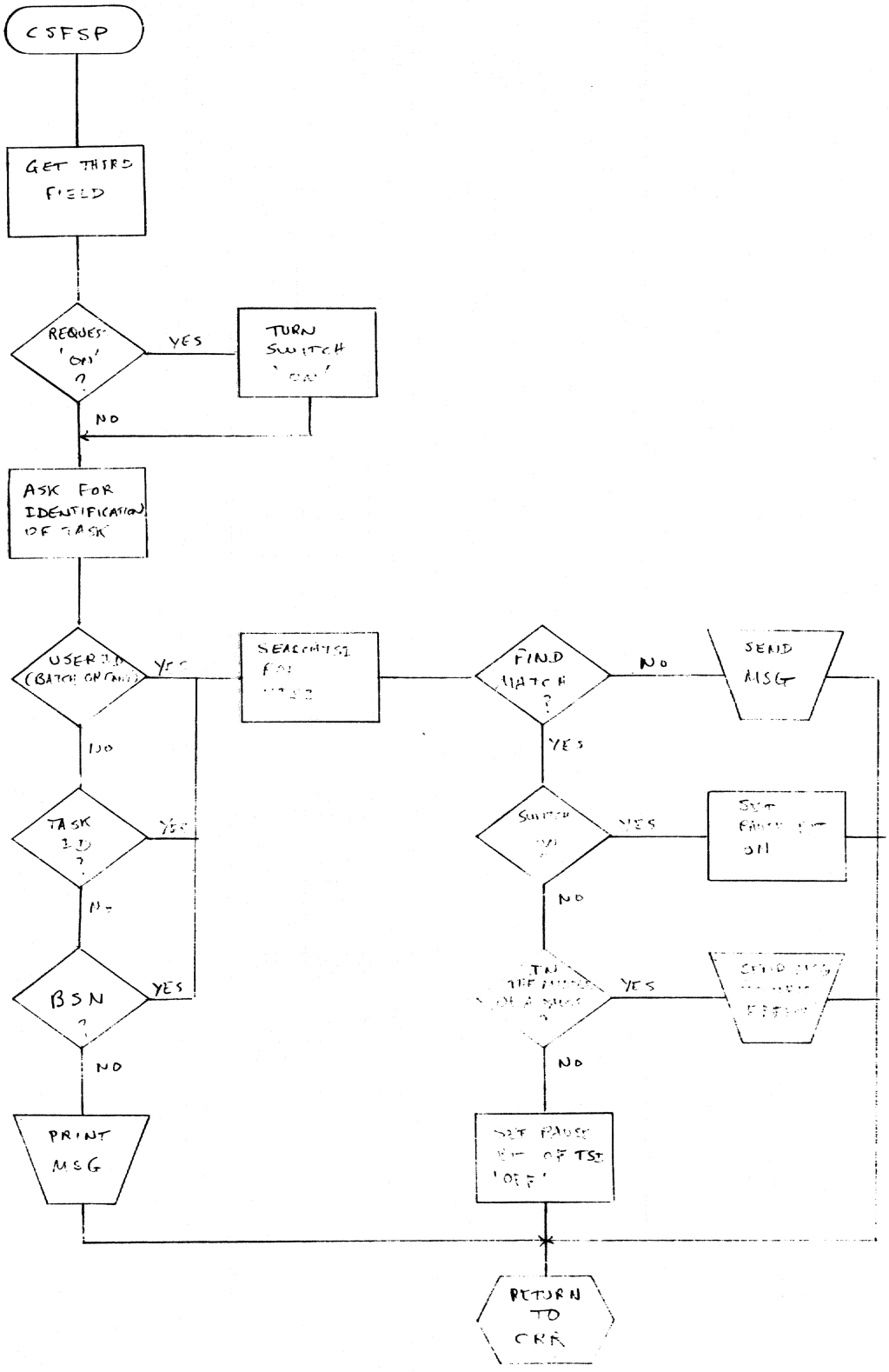


Chart BN. Pause (CSFPZ)

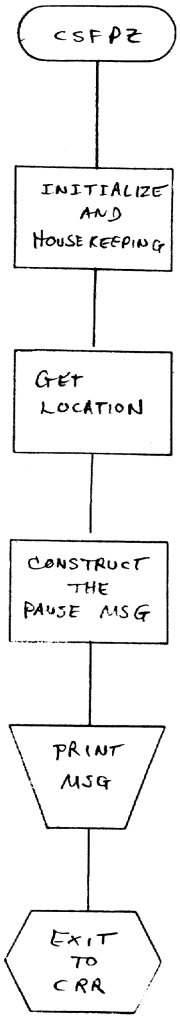


Chart BO. Dump/Display/Patch Disk (CSFD1)

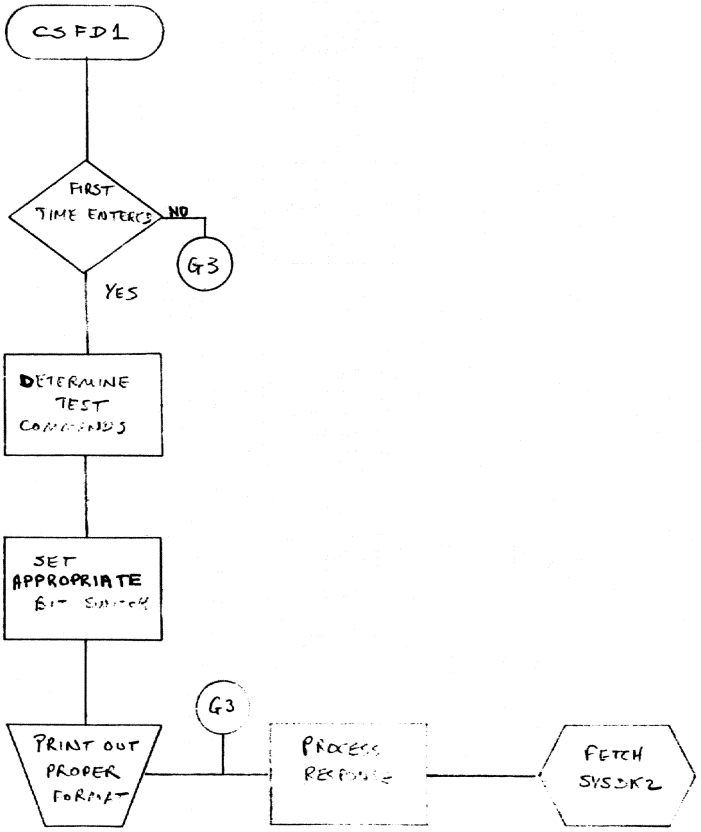


Chart BP. Dump/Display/Patch Disk (CSFD2)

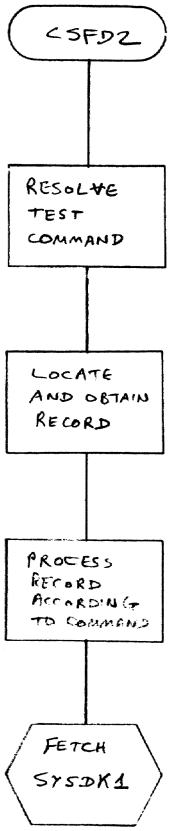


Chart BQ. Abridged Table Dump (CSFAT)

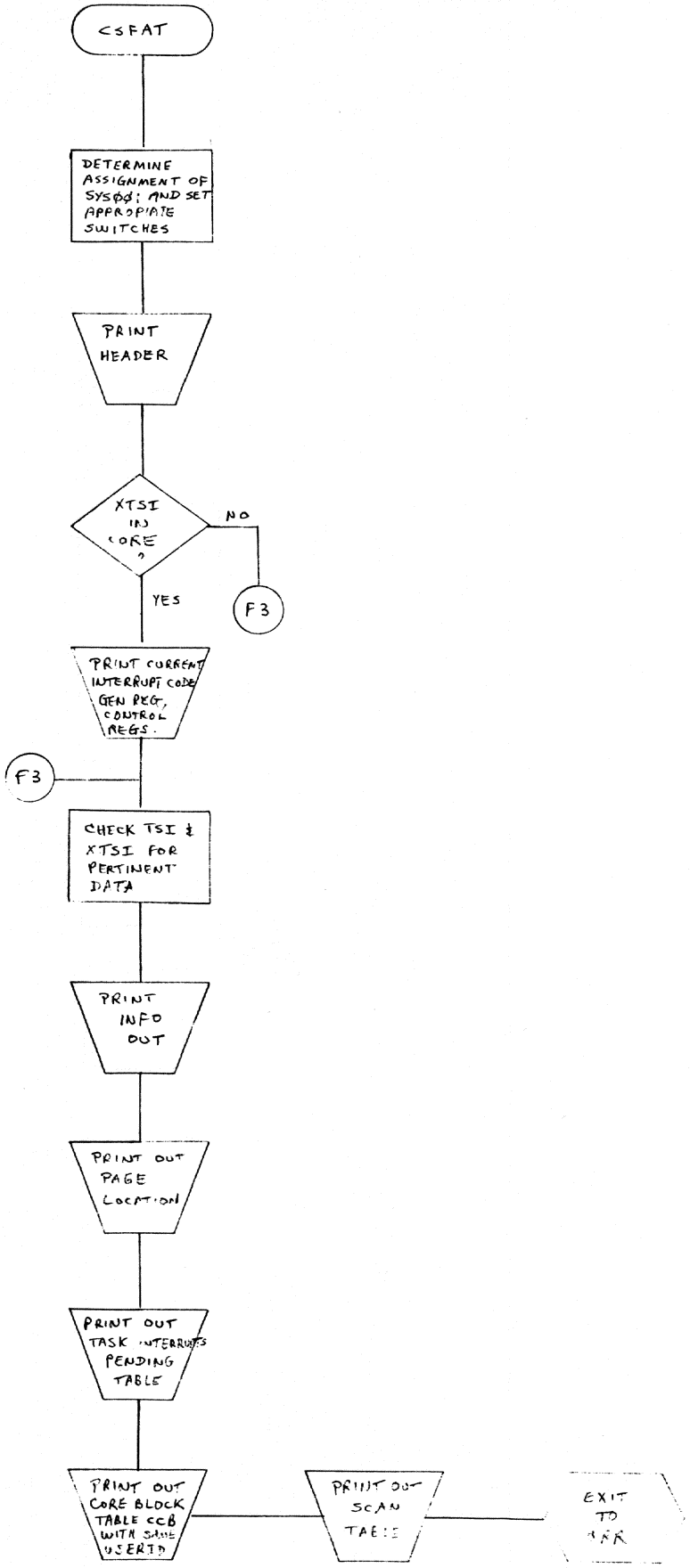
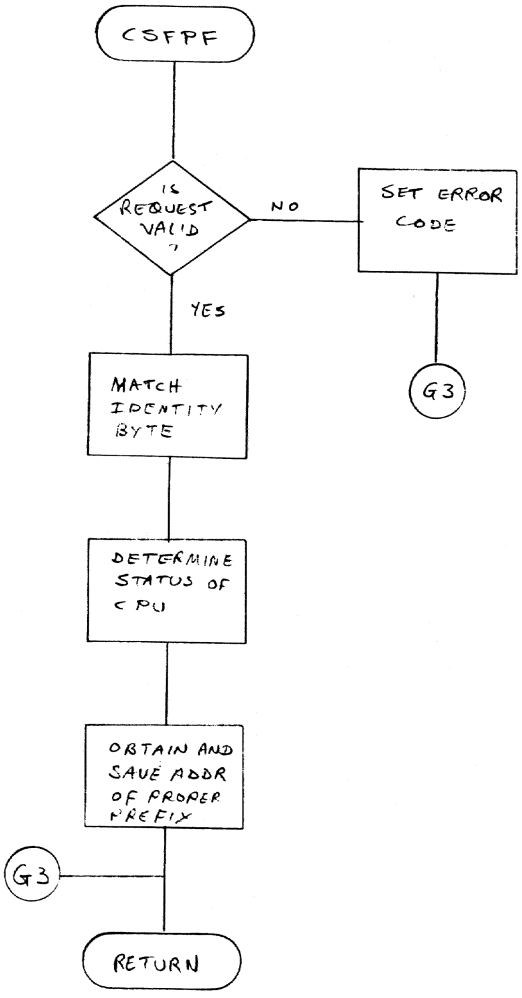


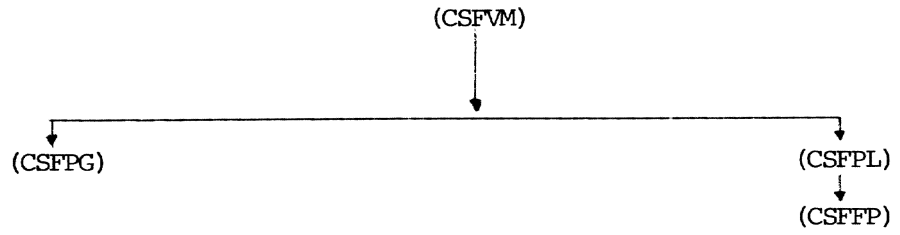
Chart BR. PSA Access Routine (CSFPF)



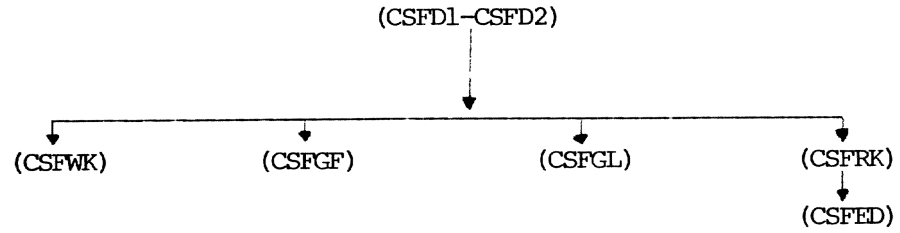
APPENDIX B: LAYOUT OF STS PHASES

This appendix depicts a mapped layout of the phases comprised by STS. A phase is a collection of object modules which have been link-edited together in core-image form with one primary entry point defined. Phases are normally loaded into core via the Fetch mechanism.

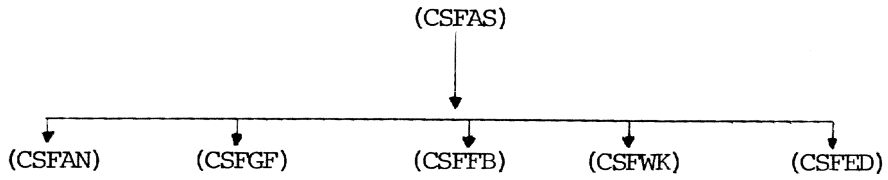
Phase SYSVMP



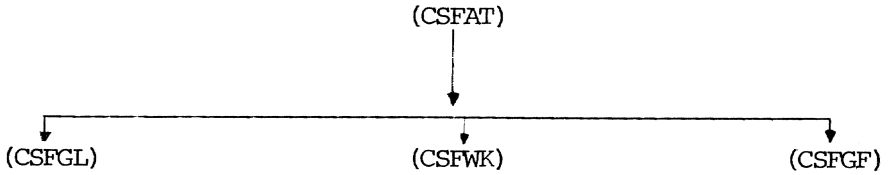
Phase SYSDK1-SYSDK2



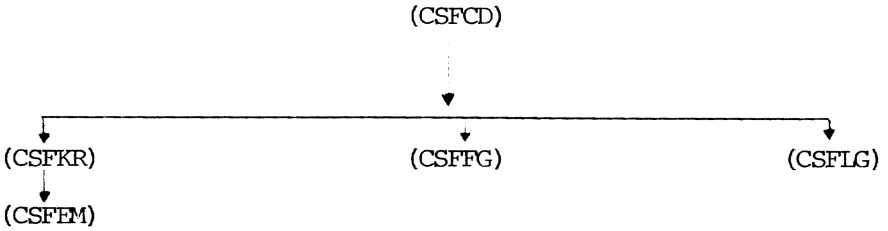
Phase SYSASN



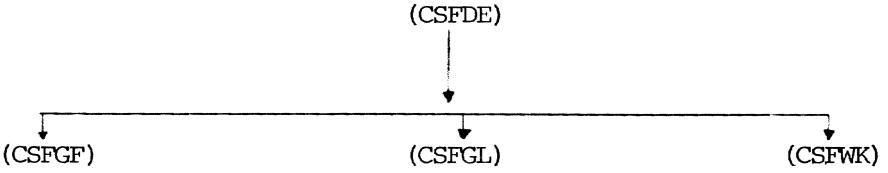
Phase SYSATD



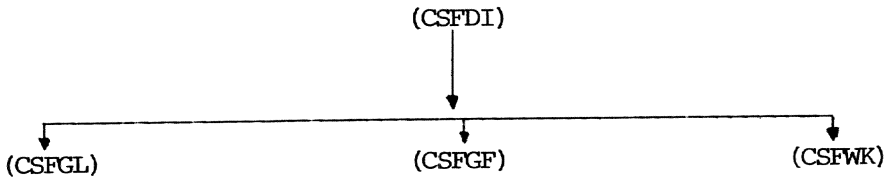
Phase SYSTRT



Phase SYSDEL



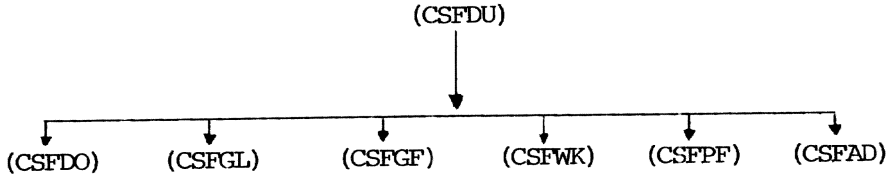
Phase SYSDIS



Phase SYSDSV

(CSFDS)

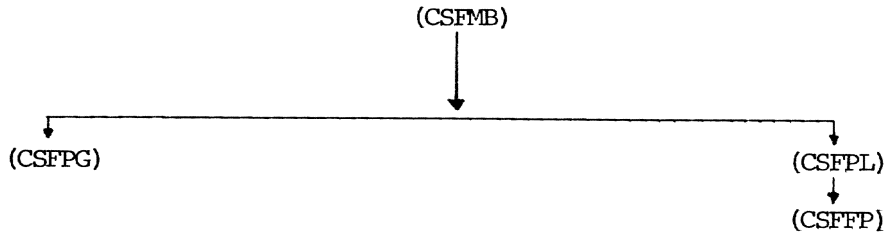
Phase SYSDUM



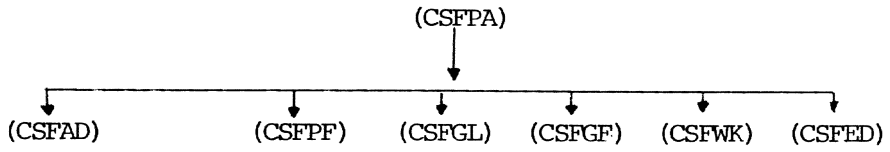
Phase LINK

(CSFLN)

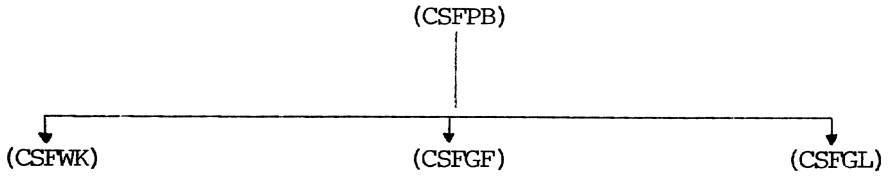
Phase SYSPVM



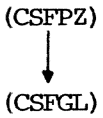
Phase SYSPAT



Phase SYSPUB



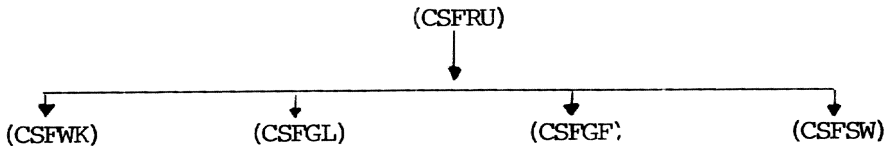
Phase PAUSE



Phase SYSREC

(CSFRE)

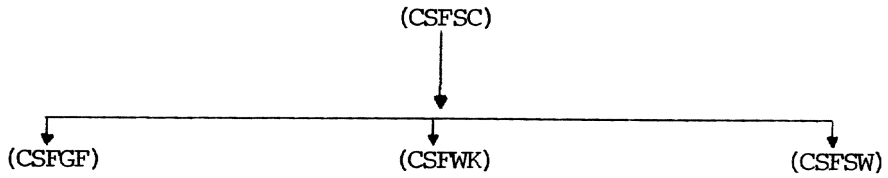
Phase SYSRUN



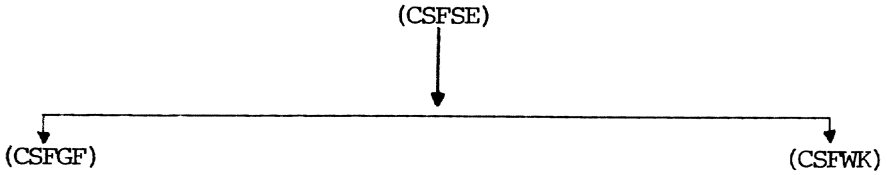
Phase SYSSAD

(CSFSA)

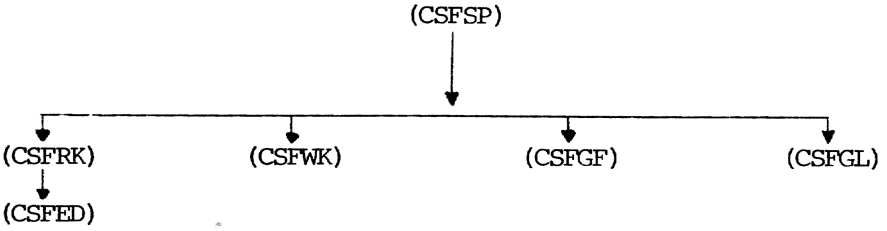
Phase SYSCPU



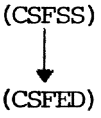
Phase SYSSET



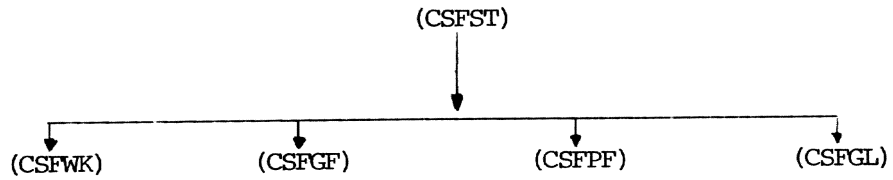
Phase SYSPAU



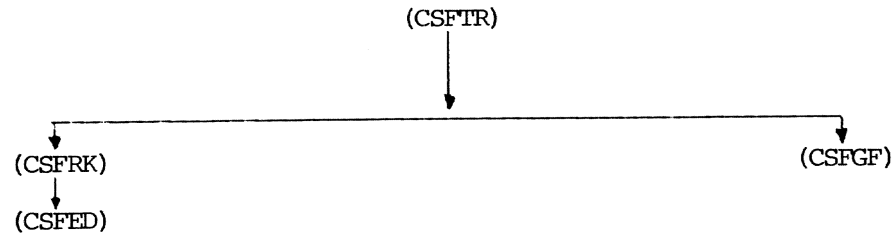
Phase SYSNSS



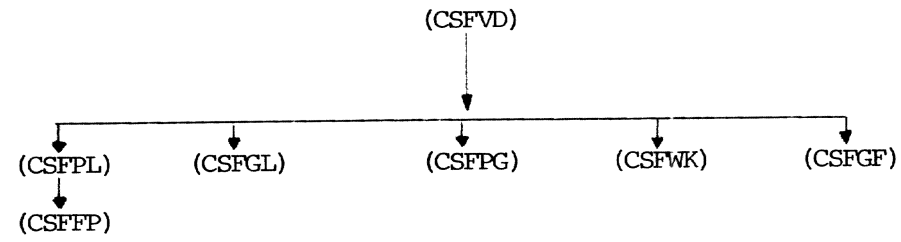
Phase SYSTAT



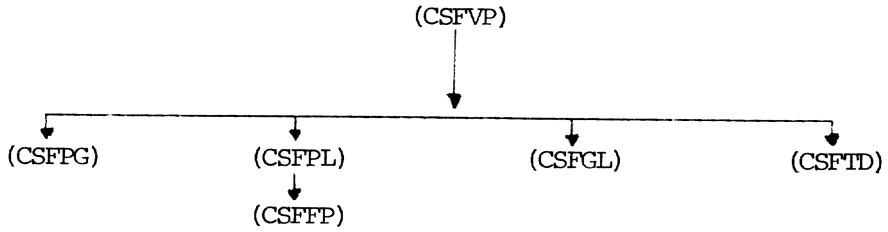
Phase SYSTRA



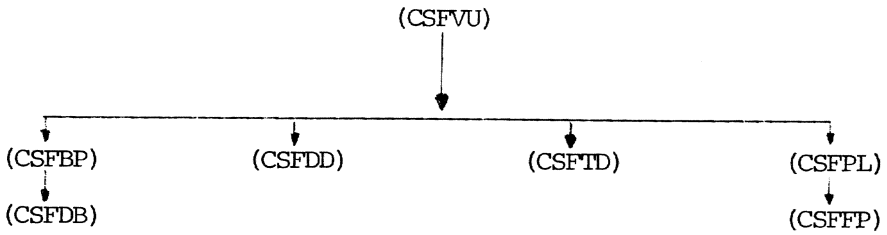
Phase SYSVDS



Phase SYSVPT



Phase SYSVDM



Phase SPVR67

(CSFSU)

Print[illegible] U.S.A. Y28-2040-0

IBM

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)