File Number S360-36
Form Y24-5084-0

Program Logic

IBM System/360
Disk Operating System
Supervisor and Physical and Logical Transients

Program Number 360N-CL-453, Version 3

This publication describes the internal logic of the IBM System/360 Disk Operating System, Supervisor and Transients Programs. It is intended for use by persons involved in program maintenance and by system programmers who are altering the program design. Program logic information is not needed for normal operation of these programs; therefore, distribution of this publication is limited to those with maintenance and alteration requirements. It is designed to be used as a supplement to the program listing.

Effective use of this manual requires an understanding of IBM System/360 operation and of IBM System/360 Disk Operating System control and service programs, macro instructions, and operating procedures. Reference publications for this information are listed in the Preface of this manual.

For overall system control logic description, this PLM is to be used with four other PLMs:

IBM System/360 Disk Operating System, Introduction to System Control Programs, Form Y24-5017.

IBM System/360 Disk Operating System, Librarian Maintenance and Service Programs, Form Y24-5079.

IBM System/360 Disk Operating System, Linkage Editor, Form Y24-5080.

IBM System/360 Disk Operating System, IPL and Job Control Programs, Form Y24-5086.

Restricted Distribution

PREFACE

This Program Logic Manual (PLM) is a detailed guide to the IBM System/360 Disk Operating System supervisor, physical transient, and logical transient programs. It supplements the program listings by providing descriptive text and flowcharts.

Prerequisite and related publications that aid in the use of this manual are:

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

IBM System/360 Disk Operating System, System Control and Service Programs, Form C24-5036.

IBM System/360 Disk Operating System, Supervisor and Input/Output Macros, Form C24-5037.

IBM System/360 Disk Operating System, System Generation and Maintenance, Form C24-5033.

IBM System/360 Disk Operating System, Operating Guide, Form C24-5022.

IBM System/360 Disk Operating System, Data Management Concepts, Form C24-3427.

IBM System/360 Disk and Tape Operating Systems: Assembler Specifications, Form C24-3414.

IBM System/360 Disk Operating System, Basic Telecommunications Access Method PLM, Form Y30-5001.

IBM System/360 Disk Operating System, Queued Telecommunications Access Method PLM, Form Y30-5002.

Titles and abstracts of other related publications are listed in the IBM System/360 Bibliography, Form A22-6822.

This manual consists of seven major sections. The first section is an introduction to the supervisor and transient programs. The next section describes the generation and organization of the supervisor. The next four sections describe the detailed operation of the supervisor, physical IOCS, physical transients, and logical transients. The last section of the manual, the appendixes, contains the label list, error messages, and other references for use in analyzing program details.

The flowcharts for all components are located at the end of the manual. The detailed flowcharts are identified by letters AA through ZZ. Numerals, such as 00 for the program level flowchart, identify the more general flowcharts.

First Edition, February 1968

This edition, Y24-5084-0, obsoletes the Supervisor and transient portions of Y24-5017-2 and its Technical Newsletters, Y24-5059 and Y24-5070.

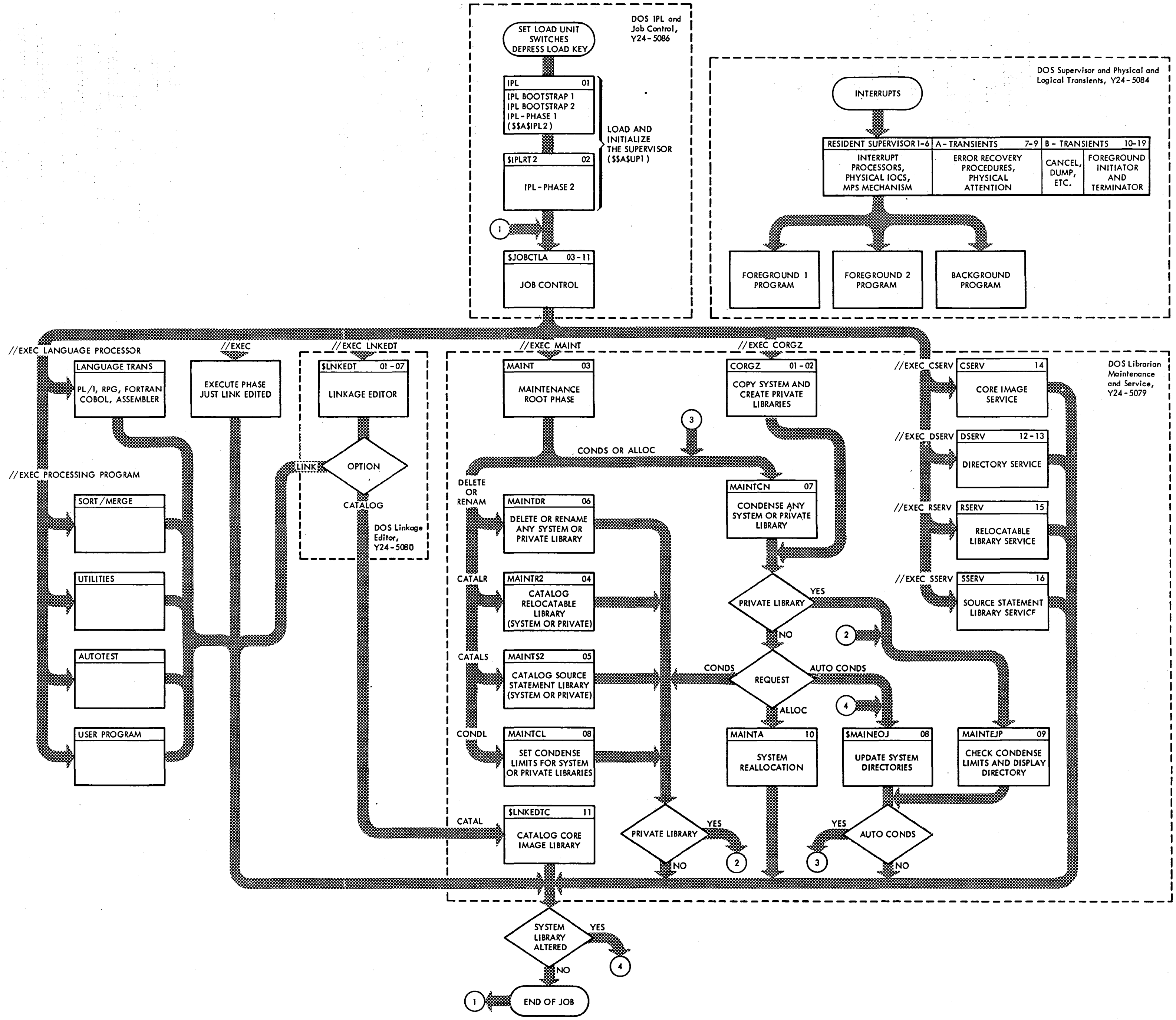
Significant changes or additions to the specifications contained in this publication are continually being made. When using this publication in connection with the operation of IBM equipment, check the latest SRL Newsletter for revisions or contact the local IBM branch office.

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 readers' comments. If the form has been removed, comments may be addressed to: IBM Corporation, Programming Publications, Endicott, New York 13760.

CHARTS

Chart 00. System Program Flow


DOS IPL and Job Control, Y24-5086
SET LOAD UNIT SWITCHES DEPRESS LOAD KEY
IPL 01
IPL BOOTSTRAP 1
IPL BOOTSTRAP 2
IPL-PHASE 1 ($$A$IPL2)
$IPLRT2 02
IPL-PHASE 2
LOAD AND INITIALIZE THE SUPERVISOR ($$A$SUP1)
1
$JOBCTLA 03-11
JOB CONTROL
DOS Supervisor and Physical and Logical Transients, Y24-5084
INTERRUPTS
RESIDENT SUPERVISOR 1-6
INTERRUPT PROCESSORS, PHYSICAL IOCS, MPS MECHANISM
A-TRANSIENTS 7-9
ERROR RECOVERY PROCEDURES, PHYSICAL ATTENTION
B-TRANSIENTS 10-19
CANCEL, DUMP, ETC.
FOREGROUND INITIATOR AND TERMINATOR
FOREGROUND 1 PROGRAM
FOREGROUND 2 PROGRAM
BACKGROUND PROGRAM
//EXEC LANGUAGE PROCESSOR
LANGUAGE TRANS
PL/I, RPG, FORTRAN COBOL, ASSEMBLER
//EXEC
EXECUTE PHASE JUST LINK EDITED
//EXEC LNKEDT
$LNKEDT 01-07
LINKAGE EDITOR
LINK
OPTION
CATALOG
DOS Linkage Editor, Y24-5080
//EXEC PROCESSING PROGRAM
SORT/MERGE
UTILITIES
AUTOTEST
USER PROGRAM
//EXEC MAINT
MAINT 03
MAINTENANCE ROOT PHASE
CONDS OR ALLOC
3
DELETE OR RENAM
MAINTDR 06
DELETE OR RENAME ANY SYSTEM OR PRIVATE LIBRARY
CATALR
MAINTR2 04
CATALOG RELOCATABLE LIBRARY (SYSTEM OR PRIVATE)
CATALS
MAINTS2 05
CATALOG SOURCE STATEMENT LIBRARY (SYSTEM OR PRIVATE)
CONDL
MAINTCL 08
SET CONDENSE LIMITS FOR SYSTEM OR PRIVATE LIBRARIES
CATAL
$LNKEDTC 11
CATALOG CORE IMAGE LIBRARY
PRIVATE LIBRARY
YES
NO
2
//EXEC CORGZ
CORGZ 01-02
COPY SYSTEM AND CREATE PRIVATE LIBRARIES
MAINTCN 07
CONDENSE ANY SYSTEM OR PRIVATE LIBRARY
PRIVATE LIBRARY
YES
NO
2
REQUEST
CONDS
AUTO CONDS
ALLOC
4
MAINTA 10
SYSTEM REALLOCATION
$MAINEOJ 08
UPDATE SYSTEM DIRECTORIES
MAINTEJP 09
CHECK CONDENSE LIMITS AND DISPLAY DIRECTORY
AUTO CONDS
YES
NO
3
DOS Librarian Maintenance and Service, Y24-5079
//EXEC CSERV
CSERV 14
CORE IMAGE SERVICE
//EXEC DSERV
DSERV 12-13
DIRECTORY SERVICE
//EXEC RSERV
RSERV 15
RELOCATABLE LIBRARY SERVICE
//EXEC SSERV
SSERV 16
SOURCE STATEMENT LIBRARY SERVICF
SYSTEM LIBRARY ALTERED
YES
4
NO
END OF JOB
1

The resident version of the IBM System/360 Disk Operating System (DOS), System Control, Version 3, provides disk operating system capabilities for 16K and larger System/360 configurations. At least one IBM 2311 Disk Storage Drive or IBM 2314 Direct Access Storage Facility is required.

Systems larger than 16K can benefit from this 16K package if they do not require the expanded functions of the larger disk operating system packages offered by IBM. The system is disk resident, using the IBM 2311 or IBM 2314 Disk Storage Drive for on-line storage of all programs. Depending on the requirements of the particular application, the system can be expanded to include all processing programs used to perform the various jobs of a particular installation, or it can be tailored to a minimum system to control a single program.

The operating system includes the following components: CPU, input/output channels, input/output control units, input/output devices, microprogramming, system control programs, support programs, user programs, user data files, Teleprocessing capability, and multiple programming capability. This PLM discusses the supervisor and the physical and logical transients that are part of system control. The supervisor and physical IOCS are specifically designed for a user's configuration by means of a one time assembly (generation time). They require reassembly only if the user's configuration changes.

The supervisor and physical IOCS provide the required interface between the program being executed and the other components of the operating system. The program currently being executed is identified to the operating system as the current program. The last program interrupted is identified as the problem program. The problem program or the current program can be, at any given time, either a system control program, a support program, or a user program.

The supervisor program operates with problem programs when job processing (problem program execution) occurs. The supervisor program is divided into two parts:

1. The resident part, called the <u>supervisor nucleus</u>;
2. The nonresident part, called a <u>supervisor transient</u>.

The nucleus is loaded into main storage at IPL time and remains there throughout job processing. A transient (one of many) is loaded, as needed, from the core image library of SYSRES. When a transient has finished performing its service, it can be overlaid by some other transient when some other type of service is required. This technique maximizes the use of main storage allotted to the supervisor. The basic functions performed by the supervisor are:

- Storage protection (required for multiprogramming)
- Interrupt handling
- Channel scheduling
- Device error recovery
- Operator communications
- Program retrieval (fetch or load)
- End-of-job processing
- Timer services (optional).

Each installation must generate its own custom made supervisor by means of a one time assembly. Supervisor generation macros control the generation of the supervisor control program. The user must reassemble the supervisor if its functions are to be modified (for example, when an installation configuration changes).

MULTIPROGRAMMING

For systems with main storage equal to or greater than 24K, disk operating system offers multiprogramming support. This support is referred to as <u>fixed partitioned multiprogramming</u>, because the number and size of the partitions is defined during system generation. The size of the partitions may be redefined by the console operator, after system generation, to meet the needs of a specific program to be executed. Figure 1 illustrates the relationship among programs in a multiprogramming environment.

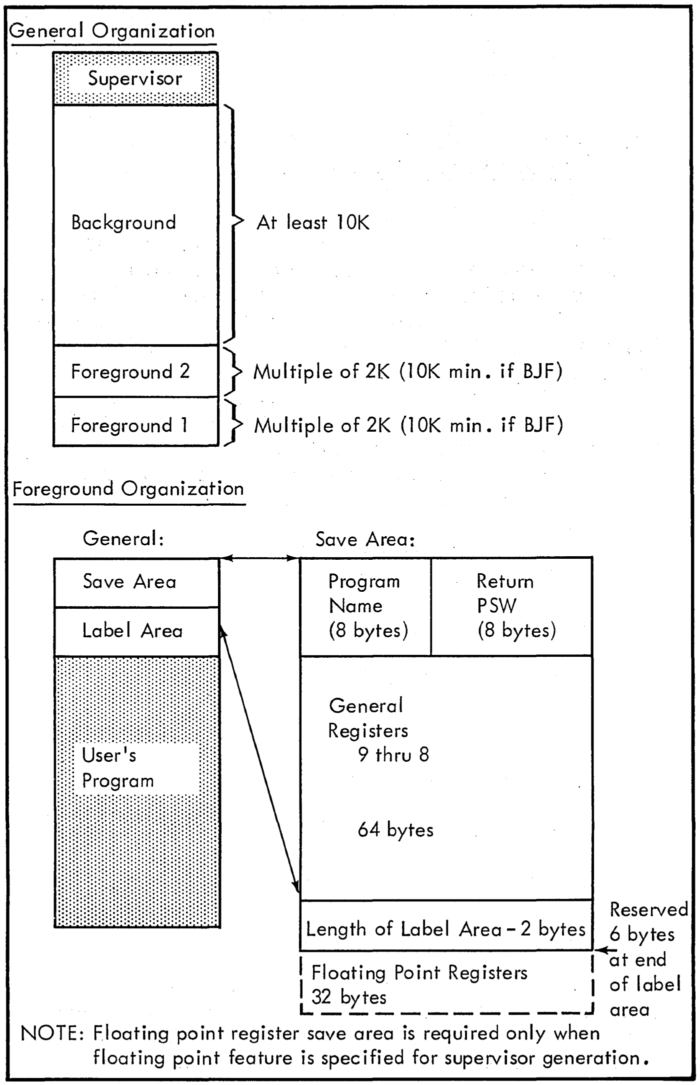

General Organization
Supervisor
Background
At least 10K
Foreground 2
Multiple of 2K (10K min. if BJF)
Foreground 1
Multiple of 2K (10K min. if BJF)
Foreground Organization
General:
Save Area:
Save Area
Label Area
User's Program
Program Name (8 bytes)
Return PSW (8 bytes)
General Registers 9 thru 8
64 bytes
Length of Label Area - 2 bytes
Reserved 6 bytes at end of label area
Floating Point Registers 32 bytes
NOTE: Floating point register save area is required only when floating point feature is specified for supervisor generation.

Figure 1. Multiprogram Main Storage Organization

BACKGROUND VS FOREGROUND PROGRAMS

There are two types of problem programs in multiprogramming, background and foreground. Foreground programs may operate in either the batched-job mode or in the single-program mode. Background programs and batched-job foreground programs are initiated by job control from the batched-job input streams. Single-program foreground programs are initiated by the operator from the printer-keyboard. When one completes, the operator must explicitly initiate the next program. Background and foreground programs initiate and terminate independently of each other.

The system can operate one background program and one or two foreground programs simultaneously. The supervisor controls priority for CPU processing. Foreground programs have priority over background programs. All programs operate with interrupts enabled. When an interrupt occurs, the supervisor gains control, processes the interrupt, and gives control to the highest priority program that is in a ready state.

Control is taken away from a high priority program when that program encounters a condition that prevents continuation of processing until a specified event has occurred. Control is taken away from a lower priority program at the completion of an event for which a higher priority program is waiting. When all programs in the system are simultaneously waiting (i.e., no program can process), the system is enabled for interruptions in the wait state.

The supervisor receives and processes interruptions. When an interruption satisfies a program's wait condition, that program becomes active and competes with other programs for CPU processing time.

In addition to at least 24K positions of main storage, multiprogramming support requires the storage protection feature.

If the batched-job foreground option is selected when the system is generated, all types of programs may be run as foreground programs. (Specifying the option causes the generation of individual communications regions for each partition.) However, the linkage editor and the maintenance functions of the librarian are restricted to the background partition.

This section of the manual describes:

1. Techniques used in supervisor generation.
2. Generation macros and their optional operands.
3. The relationship between the outer generation macros and the inner macros that generate the bulk of the supervisor code.
4. The organization of the supervisor, including the nucleus code, tables, and information blocks.

To understand macro definition language structure and usage, refer to the DOS Supervisor and I/O Macros publication referenced in the preface. With this information, an SSERV listing of the supervisor generation macros, and the PLM material, the reader can identify those sections of code that are generated for his own supervisor program. The basic instruction used in macro definition language is the AIF (ask if) statement. The following examples show how it is used:

1. AIF (&BG20).MP1
   This instruction asks if multiprogramming support is required. If BG20 is on, the next significant line in the SSERV listing is found at the label, .MP1, and any intervening code is rejected by the language translator. If BG20 is not on, the next sequential line on the SSERV listing is significant.

2. AIF (NOT &BG20).NO23
   This instruction tests the opposite status of the BG20 switch. In this case, the line at location .NO23 is the next significant line in the SSERV listing only when BG20 is not on, that is, only when multiprogramming support is not required.

A detailed description of the AIF instruction and the other instructions used in the SSERV listing is given in the Systems Reference Library publication, IBM System/360 Disk and Tape Operating Systems, Assembler Specifications, Form C24-3414.

SUPERVISOR GENERATION

The supervisor is assembled with a series of macros that describe the installation's functional requirements and its configuration. At system generation time, a source deck containing the supervisor generation macros is assembled into an object deck. The job control program places the results of the assembly on SYSLNK (I/O device for the linkage editor program) and calls the linkage editor program. The deck is link-edited and cataloged to the core image library on SYSRES. A corresponding core image library directory entry is posted for the new supervisor, and the Program Information Department (PID) supervisor directory entry is deleted.

Normally, a condense maintenance program would then be executed to remove the PID supervisor from the core image library. The procedures and sequence of events used in system generation are described in the publication, IBM System/360 DOS System Generation and Maintenance, Form C24-5033.

Whenever a new supervisor generation is required by the user the same general steps are taken:

1. Punch the macro instructions, together with the selected optional operands, into a card deck.
2. Execute an assembly, and put the object modules on SYSLNK (using an include control statement with no operand) via the job control program.
3. Link-edit the new supervisor, cataloging it to the core image library, deleting the old supervisor directory entry, and posting the new supervisor directory entry.
4. Re-IPL with the new supervisor.
5. Execute a maintenance program to condense the core image library, deleting the old supervisor program.

SUPERVISOR GENERATION MACROS

The list of supervisor macros and optional operands in Figure 2 is presented to give the reader:

- Supervisor generation macro names.

- Required macro sequence (as listed in Figure 2).

- Macro parameters. (Where there is an assumed value, that value is underlined.)

- A brief description of what the generated macro does.

- A brief description of what the individual parameter options do.

Name	Macro Description	Parameter=Option	Option Description
SUPVR	Describes system environment	SYSTEM=DISK	System residence (SYSRES) must be on a disk device.

		MPS= {NO
		YES
		BJF}
			Indicates multiprogramming support. If YES or BFJ is specified, the system generated is capable of supporting two foreground programs. YES or BJF must be specified if TP=QTAM. If BJF is specified, the system generated will support batched mode for both foreground partitions. Multiple communications regions are generated only if MPS=BJF. MPS=YES is implied if MPS=BJF.

		TP= {NO
		BTAM
		QTAM}
			Specify if Basic or Queued Teleprocessing Access Method (BTAM or QTAM) is desired. When QTAM is specified, SVC support for BTAM is also included.

		MICR= {NO
		1412
		1419
		1419D}
			Indicates whether the supervisor is to support magnetic ink character readers. If both 1412's and 1419's are present indicate 1419. 1419D indicates Dual Address Adapter 1419's. If 1412/1419's are attached to the multiplexor channel the PIOCS parameter BMPX=YES is not supported.

CONFG	Describes hardware features	MODEL= {30
		nn}
			nn defines the System/360 model number (30, 40, etc.).

		SP= {NO
		YES}
			Indicates the storage protection feature is desired. YES is assumed if MPS=YES or BJF in the SUPVR macro.

		DEC= {NO
		YES}
			Decimal feature.

		FP= {NO
		YES}
			Floating-point feature.

		TIMER= {NO
		YES}
			Timer feature. If TIMER=YES the supervisor macro GETIME is supported.

STDJC	Sets standard values for job control variables.	DECK= {YES
		NO}
			Output modules on SYSPCH.

		LIST= {YES
		NO}
			Source modules listings from language translators on SYSLST.

		XREF= {YES
		NO}
			Language translators output symbolic cross-reference lists on SYSLST.

		ERRS= {YES
		NO}
			Compilers summarize all errors in source programs on SYSLST.

		LOG= {YES
		NO}
			Listing of all control statements on SYSLST.

		DUMP= {YES
		NO}
			Dump of registers and main storage on SYSLST.

		LINES= {56
		nn}
			Number of lines per page on SYSLST.

		DATE= {MDY
		DMY}
			Format of date.

		CHARSET= {48C
		60C}
			Specifies the 48 or 60 character set for language translator input on SYSIPT.

		LISTX= {NO
		YES}
			Hexadecimal object module listings from compilers on SYSLST.

		SYM= {NO
		YES}
			Assembler output symbol tables on SYSPCH.

Figure 2. Supervisor Macros (Part 1 of 3)

Name	Macro Description	Parameter=Option	Option Description

FOPT	Describes functional supervisory options	OC= {NO
		YES}
			Operator initiated communications to problem programs. If OC=YES the facility is available to all programs in MPS.

		PC= {NO
		YES}
			Problem program routine for program check. If PC=YES, the facility is available to all programs in MPS.

		IT= {NO
		BG
		F1
		F2}
			Problem program ability to set timer intervals and specify a timer interrupt routine. BG, F1, or F2 indicates which program has the facility. When IT is specified for a partition, TIMER=YES is assumed in the CONFG macro. Timer support is available to only one program in MPS.

		TEB= {NO
		n}
			Tape error statistics are to be accumulated and logged where n is the number of tape drives attached to the system.

		SKSEP= {NO
		YES
		n}
			Specifies if SEEKs are to be separated from the remainder of channel programs. Seek separation allows other devices on the channel to be accessed (including other seeks) during the seek. YES indicates support for all DASD type devices specified by the DVCGEN macro at system generation time. n is the number of DASD devices to be supported and cannot be less than the number of DASD devices specified at system generation. The maximum number is 255.

		CE= {NO
		YES
		n}
			Specifies the amount of core to be allocated to customer engineer serviceability routines. When YES is specified, 600 bytes are allocated; when n bytes are specified, it must be for a minimum of 600 bytes.

		CCHAIN= {NO
		YES}
			Command chaining support for retry on I/O operations. When an error occurs and CCHAIN=YES, the user is allowed to retry at the last CCW executed instead of at the first CCW in the channel program as is the case in a normal retry. This option requires that the appropriate bit be set in the CCB.

		DASDFP= {NO (n,n, {2311 2314 2321})}	Supervisory DASD file protection, where (n,n) indicates the range of channels to which DASD's may be attached. Specifying 2311 or 2314 indicates support for both. 2321 option indicates support for 2321 device as well as for 2311 and 2314. DASDFP prevents the user from writing outside the extents of his file in case of program error. Extents are protected to the nearest cylinder.
		*SYSFIL= {NO (2311[,n1,n2] 2314)}	System input and system output (SYSRDR, SYSIPT, SYSLST, SYSPCH) files may be assigned to a 2311, 2314, or both. Specifying either indicates support for both. If MPS=BJF in the SUPVR macro, support is given for foreground logical units when running in batched mode. n1 = residual capacity (in records) for beginning of operator notification when SYSLST is assigned to a 2311 or 2314. 100 ≤ n1 ≤ 65536 If n1 is omitted, 1000 is assumed n2 = residual capacity (in records) for beginning of operator notification when SYSPCH is assigned to a 2311 or 2314. 100 ≤ n2 ≤ 65536 If n2 is omitted, 1000 is assumed.
		* Valid when at least 24K bytes of main storage are available.	

PIOCS	Describes the system I/O configuration	SELCH= {YES
		NO}
			Selector channels attached to the system.

		BMPX= {NO
		YES}
			Burst mode devices will be supported on multiplexor channel. BMPX=YES not supported with MICR.

		CHANSW= {NO RWTAU TSWTCH}	Channel switching tape control unit. RWTAU=2404 or 2804, TSWTCH=2816. If either 2404 or 2804 and 2816, RWTAU must be specified.
		TAPE= NO 9 7	Indicates required tape PIOCS support. 9=nine track only, 7=seven or nine track, NO=No tape drives attached. NO is the assumed value.

Figure 2. Supervisor Macros (Part 2 of 3)

Name	Macro Description	Parameter=Option	Option Description
ALLOC	Partitions storage for MPS (Optional macro).	{F1=nK, F2=nK}	Specifies storage partitioning MPS, where n must be a multiple of 2. Must be at least 10K if MPS=BJF.
IOTAB	Describes installation requirements for I/O tables.	IODEV=$\left\{\begin{matrix}\underline{10}\\ n\end{matrix}\right\}$	Number of I/O devices attached to the system
		BGPGR=$\left\{\begin{matrix}\underline{10}\\ n\end{matrix}\right\}$	Number of symbolic units of the class SYSnnn for the background program.
		F1PGR=$\left\{\begin{matrix}\underline{5}\\ n\end{matrix}\right\}$	Number of symbolic units of the class SYSnnn for F1. Valid only in MPS. Otherwise zero is assumed.
		F2PGR=$\left\{\begin{matrix}\underline{5}\\ n\end{matrix}\right\}$	Number of symbolic units of the class SYSnnn for F2. Valid only in MPS. Otherwise zero is assumed.
		CHANQ=$\left\{\begin{matrix}\underline{6}\\ n\end{matrix}\right\}$	Number of I/O requests in the channel queue. 6 is the minimum value generated.
		JIB=$\left\{\begin{matrix}\underline{5}\\ n\end{matrix}\right\}$	Number of Job Information Blocks (JIBs) for the system. Requirements are: 1. One JIB for each temporary logical unit assignment. 2. One JIB for each alternate logical unit assignment. 3. One JIB for each open 2311 or 2314 extent with the DASD file protect feature. 4. Two JIBs for each open 2321 extent with the DASD file protect feature.
DVCGEN	Specifies I/O devices. Each device typе requires a separate DVCGEN macro. (See note 1 for DVCGEN rules. This is an optional macro.)	CHUN= {X'cuu'}	Specify the hexadecimal number of the channel and unit for the device.
		DVCTYP= {xxxxxx}	Specify the device type. See Figure 21.
		CHANSW= $\left\{\begin{matrix}\underline{NO}\\ YES\end{matrix}\right\}$	Specify if the device is attached to more than one selector channel. If it is, the device can be switched.
		MODE= {X'ss'}	1. 2400T9. MODE specifies the tape mode. X'C0' is the default value. 2. 2400T7. MODE specifies the tape mode. X'90' is the default value. See Figure 22 for other values. 3. 2702. MODE designates the SADxxx command. X'00' is the default value. X'00' SAD0 X'01' SAD1 X'02' SAD2 X'03' SAD3 4. 2260 (Local). MODE is used to specify the 1053 printer when CHUN=X'cuu' refers to a 1053 attached to a 2848. The operand must be entered as MODE=X'01'. 5. 1412/1419. MODE designates the external interrupt bit associated with Magnetic Ink Character Readers. The modes X'01' through X'20' correspond to external interrupt PSW bits 31 through 26 respectively. For the dual address adapter 1419, this parameter is needed for both the 1419P and 1419S device types. X'01' Device attached to external line 7. X'02' Device attached to external line 6. X'04' Device attached to external line 5. X'08' Device attached to external line 4. X'10' Device attached to external line 3. X'20' Device attached to external line 2.
ASSGN	Sets standard background I/O assignments. A separate macro is required for each standard assignment desired. (Optional macro) CAUTION: The ASSGN macro allows SYSRDR, SYSLST, SYSPCH, and SYSIPT to be assigned to a tape or DASD. However, IPL unassigns any such assignments.	{SYSxxx,X'cuu'}	SYSxxx is any background symbolic logical unit (SYSIPT, SYSLOG, etc.) or programmer logical unit (SYS000, SYS001, etc.). X'cuu' is the hexadecimal number of the channel and unit to which the symbolic device is attached.
SEND	Indicates end of supervisor generation.	{n}	Specifies the beginning address of the problem program area. An area should be reserved for supervisor expansion and maintenance. The parameter is optional. If not specified, no area is reserved beyond the assembled last address of the supervisor.

Note 1: Rules for Using DVCGEN

1. A separate DVCGEN macro instruction is required for each device.
2. The total number of DVCGEN macros must not exceed the total number of devices specified in the IODEV parameter of the IOTAB macro.
3. DVCGEN macros must be specified in ascending channel address sequence.
4. Switchable units (attached to more than one selector channel) must be defined once. They are defined on the lowest channel on which they are addressable.
5. The sequence of the DVCGEN cards determines the priority of the devices on their channel. Switchable units must be the last devices for each channel, and must be on consecutive channels.
6. The specifications of these macros may be altered by IPL ADD and DEL statements. See IPL PLM, Y24-5086.

Figure 2. Supervisor Macros (Part 3 of 3)

MACRO RELATIONSHIPS

The code generated by the assembler for any selected supervisor generation is a function of the generation macros described in Figure 2 and of a group of inner macros called by the generation macros. The primary purpose of the generation macros is to set global values, based on parameter options, that can be tested by the inner macros. These macros then generate the bulk of the supervisor code. The specific instructions assembled depend on the global settings. Some of the generation macros also generate code; however, these can be treated as exceptions and are identified in this subsection.

The most important global values used in supervisor generation are the B-globals. Therefore, this subsection emphasizes the generation macros that establish B-global values. However, some A-globals that are tested in the same manner as B-globals are also described in this subsection. A-globals that provide arithmetic values and all C-globals are not described. Two figures in this subsection show macro relationships. Figure 3 shows the code generated, if any, and the globals set, if any. Figure 4 indicates the on-off conditions of the globals.

ORGANIZATION

The physical organization of the supervisor depends on the sequence of the supervisor generation macros. The sequence is predetermined and can not be changed by the user. The logical organization depends on the parameter options selected at generation time. Figure 5 describes the assembled supervisor by a main storage map, which illustrates the supervisor physical organization in four major areas:

- Low Core
- Nucleus Code
- I/O Tables and Information Blocks
- Logical and Physical Transient Areas

The logical organization is not described in this manual because of the variety of options available. The reader must determine the logical organization for his individual supervisor generation. By using the program level flowcharts to point to the detailed flowcharts, the reader selects the correct group of flowcharts for the desired generation.

LOW CORE

The main storage locations that make up low core can be classified as PSWs, CSWs, CAWs, and main storage areas. PSWs, CAWs, and CSWs are described in Figures 29, 30 and 31, respectively. The main storage areas include:

Byte (hex)	Function
0-4	Message area when SYSLOG is disabled.
0-1	Contains the error code for the SEREP diagnostic program.
14	Contains the address of the background communications region located within the nucleus code.
50	Contains the system timer used with microprogramming.
54	Contains system time of day set by job control and IPL, updated by the supervisor timer routine (optional).
80	Beginning of the diagnostic scan-out area.

Macro	Type	Code Generated	Critical Globals Set
SUPVR	generation	Defines low main storage	BG20 BG21 BG24 BG35 BG36
CONFG	generation	None	BG1 BG2 BG22 BG23
STDJC	generation	None	BG34
FOPT	generation	General cancel General exit General entry Communications region	AG0 BG6 BG7 BG8 BG30 BG32 BG33 AG7 AG10 AG21 AG22 AG23
PIOCS	generation	None directly calls inner macros	BG3 BG4 BG9 BG10 BG11 BG12 BG31
SGTCHS	inner	Channel scheduler Start I/O I/O interrupt	none
SGUNCK	inner	Unit check Error recovery exits	none
SGDFCH	inner	Fetch subroutine	none
SGSVC	inner	Supervisor interrupts Program check interrupts External interrupts	none
SGDSK	inner	Disk error recovery	none
SMICR	inner	External interrupts for MICR devices. Program checks in stacker select routine. Error recovery for test I/O and start I/O.	none
SGTCON	inner	SVEREG subroutine, VLDADR1 subroutine, ATNRTN routine, CCW chain, disk information blocks, error recovery block, SVC interrupt table, PC option table, IT option table, and OC option table, logical transient save area.	none
ALLOC	generation	None	none
IOTAB	generation	Supervisor table expansions - PIBs, channel queue table (CHANQ), and PUB, JIB, TEB, and LUB tables.	none
DVCGEN	generation	Overlays for PUB table entries	AG8
ASSGN	generation	Overlays for LUB table entries	none
SEND	generation	Generates communications regions extension(s), SABs, and CE table. Defines end of supervisor nucleus, beginning of A and B transient areas, start of problem program area, CE area, BG save area	none
COMMN	inner	Communications regions for all partitions	none

Figure 3. Macro Functions

Global	Purpose	On Setting
BG0	Determines if extension(s) to the communications region(s) will be generated.	CE = YES or n
BG1	Determines whether the storage protect feature is used.	SP = YES
BG2	Determines whether the timer feature is used.	TIMER = YES
BG3	Determines whether channel switching is supported (2816).	CHANSW = TSWTCH
BG4	Determines if tape error statistics are to be accumulated and logged.	TEB = n
BG5	Reserved	
BG6	Determines if the asynchronous user interrupt key routine is supported.	OC = YES
BG7	Determines whether the internal timer option is supported.	IT = F1, or F2, or BG
BG8	Determines if the user program check routine is supported.	PC = YES
BG9	Determines whether channel switching is supported (2404, 2804).	CHANSW = RWTAU
BG10	Indicates whether selector channels are supported.	SELCH = YES
BG11	Indicates whether burst mode devices will be supported on the multiplexor channel.	BMPX = YES
BG12	Determines the type of tape support required.	TAPE = 7 or 9
BG20	Determines whether multiprogramming support is required.	MPS = YES
BG21	Determines whether Teleprocessing support is required.	TP = BTAM or QTAM
BG22	Determines if the decimal feature is used.	DEC = YES
BG23	Determines if the floating point feature is used.	FP = YES
BG24	Indicates that batched jobs will be run in foreground partitions.	MPS = BJF
BG30	Determines if command chaining support for retry on I/O operations is used.	CCHAIN = YES
BG31	Determines if 9 track tape support is required.	TAPE = 9
BG32	Determines whether the DASD file protect feature is supported.	DASDFP = n,n
BG33	Determines if logical system I/O units are a disk device.	SYSFIL = 2311 or 2314
BG34	Determines the type of date configuration to be supported.	DATE = MDY
BG35	Determines if any MICR Device is supported.	MICR = 1412, 1419, 1419D
BG36	Determines if 1419D (MICR device with dual address adapter) only is supported.	MICR = 1419D
AG7	Determines that the seek separation option is desired.	SKSEP = YES or n
AG8	Sets count of direct access storage devices generated in DVCGEN.	SKSEP = YES or n
AG10	Indicates that an area is reserved for CE routines.	CE = YES or n
AG21	Determines if a timer interrupt routine is for a BG program.	IT = BG
AG22	Determines if a timer interrupt routine is for a F2 program.	IT = F2
AG23	Determines if a timer interrupt routine is for a F1 program.	IT = F1

Figure 4. Global Settings

NUCLEUS CODE

The main storage map (Figure 5) illustrates the major routine and subroutine organization of the supervisor. Specific instructions are included or omitted depending on generation options. This manual describes the disk error recovery as the resident error recovery routine. The background communications region is one part of the nucleus coding that does not change from generation to generation. Figure 6 illustrates the structure of the communications region. The starting address of the communications region is made available to a user in general register 1 through the COMRG macro. For certain options, extensions to the communications regions are generated at the end of the I/O Tables by the SEND macro.

I/O TABLES

The I/O tables that comprise this section of the supervisor establish the interface between a user's program and the hardware channels. Collectively, these tables are called the system control center (See Figure 7). For every device used on the system, there must be a PUB (Physical Unit Block). For every logical unit name (SYSXXX) used, there must be a LUB (Logical Unit Block). When an I/O request is made, an entry is made in CHANQ (the channel queue). The entry contains a CCB (Channel Command Block) address which, in turn, points to a CCB that contains a code (LUB table index) for the logical unit name.

The supervisor processes the request when possible on the device assigned to the logical unit. If the TEB=YES option was selected at supervisor generation time, counts of tape errors are kept in TEBs (Tape Error Blocks).

To understand the interaction between the various I/O tables, the reader should know the classification and sequence of the symbolic unit references (SYSXXX). The systems class (symbolic unit names reserved for system use) is made up of:

1. SYSRDR
2. SYSIPT
3. SYSPCH
4. SYSLST
5. SYSLOG
6. SYSLNK
7. SYSRES
8. SYSSLB
9. SYSRLB
10. SYSUSE
11. SYSFGI (This entry is reserved in the LUB table, but this LUB is not used in DOS Version 3.)

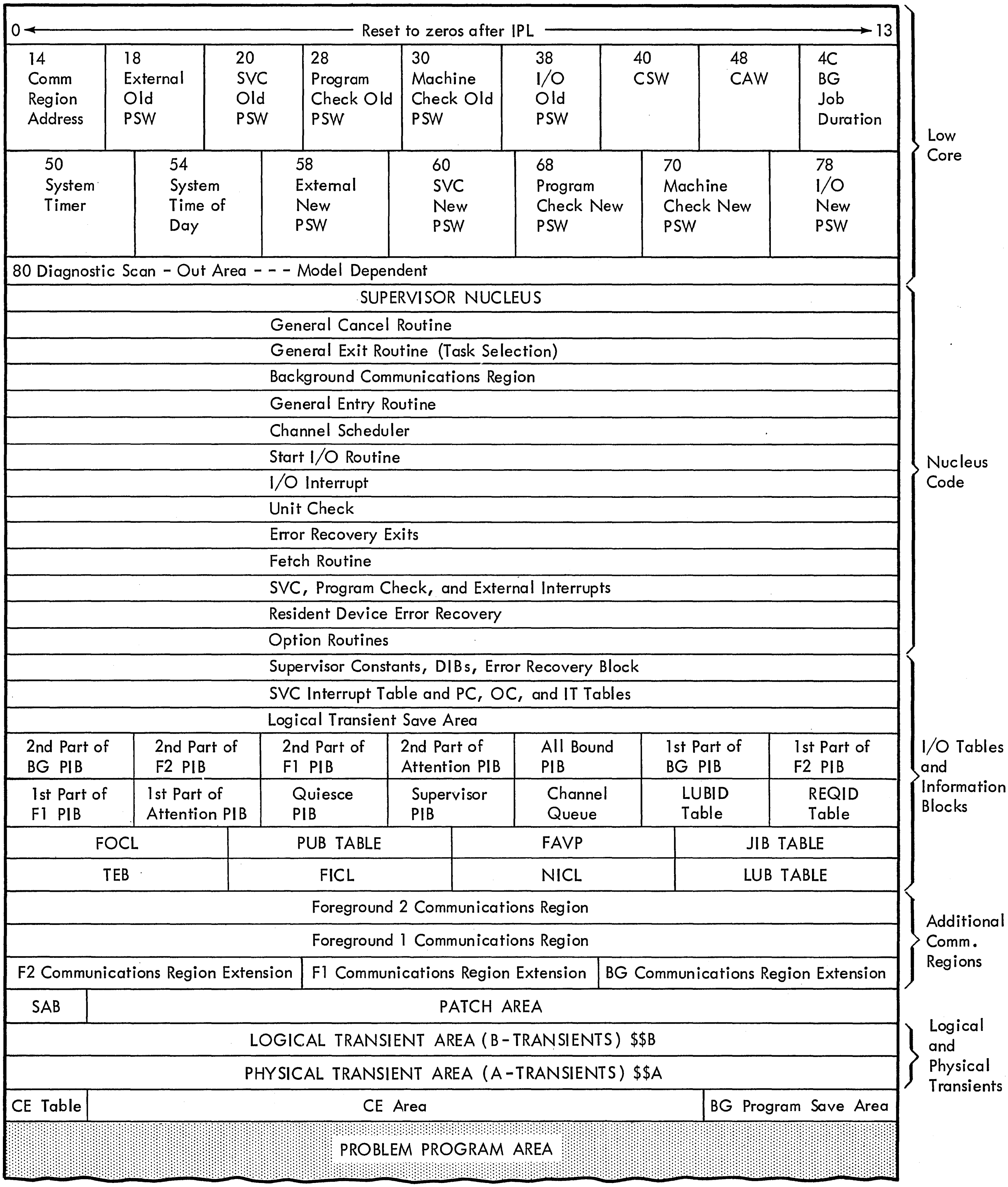
Foreground programs can use all the system unit names except SYSLNK, SYSSLB, and SYSRLB. The programmer class (symbolic unit names reserved for programmer use) is made up of SYS000 to SYS221. This class is subdivided into these classifications:

1. Background logical unit class (minimum of 10).

2. Foreground two logical unit class (minimum of 5).

3. Foreground one logical unit class (minimum of 5).

PUBs are built at system generation or IPL time. LUBs are built at system generation time. PUBs are assigned to LUBs at system generation or by the job control program, or by the single program initiator. CHANQ and TEB entries are built and processed by the supervisor program. Figure 8 illustrates the I/O table interrelationships. Figures 9 through 19 are illustrations of individual I/O tables.



0 ← Reset to zeros after IPL → 13

14 Comm Region Address	18 External Old PSW	20 SVC Old PSW	28 Program Check Old PSW	30 Machine Check Old PSW	38 I/O Old PSW	40 CSW	48 CAW	4C BG Job Duration

50 System Timer	54 System Time of Day	58 External New PSW	60 SVC New PSW	68 Program Check New PSW	70 Machine Check New PSW	78 I/O New PSW

80 Diagnostic Scan - Out Area - - - Model Dependent

Low Core

SUPERVISOR NUCLEUS

General Cancel Routine

General Exit Routine (Task Selection)

Background Communications Region

General Entry Routine

Channel Scheduler

Start I/O Routine

I/O Interrupt

Unit Check

Error Recovery Exits

Fetch Routine

SVC, Program Check, and External Interrupts

Resident Device Error Recovery

Option Routines

Nucleus Code

Supervisor Constants, DIBs, Error Recovery Block

SVC Interrupt Table and PC, OC, and IT Tables

Logical Transient Save Area

2nd Part of BG PIB	2nd Part of F2 PIB	2nd Part of F1 PIB	2nd Part of Attention PIB	All Bound PIB	1st Part of BG PIB	1st Part of F2 PIB
1st Part of F1 PIB	1st Part of Attention PIB	Quiesce PIB	Supervisor PIB	Channel Queue	LUBID Table	REQID Table

FOCL	PUB TABLE	FAVP	JIB TABLE
TEB	FICL	NICL	LUB TABLE

I/O Tables and Information Blocks

Foreground 2 Communications Region

Foreground 1 Communications Region

F2 Communications Region Extension | F1 Communications Region Extension | BG Communications Region Extension

Additional Comm. Regions

SAB | PATCH AREA

LOGICAL TRANSIENT AREA (B-TRANSIENTS) $$B

PHYSICAL TRANSIENT AREA (A-TRANSIENTS) $$A

Logical and Physical Transients

CE Table | CE Area | BG Program Save Area

PROBLEM PROGRAM AREA

Note: For PSW format see Figure 29. For CSW format see Figure 31. For CAW format see Figure 30.

Figure 5. Supervisor Storage Allocation

COMREG *

Displacement hexadecimal	0	8	0A	0C	17	18	20	24
Displacement decimal	0	8	10	12	23	24	32	36
	Date	Address of PPBEG	Address of EOSSP	Problem Program Use	UPSI Byte	Job Name	Highest Storage Address	End Address of Last Phase Fetched or Loaded
	XXXXXXXX	XX	XX	XXXXXXXXXXX	X	XXXXXXXX	XXXX	XXXX

Displacement hexadecimal	28	2C	2E	30	34	35	36	37	38	39	3A	3B
Displacement decimal	40	44	46	48	52	53	54	55	56	57	58	59
	End Address of Longest Phase Fetched or Loaded	Label Area Length	PIK	End of Storage Address	Machine Confg. Byte	System Confg. Byte	Standard Options		Job Control Byte	Linkage Control Byte	Language Translator Control Byte	Job Duration Indicator Byte
	XXXX	XX	XX	XXXX	X	X	XX		X	X	X	X

Job Control Switches

Displacement hexadecimal	3C	3E	40	42	44	46	48	4A	4C	4E	4F
Displacement decimal	60	62	64	66	68	70	72	74	76	78	79
	Disk Address of Label Cylinder	Address of FOCL	Address of PUB	Address of FAVP	Address of JIB	Address of TEB	Address of FICL	Address of NICL	Address of LUB	Line Count for SYSLST	System Date
	XX	XX	XX	XX	XX	XX	XX	XX	XX	X	XXXXXXXXX

Displacement hexadecimal	58	5A	5C	5E	60	62	64
Displacement decimal	88	90	92	94	96	98	100
	LIOCS Comm. Bytes	Address of 1st Part of PIB Table	ID Number of Last Checkpoint	Length of LUB ID Queue = No. of Channel Queue Entries	Address of Disk Information Block (DIB)	Address of Channel Scheduler Error Block	Address of PC Option Table
	XX	XX	XX	XX	XX	XX	XX

Displacement hexadecimal	66	68	6A	6C	6E	70	7C	7E
Displacement decimal	102	104	106	108	110	112	124	126
	Address of IT Option Table	Address of OC Option Table	Key of Program with Timer Support	Address of the LUBID Queue	Logical Transient Key	Supervisor Constants	Address of 2nd Part of PIB Table	Address of MICR DTF Table (PDTABB)
	XX	XX	XX	XX	XX	XXXXXXXXXXXX	XX	XX

Displacement hexadecimal	80	84	86	88		
Displacement decimal	128	132	134	136		
	Address of QTAM Vector Table	Address of BG Comm. Region	Reserved	Pointer to Comm. Region Extension	Communications Region Extension (see Figure 5 for location)	Address of CE Table
	XXXX	XX	XX	XXXX		XXXX

* The address of the communications region is in fixed location X'14'-X'17'.

Displacement values illustrated can be used to access the listing and/or the key that follows the figure. The key offers more detailed information about each area when necessary.

Figure 6. Supervisor Communications Region (Part 1 of 5)

Key to Communications Region Displacements:

0 MM/DD/YY or DD/MM/YY obtained from the job control date statement. Format controlled by COMREG +53 (date convention byte) bit 0.

8 Address of the problem program area.

10 Address of the beginning of the problem program area. Y (EOSSP) = Y (PPBEG) if the storage protection option has not been selected. Y (EOSSP) equals the first main storage location with a storage protection key of 1, if storage protection is supported.

12 User area. If seek separation option is specified, bytes 12 and 13 are used at IPL time for the address of the seek address block.

23 User program switch indicator.

24 Job name set by the job control program from information found in the job statement.

32 Address of the uppermost byte of the problem program area as determined by the IPL program. (Clear storage routine determines the address, ENDRD routine of $$A$IPL2 stores it.)

36 Address of the uppermost byte of the last phase of the problem program fetched or loaded. The initial value (as shown) is overlaid by the first fetch or load to the problem program area.

40 Address of the uppermost byte of the longest phase of the problem program fetched or loaded. The initial value is overlaid by the first fetch or load to the problem program area.

44 Length of the problem program label area.

46 Program Interrupt Key: Value is equal to the displacement from the start of the PIB table to the PIB for the task.
First Byte - always zero.
Second Byte - Contains the key of the program that was last enabled for interrupts. (When an interrupt occurs, the PIK indicates to the supervisor which program was interrupted.)

Task	PIK Value
*All Bound	X'00'
BG	X'10'
* F2	X'20'
* F1	X'30'
Attn Rtn	X'40'
Quiesce I/O	X'50'
Supervisor	X'60'

* These tasks do not exist in a non-MPS system.

48 Logical end of main storage address.

52 Machine Configuration Byte (Values set at supervisor generation time.)

Bit 0: 1 = Storage protect
0 = No storage protect
1: 1 = Decimal feature
0 = No decimal feature
2: 1 = Floating-point feature
0 = No floating-point feature
3: Reserved
4: 1 = Timer feature
0 = No timer feature
5: 1 = Channel switching device
0 = No channel switching device
6: 1 = Burst mode on multiplex channel support
0 = No burst mode on multiplex support
7: Reserved

Figure 6. Supervisor Communications Region (Part 2 of 5)

Key to Communications Region Displacements:

53 System Configuration Byte

Bit 0: 1 = DDMMYYJJ
0 = MMDDYYJJ (Set at generation time by STDJC)

- 1: 1 = Multiprogramming environment
  0 = Batch job environment
- 2: 1 = DASD file-protect supported
  0 = No file-protect support for DASD
- 3: 1 = DASD SYSIN-SYSOUT
  0 = No DASD SYSIN-SYSOUT
- 4: 1 = Teleprocessing
  0 = No Teleprocessing
- 5: 1 = Batch job in foreground
  0 = No BJF
- 6-7: Reserved

54 This byte contains the standard language translator I/O options (set by the STDJC macro).

Bit 0:	DECK option	1 = yes, output object modules on SYSPCH
1:	LIST option	1 = yes, output source module listings and diagnostics on SYSLST
2:	LISTX option	1 = yes, output hexadecimal object module listings on SYSLST (compilers only)
3:	SYM option	1 = yes, output symbol tables on SYSLST/SYSPCH
4:	XREF option	1 = yes, output symbolic cross reference list on SYSLST
5:	ERRS option	1 = yes, output diagnostics on SYSLST (compilers only)
6:	CHARSET option	1 = 48, input on SYSIPT is 48 or 60 character set
7:	Reserved	

55 This byte contains the standard supervisor options for abnormal EOJ and control statement display.

Bit 0:	Always on	
1:	DUMP option	1 = yes, dump registers and storage on SYSLST
2:	Reserved	
3:	LOG option	1 = yes, list all control statements on SYSLST
4-7:	Reserved	

56 Job control byte

- Bit 0: Reserved
- 1: 1 = Return to caller on LIOCS disk open failure
  0 = Do not return to caller on LIOCS disk open failure
- 2: 1 = Job control input from SYSRDR
  0 = Job control input from SYSLOG
- 3: 1 = Job control output on SYSLOG
  0 = Job control output not on SYSLOG
- 4: 1 = Cancel job
  0 = Do not cancel job
- 5: 1 = Pause at end-of-job step
  0 = No pause at end-of-job step
- 6: 1 = SYSLOG is not a 1052
  0 = SYSLOG is a 1052
- 7: 1 = SYSLOG is assigned to the same device as SYSLST
  0 = SYSLOG is not assigned to the same device as SYSLST

Figure 6. Supervisor Communications Region (Part 3 of 5)

Key to Communications Region Displacements:

57 Linkage control byte

Bit 0: 1 = SYSLNK open for output
0 = SYSLNK not open for output
1: 1 = $ or FG program phase deleted, renamed, or cataloged (flag bit for $MAINEOJ)
2: 1 = Allow EXEC
0 = Suppress EXEC
3: 1 = Catalog linkage editor output
0 = Do not catalog linkage editor output
4: 1 = Supervisor has been updated
0 = Supervisor has not been updated
5: 1 = Executing in AUTOTEST mode
0 = Not executing in AUTOTEST mode
6: 1 = Reallocate or condense in progress
7: 1 = Fetch $MAINEOJ at end of job to update system directory
0 = Do not fetch $MAINEOJ at end of job for update

58 Language processor control byte. This is a set of switches used to specify nonstandard language translator options. The switches within the byte are controlled by job control OPTION statements and when set to 1, override standard options. The format of this byte is identical to the standard option byte (displacement 54) with one exception: Bit 7 in this byte is used to indicate to LIOCS that the rewind and unload option has been specified.

59 Job duration indicator byte

Bit 0: 1 = Within a job condition
0 = Outside a job condition
1: 1 = Dump on an abnormal end-of-job condition
0 = No dump on abnormal EOJ
2: 1 = Pause at EOJ }
0 = No pause at EOJ } Set by Attention Routine for Job Control
3: 1 = Job control output on SYSLST
0 = Output not on SYSLST
4: 1 = Job is being run out of sequence with a temporary assignment for SYSRDR
0 = Conditions for 1 setting not met
5-6: Reserved
7: 1 = Batch command just issued
0 = Condition for 1 setting did not occur

60 Binary disk address of the volume label area (label cylinder).

62 → 76 As illustrated (Figures for information blocks, I/O tables, and pointers begin at Figure 8, which refers to more detailed figures).

78 Set to the value nn specified in the LINES=nn parameter of the STDJC macro.

79 The format of the system date contained within this field is determined by the IPL program from information supplied in the date convention byte (displacement 53). Bytes 85-87 contain the day count.

88 Bytes reserved for use by LIOCS. Transient dump programs insert a key to indicate to the LIOCS end-of-volume routine, $$BCMT07, that it was called by a B-transient.

90 Address of the first part of the program information block (PIB) table. (See Figures 13 and 14.)

92 ID number of the last checkpoint.

94 Length of the LUBID queue (in bytes). This equals the number of channel queue entries. It can also be used to access the REQID queue. (See Figure 12.)

Figure 6. Supervisor Communications Region (Part 4 of 5)

Key to Communications Region Displacements:

96 Address of disk I/O position data. This is the starting address of the disk information block (DIB) table (See Figure 16).

98 Address of the beginning of the error recovery block. The error recovery block contains addresses of error recovery exits, error recovery queue information that can be used by physical transients routines, and defines storage for the error queue entries (See Figure 33).

100 → 104 As illustrated (See Figure 19).

106 Key of the program (BG, F2, or F1) that has timer support.

108 As illustrated (See Figure 12).

110 Logical Transient Key (LTK) contains the same value as the PIK (Displacement 46) when the logical transient is requested. When the transient area is not in use, LTK is equal to zero. The SVC 2 routine sets the LTK. The SVC 11 routine resets the LTK.

112 Register save area (ERA)- not considered COMREG information.

124 Address of second part of program information block (PIB) table (See Figure 15).

126 Address of PDTABB, table of DTF addresses for MICR support (See Figure 20).

128 Address of QTAM vector table (IJLQTTAD).

132 Address of background communications region.

134 Reserved.

136 Pointer to communications region extension (See Below).

BGXTNSN = 4- Byte address of CE table located in supervisor nucleus (See Note).

Note: If communications regions are generated for the foreground partitions, the labels in those extensions will be F2XTNSN and F1XTNSN.

Figure 6. Supervisor Communications Region (Part 5 of 5)

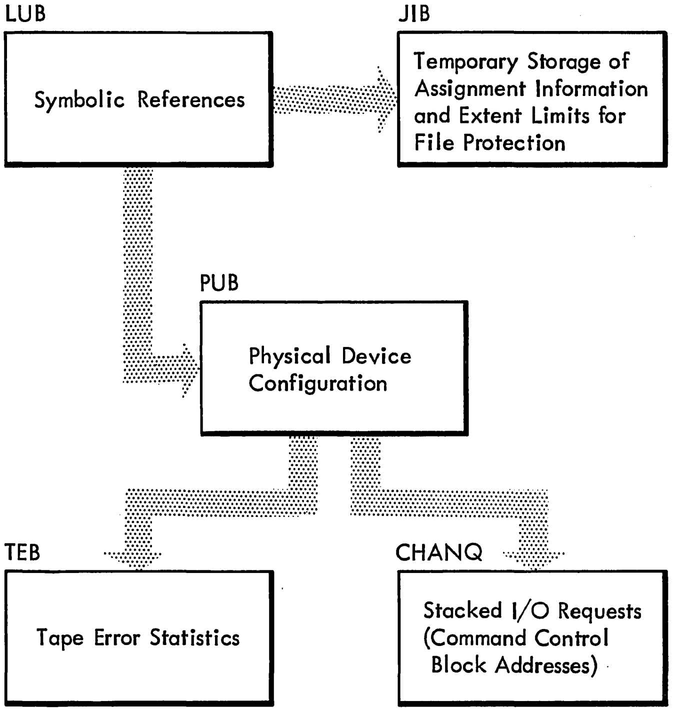
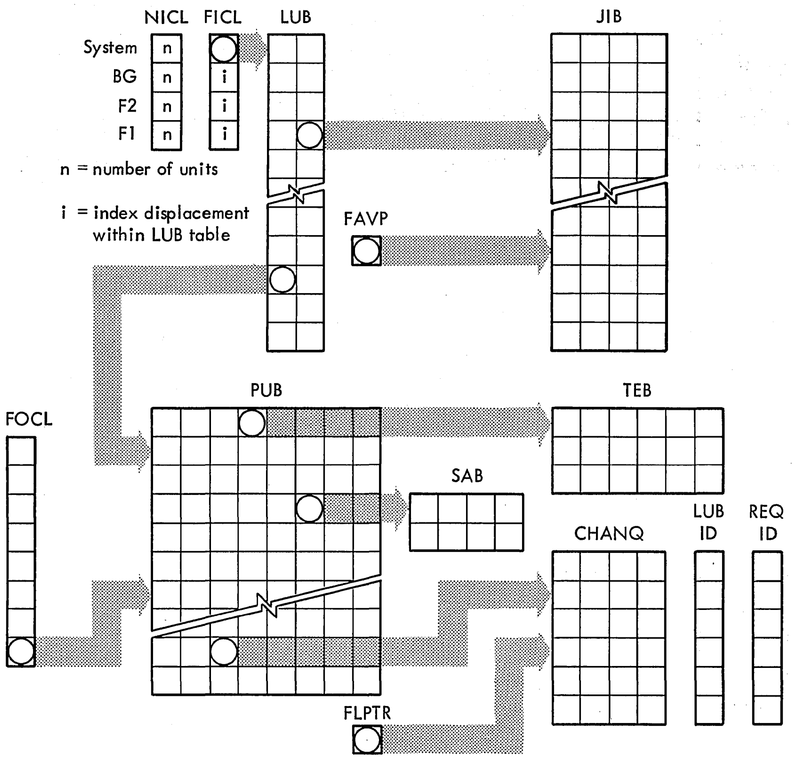

LUB
Symbolic References
JIB
Temporary Storage of Assignment Information and Extent Limits for File Protection
PUB
Physical Device Configuration
TEB
Tape Error Statistics
CHANQ
Stacked I/O Requests (Command Control Block Addresses)

Figure 7. System Control Center



KEY:

NICL (Number in Class) : The first byte contains the number of system class units. The second, third, and fourth bytes contain the number of programmer class units (BG, F2, F1) (Figure 10).

FICL (First in Class) : The first byte points to the first system class unit in the LUB table. (Always the first LUB table entry.) The second byte points to the first programmer class unit in the LUB table BG area. The third points to the first programmer class unit in the LUB table F2 area. The fourth points to the first programmer class unit in the LUB table F1 area (Figure 10).

LUB (Logical Unit Block) Table : The first byte points to a PUB table entry. The second byte points to a JIB table entry (Figure 10).

PUB (Physical Unit Block) Table : The first two bytes contain the channel and unit address of the physical device. The third byte contains a CHANQ pointer. The fourth byte contains a TEB pointer. The fifth byte contains the SAB pointer for DASD with seek separation, or device type code information for tape devices. The sixth byte is reserved for device options. The seventh and eighth bytes contain channel scheduler and job control flags respectively (Figure 9).

FOCL (First on Channel List) : The first byte points to the first PUB (highest priority) on channel zero. The next byte points to the first PUB (highest priority) on channel one, etc. A hexadecimal FF indicates the associated channel is not supported.

TEB (Tape Error Block) : One TEB is built for each tape unit at supervisor generation time if tape error statistics are required (Figure 11).

FAVP (First Available Pointer) : A one-byte pointer to the next available JIB entry.

JIB (Job Information Block) : The first two bytes contain extent or LUB information. The third contains ownership and JIB flags. The fourth contains JIB chaining information (Figure 17).

CHANQ (Channel Queue) Table : The first byte contains the chain field (a pointer to the next in queue). The last three bytes contain the CCB address (Figure 12).

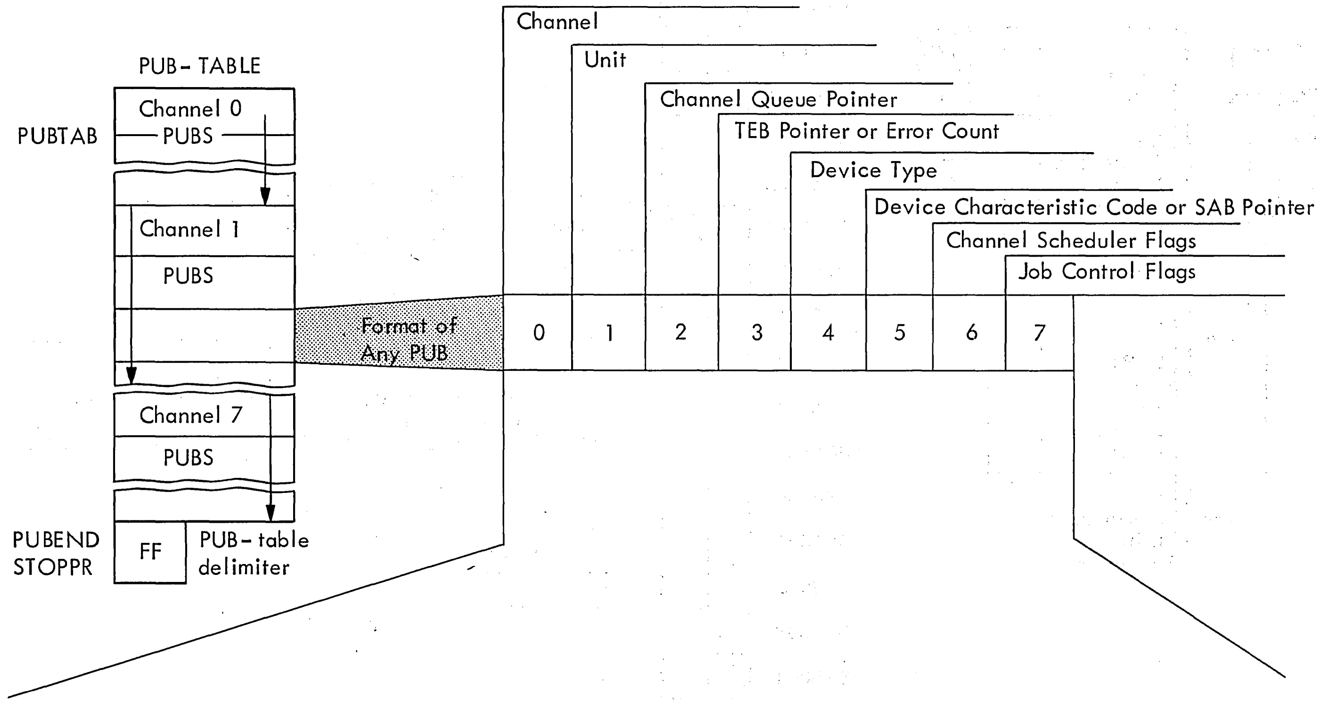
LUBID : A one-byte pointer to the LUB making the I/O request.

REQID : A one-byte pointer to the program containing the CCB (Figure 12).

FLPTR : A one-byte pointer to the next free entry in the channel queue (Figure 12).

SAB : A four-byte (BCCH) address that is the current disk address of the device.

Figure 8. I/O Table Interrelationship



BYTE 0 - Channel number. (Hex 0-7, FF=NULL)

BYTE 1 - I/O device unit number. (HEX 1F=1052, HEX 80=magnetic tape unit 0 . . .).

BYTE 2 - HEX 0, 1, 2, . . . points to the first channel queue entry for this device.

BYTE 3 - If device is a magnetic tape unit and TEBS are specified, this byte is a TEB pointer (HEX 1, 2, 3 . . .).

If device is a magnetic tape unit but TEBS are not specified, this byte is an error counter.

If device is not a magnetic tape unit, this byte is an error counter.

BYTE 4 - See Figure 21 for device type codes.

BYTE 5 - SS of the MODE=parameter in the DVCGEN macro for a tape unit. (See Figure 22.)

For DASD with seek separation, this byte is used as the SAB pointer.

BYTE 6 -

Bit 0: 1 = Device busy
1: 1 = Switchable device
2: 1 = EOJ for SYSRDR or SYSIPT
3: 1 = I/O error queued for recovery
4: 1 = Operator intervention required
5: 1 = Device end posting required
6: 1 = Burst device on MPX
7: 1 = 7-track tape unit

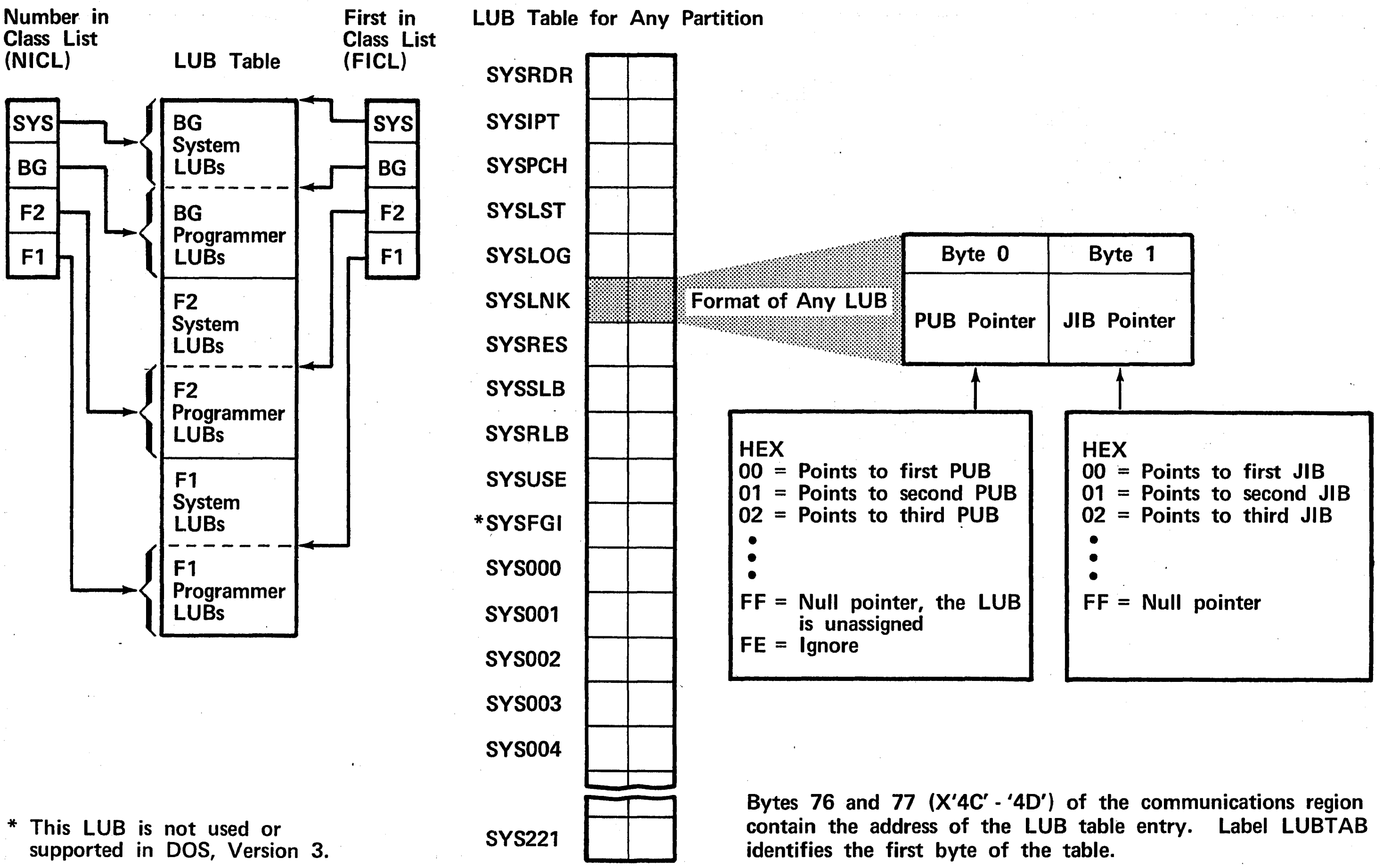
BYTE 7 -

Bit 0-4: standard MODE assignment for 7-track tape (all ones if not tape, all zeros if device is down).
5: Device is assigned to a background job
6: device is assigned to a foreground 1 job
7: device is assigned to a foreground 2 job

NOTE: A null PUB is generated for each device to be supported by the supervisor.
Standard physical unit assignments are made to the PUB table at supervisor generation time. PUBs are ordered by channel and priority within a channel.

Bytes 64 and 65 (X'40'- '41') of the communications region contain the address of the PUB table entry.
Label PUBTAB identifies the first byte of the table.

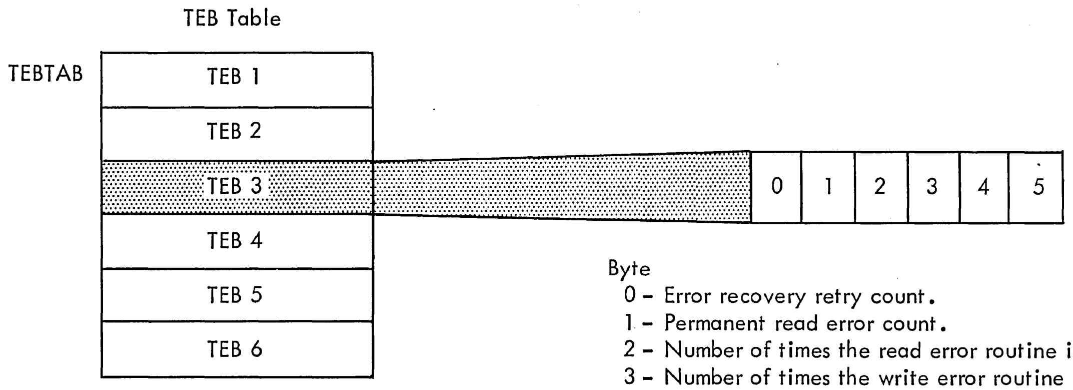
Figure 9. Physical Unit Block (PUB) Table



* This LUB is not used or supported in DOS, Version 3.

Bytes 76 and 77 (X'4C' - '4D') of the communications region contain the address of the LUB table entry. Label LUBTAB identifies the first byte of the table.

Figure 10. Logical Unit Block (LUB) Table

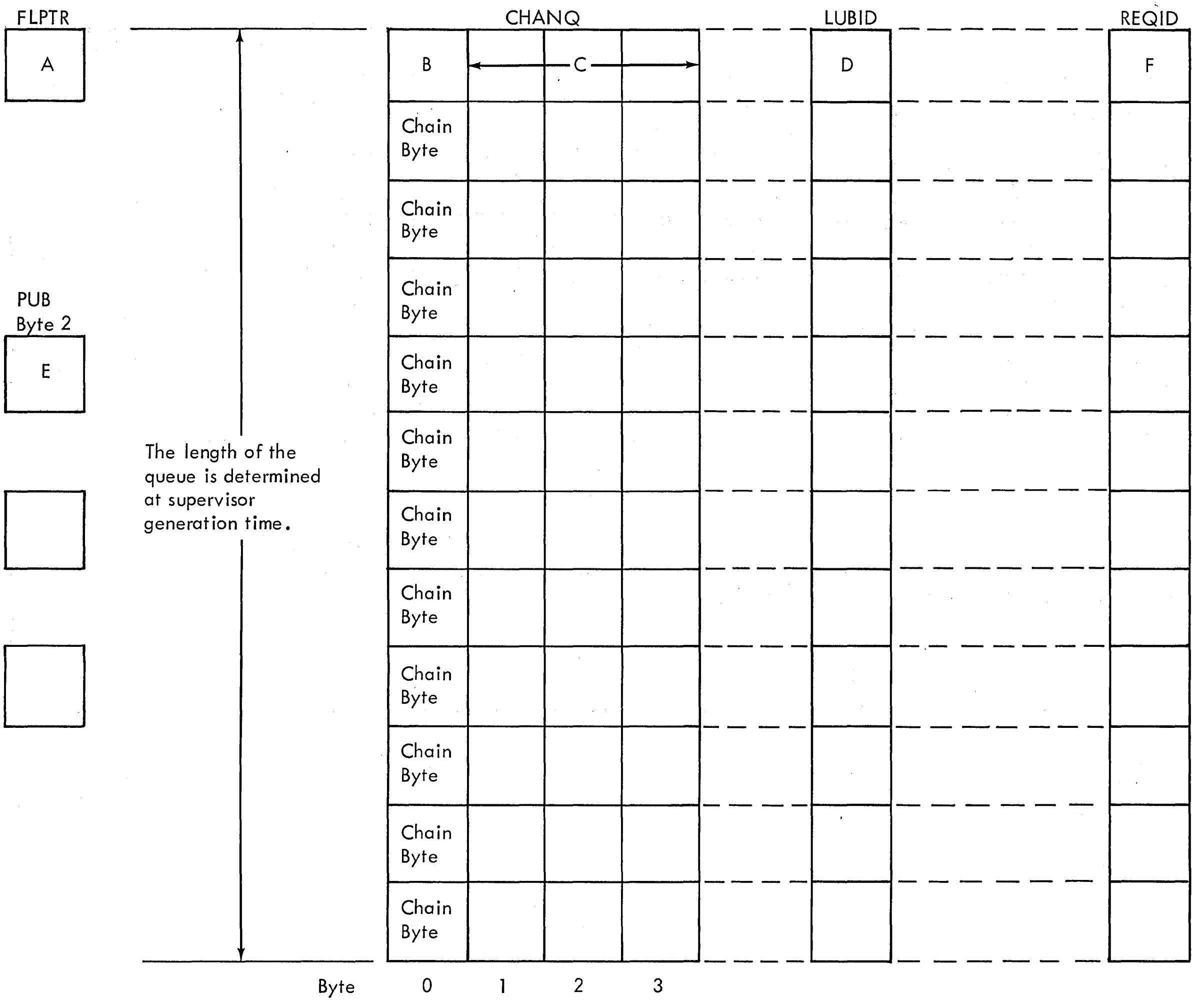


Byte
0 - Error recovery retry count.
1 - Permanent read error count.
2 - Number of times the read error routine is entered.
3 - Number of times the write error routine is entered.
4 - Write skip (erase gap) count.
5 - Noise record count.

One TEB is generated for each magnetic tape unit if the CONFG macro contains the TEB=yes parameter.
Job Control resets each TEB at normal or abnormal End-of-Job. An unused TEB contains HEX'FF0000000000'.
A TEB is referenced from byte 3 of a magnetic tape unit PUB.

Bytes 70 and 71 (X'46' - '47') of the communications region contain the address of the TEB table entry.
Label TEBTAB identifies the first byte of the table.

Figure 11. Tape Error Block (TEB)


FLPTR
A
CHANQ
B
C
Chain Byte
LUBID
D
REQID
F
PUB Byte 2
E
The length of the queue is determined at supervisor generation time.
Byte
0
1
2
3

KEY

A. The free list pointer contains a displacement index to a free list entry within the channel queue. The free list is a group of entries that function in essentially the same manner as a device queue. When the free list pointer contains a hexadecimal FF, it indicates that no more free list entries are available.

B. The first byte of the channel queue entry (chain byte) contains a pointer (displacement index) to the next channel queue entry for that device. A hexadecimal FF indicates the last channel queue entry for that device. New requests on a given device are queued at the end of a given device queue.

C. CCB address for the specified device.

D. A pointer (displacement index) to the LUB table identifying the logical unit making the I/O request.

E. Contains a pointer (displacement index) to the first channel queue entry for a specific device (Figure 9).

F. Contains a code identifying the program making the I/O request. The one-byte entry is called a RID (Requestor Identification). The RID indicates what program the CCB belongs to. The RID is in the form X'nk'.

n = user-storage protection key (supervisor=0, BG=1, F2=2, F1=3).
k = 0 for all user requests and all supervisor CCBs, where n = 0.
k = 1 for supervisor CCBs to SYSLOG that bypass ID prefix.
k = 2 for a fetch CCB.
nk = FF for any unused channel queue entries.

Bytes 108-109 (X'6C'-'6E') of the communications region contain the address of the LUBID Table. Label LUBIDTAB identifies the first byte of the table. The addresses of the other two tables are not at fixed locations. They can be found in the program listing cross-reference by using the labels CHANQ and REQIDTAB.

Figure 12. CHANQ, LUBID, REQID Tables

PIB TABLE

Byte Number	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
All Bound PIB	Flag Byte See A *	Reserved	SP Prefix		Branch Instruction to the All Bound Routine				Reserved							
Problem Program PIB (Note 1)	Flag Byte See B *	Cancel Code (Fig. 25)	SYSLOG ID (BG, F2, or F1)		NOP Instruction (CR)	Address of the Partition Save Area			Number of Core Blocks (Note 2)	Address of the Origin of the Partition			PIB Assign Flag See D *	User LUB Index	Number of Program LUBs	Flag Byte See C *
Attention PIB	Flag Byte See E *	Cancel Code (Fig. 25)	SYSLOG ID (AR)		Branch Code (BC)	Active = Address of Save Area Inactive = Remainder of BC Instruction			Switch Byte See F *	Logical Transient Bucket (contains save area address)			X '07' See D *	Reserved	Address of the Logical Transient	
Quiesce PIB	Flag Byte See A *	Cancel Code (Fig. 25)	Reserved		Branch Instruction to Quiesce I/O Routine				Reserved							
Supervisor PIB	Flag Byte See A *	Cancel Code (Fig. 25)	SP Prefix		Branch Instruction to General Exit Routine				Address of SYSRES PUB		Reserved		X '07' See D *	Reserved		

=16 Byte Length

Note 1: The PIB table is built in this sequence when the MPS feature is selected as a generation option:

All Bound PIB
Background PIB
Foreground 2 PIB
Foreground 1 PIB
Attention PIB
Quiesce PIB
Supervisor PIB

When a batch-only environment is established at generation time, the All Bound and Foreground PIB's are excluded from the table.

Note 2: Number is in multiples of 1K for BG or of 2K for F2 and F1.

* See Figure 14 for flag byte expansions

Bytes 90 and 91 (X'5A'-'5B') of the communications region contain the address of the first part of the PIB Table. Label PIBTAB identifies the first byte of the table.

Figure 13. First Part of Program Information Block (PIB) Table

A Supervisor, Quiesce, and All Bound PIB Flags:

Bit 0: 1 = Registers stored
0 = Registers not stored
1-4: 0 = Always zero
5: 1 = Always one
6: 0 = Always zero
7: 1 = Active
0 = Inactive

B Problem Program PIB Flag (First Byte in PIB):

Bit 0: 1 = Registers stored
0 = Registers not stored
1-4: 0 = Always zero
5: 0 = Normal execution
1 = Program has seized the system
6: 1 = Unbound
0 = SVC 2-bound (B-transient in progress)
7: 1 = Unbound
0 = SVC 7-bound (waiting for an I/O interrupt)
X'80' indicates the program is not present in the system.

C Problem Program PIB Flag (Last Byte in PIB):

Bit 0: 1 = Batched Job in Foreground
0 = No BJF
1: Reserved
2: 1 = /& on SYSIN if DASD
0 = No /& on SYSIN
3-7: Reserved

D PIB Assign Flag

X'80' = SYRES DASD file protect inhibited (allow write operation on SYSRES)
X'40' = Channel appendage exit allowed (BTAM)
X'20' = Cancel in progress (used in terminator function)
X'10' = Cancel control (set on a foreground cancel)
X'08' = Hold-Release flag for foreground assignments
X'07' = Supervisor or Attention routine PIB assign flag setting
X'04' = Background program PIB assign flag setting
X'02' = Foreground 1 program PIB assign flag setting
X'01' = Foreground 2 program PIB assign flag setting

E Attention PIB Flag

Bit 0: 1 = Registers stored
0 = Registers not stored
1-5: 0 = Always zero
6: 1 = Attention routine active
0 = Attention routine SVC 2-bound
7: 1 = Active
0 = SVC 7-bound

X'80' indicates the attention routine is not present in the system.

F Attention PIB Switch Byte

Bit 0-4: Reserved
5: 1 = Physical Attention Recall Switch ON
0 = Physical Attention Recall Switch OFF
6: 1 = Attention Request Switch ON
0 = Attention Request Switch OFF
7: 1 = External Interrupt Request Switch ON
0 = External Interrupt Request Switch OFF

Figure 14. PIB Flag Expansions

Byte Number	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	= 16 Byte Length
Background PIB	Address of BG Comm. Region		System LUB Index		Reserved												
FG2 PIB *	Address of Area Comm. Region **		System LUB Index														
FG1 PIB *	Address of Area Comm. Region **		System LUB Index														
Attention PIB	Address of BG Comm. Region		0	0													

* Generated only if MPS is specified.

** Always background communications region except when MPS=BJF.

Bytes 124 and 125 (X'7C' - '7D') of the communications region contain the address of the second part of the PIB Table. Label PIB2AD identifies the first byte of the table.

Figure 15. Second Part of Program Information Block (PIB) Table

	Current Address										End Address						R	U.L.	L.L.		R.C.		Reserved	
SYSLNK	B	B	C	C	H	H	R	*	← This area not used for SYSLNK DIB →															
SYSIN	B	B	C	C	H	H	R	K	D	D	B	B	C	C	H	H	X	H	H	*	XX	XX		
SYSPCH																				*				
SYSLST																				*				
Number of Bytes	7							3			6						1	1	1	1	2		2	

KEY: Current Address: The next address to be used (for both input and output).

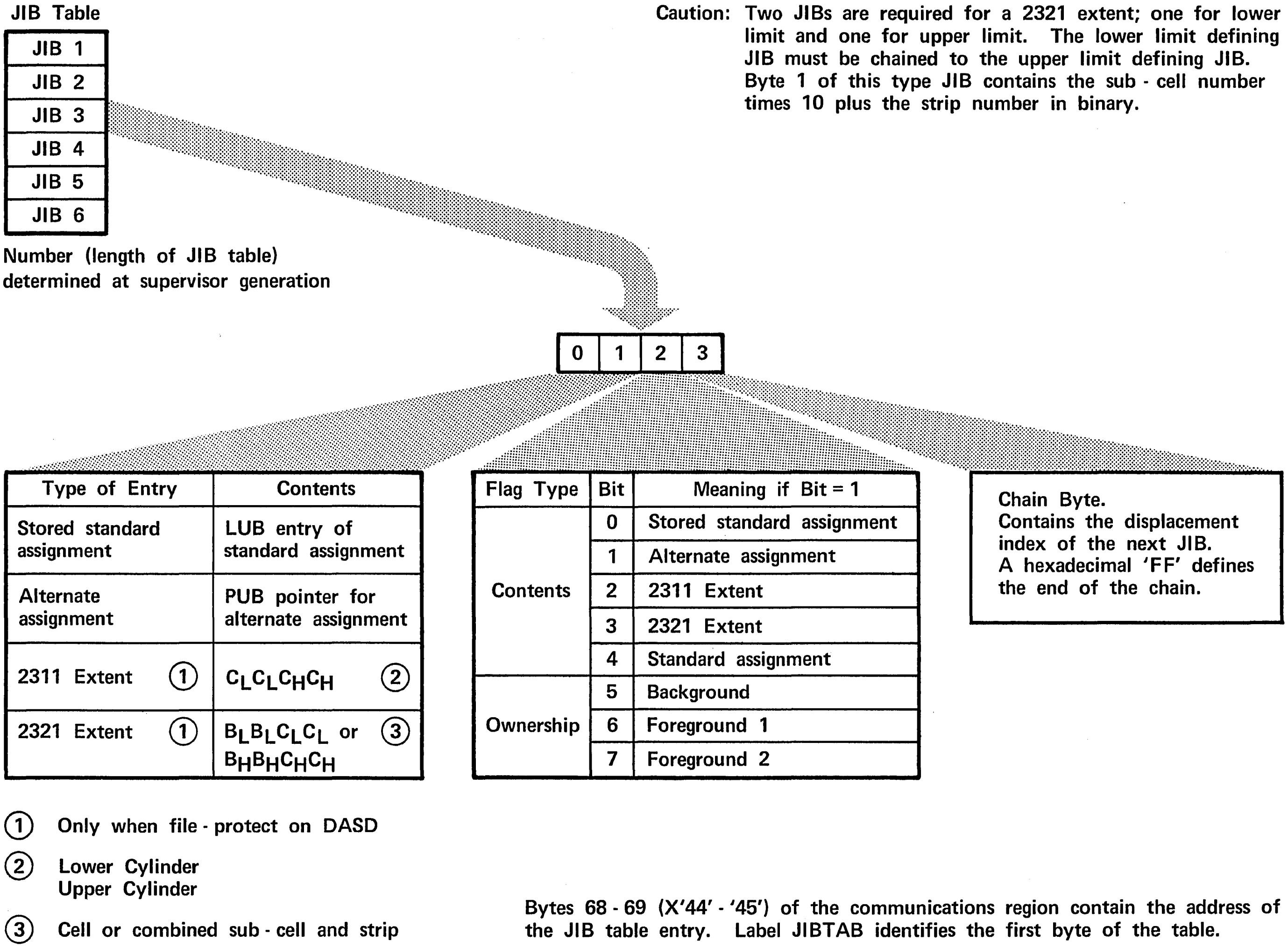
* Not used

End Address : The last address within the limits of the extent.
R : Record count
U.L. : Upper head limit
L.L. : Lower head limit
R.C. : Last record number on track (U.L. minus 1). When the end address minus the current address is less than, or equal to, the value in R.C., a warning message is issued by job control.
KDD : Key and data length for the symbolic device.

KDD for SYSIN = X'000050'
KDD for SYSPCH = X'000051'
KDD for SYSLST = X'000079'

Bytes 96 and 97 (X'60'- '61') of the communications region contain the address of the SYSLNK entry.
Label DSKPOS identifies the first byte of the table.

Figure 16. Disk Information Block (DIB) Table



JIB Table

JIB Table
JIB 1
JIB 2
JIB 3
JIB 4
JIB 5
JIB 6

Number (length of JIB table) determined at supervisor generation

Caution: Two JIBs are required for a 2321 extent; one for lower limit and one for upper limit. The lower limit defining JIB must be chained to the upper limit defining JIB. Byte 1 of this type JIB contains the sub - cell number times 10 plus the strip number in binary.

0	1	2	3

Type of Entry	Contents
Stored standard assignment	LUB entry of standard assignment
Alternate assignment	PUB pointer for alternate assignment
2311 Extent ①	$C_LC_LC_HC_H$ ②
2321 Extent ①	$B_LB_LC_LC_L$ or ③ $B_HB_HC_HC_H$

Flag Type	Bit	Meaning if Bit = 1
Contents	0	Stored standard assignment
	1	Alternate assignment
	2	2311 Extent
	3	2321 Extent
	4	Standard assignment
Ownership	5	Background
	6	Foreground 1
	7	Foreground 2

Chain Byte.
Contains the displacement index of the next JIB.
A hexadecimal 'FF' defines the end of the chain.

① Only when file - protect on DASD

② Lower Cylinder
Upper Cylinder

③ Cell or combined sub - cell and strip

Bytes 68 - 69 (X'44' - '45') of the communications region contain the address of the JIB table entry. Label JIBTAB identifies the first byte of the table.

Figure 17. Job Information Block (JIB) Table

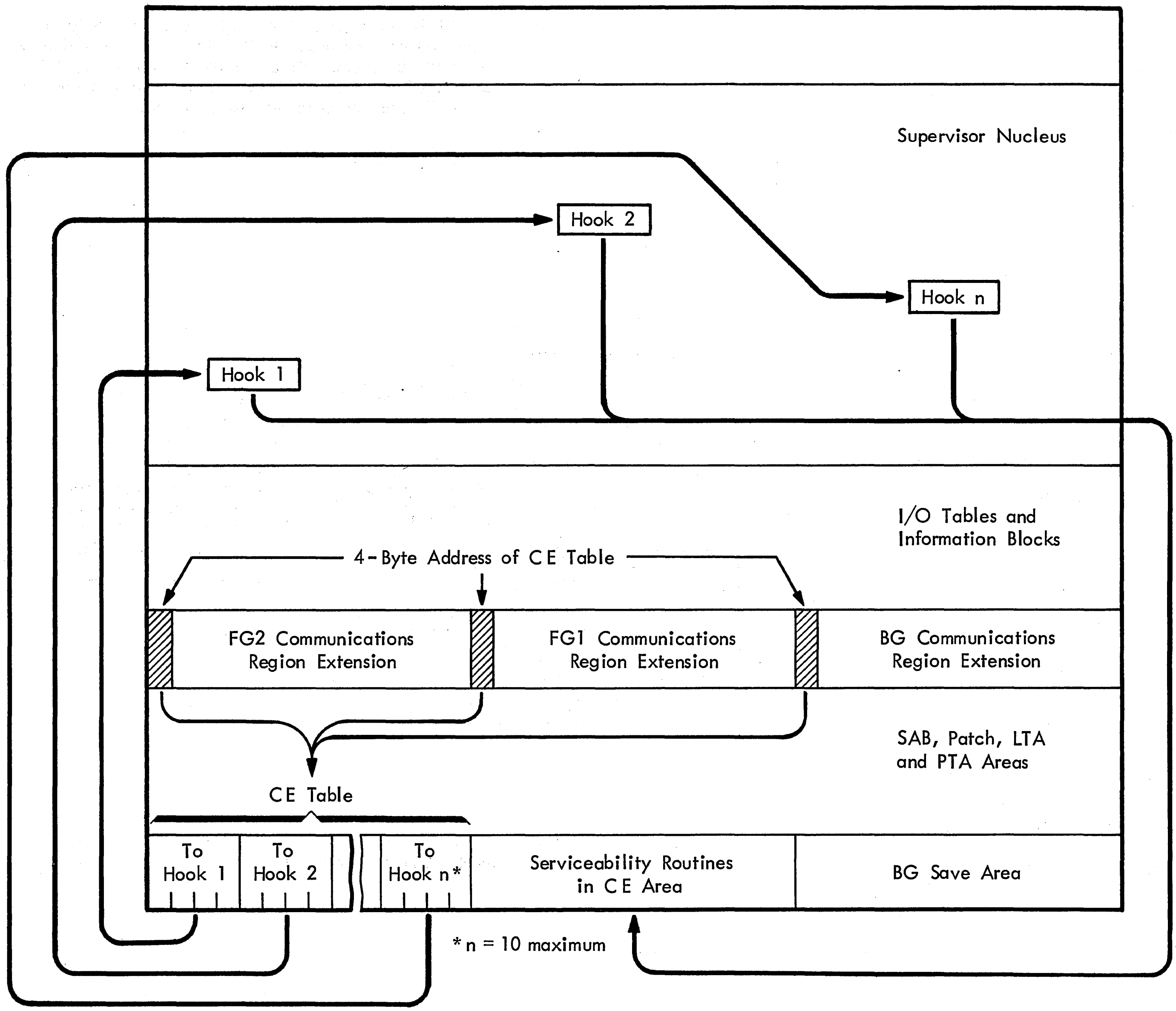

Supervisor Nucleus
Hook 2
Hook n
Hook 1
I/O Tables and Information Blocks
4-Byte Address of CE Table
FG2 Communications Region Extension
FG1 Communications Region Extension
BG Communications Region Extension
SAB, Patch, LTA and PTA Areas
CE Table
To Hook 1
To Hook 2
To Hook n*
Serviceability Routines in CE Area
BG Save Area
*n = 10 maximum

Figure 18. Accessing CE Serviceability Routines

PC Option Table and OC Option Table:

	A	B
BG	a	b
F2	a	b
F1	a	b
	← word →	← word →

A

No STXIT given: a = 0
STXIT issued: a = address of the user program check (operator communications) routine
STXIT issued when the user routine is already in use: a = complement of user program check (operator communications) routine address

B

No STXIT given: b = 0
STXIT issued: address of the user save area

IT Option Table:

C	D
c	d
← word →	← word →

C

No TECB or STXIT issued: c = 0
TECB issued: c = address of the timer event control block
STXIT issued: c = address of the user interval timer routine
STXIT issued when user routine is already in use: c = complement of the user interval time routine address

D

No TECB or STXIT issued: d = 0
TECB issued: d = complement of the TECB address
STXIT issued: d = address of the user save area

The addresses of these tables are found in the communications region at the byte locations specified below. The labels shown identify the first byte of the corresponding table.

Table	Bytes in COMREG	Label
PC	100- 101 (X'64'- '65')	PCTAB
IT	102- 103 (X'66- '67')	ITTAB
OC	104- 105 (X'68'- '69')	OCTAB

Figure 19. Option Tables

INFORMATION BLOCKS AND OTHER TABLES

To accomplish functions such as exit selection, DASD file protection, and record identification, the supervisor program requires pertinent information. At supervisor generation time certain main storage locations are set aside and, in some cases, initialized to supply the required information. The basic information blocks and their respective functions are:

PIB (Program Information Block): The first half of a PIB retains program status information for user and supervisor programs. It supplies routing information in a multiprogramming environment to allow selective program return and it contains pointers and switches used by the supervisor program. (Figures 13 and 14.) The second half contains the address of the area communications region and the system LUB Index (Figure 15).

DIB (Disk Information Block): The DIB is built at generation time if the SYSFIL option was selected. It performs a record keeping function on system class units assigned to a DASD. The DIB contains the current seek address when the system is operating in a batched job environment. The block is initialized by job control with extent information and updated by physical IOCS. It is located in the SGTCON macro expansion. (See Figure 16.)

JIB (Job Information Block): The JIB contains temporary and alternate LUB assignments. (These blocks are referenced by the LUBs.) The JIB serves another purpose when DASD file protection is selected as a supervisor generation option. Extent information is supplied by the program initiator and logical IOCS open transient routines. The supervisor can then perform the file protect function for the specified file limits. File protection does not include supervisor and transient originated I/O. (See Figure 17.)

SAB (Seek Address Block): Contains a four-byte address (BCCH) for each DASD device when the seek separation feature is specified. The current address is maintained in the SAB for the particular device. Each SAB is referenced by its corresponding PUB.

Other Tables: Several optional tables are built at generation time within the supervisor:

- CE Table - Generated in the SEND macro expansion. Contains 4-byte pointers, each of which points to a hook within the supervisor that accesses the CE serviceability routines. Provision is made for up to ten entries in the CE table. See Figure 18 for an illustration of this table, how it is accessed, and how the hooks are used.

- PC, OC, IT Tables - Generated in the SGTCON macro expansion. These three tables (program check, operator communications, and interval timer) contain addresses supplied by the user with a STXIT macro. (See Figure 19.)

- DTF Address - Generated in the SMICR macro expansion. These two tables (PDTABA and PDTABB) contain pointers, DTF addresses, and other information for handling external interrupts on magnetic ink character recognition devices. (See Figure 20.)

Additional Communications Regions: The communications regions for the two foreground partitions are located at the end of the I/O tables. Their format is identical to that of the background communications region. Immediately following the F2 and F1 communications regions (or, if not BJF, following the I/O tables) are extensions of as many communications regions as have been generated. The extensions are presently generated only if the CE option is specified. However, they will provide for future COMREG needs without adding to the size of the basic supervisor.

Transient, CE, and Save Areas: Main storage locations are reserved in the area preceding the problem program area for:

1. Logical ($$B) transients - 1200 bytes.

2. Physical ($$A) transients - approximately 550 bytes.

3. CE area (only if CE=YES or n) - 600 bytes minimum.

4. Background save area - 88 bytes (120 for floating point feature). The BG save area contains six subfields: program name, PSW, general registers (9 through 8), label length, 6 reserved bytes, and optionally floating point registers (0-8). (See Figure 1.)

The table of DTF addresses (PDTABB) contains six 8-byte entries; one for each external line of the direct control feature on the system.

PDTABB

Byte →	0	1	2	3	4	5	6	7
↓						DTF address for MICR:		
0	NI	PDSTAT +1,		X'FE'	Ownership Flags	Device on line 7		
8	NI	PDSTAT +1,		X'FD'		Device on line 6		
16	NI	PDSTAT +1,		X'FB'		Device on line 5		
24	NI	PDSTAT +1,		X'F7'		Device on line 4		
32	NI	PDSTAT +1,		X'EF'		Device on line 3		
40	NI	PDSTAT +1,		X'DF'		Device on line 2		

Background = 10
Foreground 2 = 20
Foreground 1 = 30

- Bytes 0-3 --Contain an 'AND' instruction that is executed in main line coding to turn off the external line status after its detection.

  PDSTAT +1 will contain one or more of the following interrupt codes:

PSW Interrupt Code Bit	Interrupt Code (PSW Bits 26-31)*	External Interrupt Cause
31	nnnnnnn1	External signal 7
30	nnnnnn1n	External signal 6
29	nnnnn1nn	External signal 5
28	nnnn1nnn	External signal 4
27	nnn1nnnn	External signal 3
26	nn1nnnnn	External signal 2

- Byte 4--Contains the flag of the partition containing the DTF.

- Bytes 5-7--Contain the address of the DTF table.

Table of pointers (PDTABA) to DTF addresses associated with the external interrupt line. The table is set up to handle the status in descending order from Bit 31 to Bit 26 of the external old PSW.

PDTABA

Byte →	0	1	2	3	4	5	6	7
0	00	08	00	10	00	08	00	18
8	00	08	00	10	00	08	00	20
16	00	08	00	10	00	08	00	18
24	00	08	00	10	00	08	00	28
32	00	08	00	10	00	08	00	18
40	00	08	00	10	00	08	00	20
48	00	08	00	10	00	08	00	18
56	00	08	00	10	00	08	00	

*n =other external-interrupt conditions.

Bytes 126 and 127 (X'7E' - '7F') of the communications region contain the address of these tables. Label PDTABB identifies the first byte of the first table.

Figure 20. Tables for MICR DTF Addresses and Pointers

Card Code	Actual Device	Dev. Type X'nn'	Device Type
2400T9	9-track Magnetic Tapes	50	Tapes
2400T7	7-track Magnetic Tapes	50	
1442N1	1442N1 Card Reader Punch	30	Card Readers-Punches
2520B1	2520B1 Card Reader Punch	31	
2501	2501 Card Reader	10	Card Readers
2540R	2540 Card Reader	11	
2540P	2540 Card Punch	21	Card Punches
2520B2	2520B2 Card Punch	20	
1442N2	1442N2 Card Punch	22	
2520B3	2520B3 Card Punch	20	
1403	1403 Printer	40	Printers
1403U	1403 Printer with UCS Feature	42	
1404	1404 Printer	40	
1443	1443 Printer	41	
1445	1445 Printer	41	
1050A	1052 Printer-Keyboard	00	
UNSP	Unsupported Device	FF	Unsupported. No burst mode on multiplexor channel.
UNSPB	Unsupported Device	FF	Unsupported with burst mode on multiplexor channel.
2311	2311 Disk Drive	60	DASD
2314	2314 Disk Storage Facility	62	
2321	2321 Data Cell Drive	61	
1412	1412 Magnetic Char Reader	75	MICR- Magnetic Ink Character Recognition Devices
1419	1419 Magnetic Char Reader	72	
1419P	1419 Dual Address Adapter Primary Control Unit	73	
1419S	1419 Dual Address Adapter Secondary Control Unit	74	
2701 *	2701 Line Adapter Unit	D0	Teleprocessing lines A = SAD0 command when enabling the line B = SAD1 command when enabling the line C = SAD2 command when enabling the line D = SAD3 command when enabling the line
2702 A B C D	2702 Transmission Control Unit.	D1	
2703	2703 Transmission Control	D2	
2671	2671 Paper Tape Reader	70	Paper Tape Reader
1285	1285 Optical Reader	76	Optical Readers
1287	1287 Optical Reader	77	

Note: The codes used in the DVCGEN macros are the same codes used in IPL statements.

* For other teleprocessing devices, see IBM System/360, DOS BTAM and QTAM PLMs, Forms Y30-5001 and Y30-5002.

Figure 21. Device Type Codes

Density (Bytes per Inch)	Parity	Convert Feature	Translate	ss
200	odd	on	off	10
200	odd	off	off	30
200	odd	off	on	38
200	even	off	off	20
200	even	off	on	28
556	odd	on	off	50
556	odd	off	off	70
556	odd	off	on	78
556	even	off	off	60
556	even	off	on	68
800	odd	on	off	90
800	odd	off	off	B0
800	odd	off	on	B8
800	even	off	off	A0
800	even	off	on	A8
800	dual density nine-track			C8
1600	dual density nine-track			C0

Figure 22. Density Data

RESIDENT SUPERVISOR (CHARTS 01-06)

Supervisor is the storage resident portion of the Disk Operating System. It is loaded into storage at IPL time and remains there throughout system operations. Refer to the preceding section of this manual for information about generation of the resident supervisor. Refer to Figure 5 for information about the storage organization of the resident supervisor.

Infrequently-used supervisory functions are not included in the resident supervisor. They are in the form of transient programs (A and B) and are fetched or loaded from the core image library when needed.

SUPERVISOR INTERRUPT PROCESSORS

This portion of the resident supervisor processes the following system interrupts:

- Supervisor call interrupt
- I/O Interrupt
- Program check interrupt
- External interrupt
- Machine check interrupt.

MULTIPROGRAMMING SUPPORT (MPS)

General Entry and General Exit routines provide the mechanism for multiprogramming support. Refer to these areas on Chart 01 for additional descriptions for multiprogramming concepts. Figure 23 illustrates the task selection procedure associated with multiprogramming.

BATCH JOB SUPPORT

Batch jobs may be run in any of the three partitions (BG, F2, F1). Batch job support is always provided in the background partition. To run in the foreground partitions, the MPS=BJF option must be specified.

SUPERVISOR CALL INTERRUPT (SVC)

SVC is detected by microprogramming, which loads the SVC new PSW. The SVC interrupt processor (Chart 03) analyzes the SVC code placed in the SVC old PSW by the calling program. Control is transferred to the appropriate processing routine. Some SVCs are optional and cause a cancel if supervisor was generated without the option. (See Figure 24 for a list of supervisor calls.)

SVC 0: Execute the user's channel program (EXCP). The address of the user's command control block (CCB) must be supplied in general register 1 before this SVC is issued. Return may be either to the interrupted program or to the highest priority program ready to run.

Note: When an SVC 0 is issued by supervisor or A-Transient programs, the address of the CCB must be supplied in general register 15 before the SVC is issued.

SVC 1: Fetches a phase. A fetch loads a phase from the core image library and branches to the entry address in that phase. The load and entry addresses are obtained from the core image directory entry for the phase being fetched. The storage address of the phase name must be supplied in general register 1 before this SVC is issued. The user may override the linkage editor entry address by supplying an entry address in general register 0. Return may be either to the interrupted program or to the highest priority program ready to run.

SVC 2: Fetches a B-transient. Loads a B-transient program (phase name prefix equals $$B) from the core image library to the B-transient area (refer to Figure 5), and enters the B-transient at its load address plus 8 bytes. The storage address of the B-transient phase name must be supplied in general register 1.

An address in general register 0 is ignored. The B-transient is loaded at the beginning address of the B-transient area. General register 15 is loaded with this address and may be used by B-transients as a base register. Return may be either to the interrupted program or the the highest priority program ready to run.

Only one program can use the B-transient area at a time. If the B-transient program is SVC 7 bound, another program is selected. This program becomes SVC 2 bound (waiting for the B-transient area) if it issues an SVC 2. Another program is then selected.

Note: Supervisor may branch directly to the SVC 2 routine when fetching a B-transient. If the transient is not in the library when referenced by the supervisor, the system enters the wait state.

SVC 3: Fetches or returns from an A-transient. Load an A-transient program (phase name prefix equals $$A) from the core image library to the A-transient area (refer to Figure 5), and enters the A-transient at its load address plus 8 bytes. The storage address of the A-transient phase name is loaded in general register 1 before the fetch is made.

An address in general register 0 is ignored. The A-transient is loaded at the beginning address of the A-transient area. General register 11 is loaded with this address and is used by A-transients as a base register. Return is to the interrupted program.

Note: Supervisor may branch directly to the SVC 3 routine when fetching an A-transient. Only programs operating in the supervisor mode can issue an SVC 3. If the transient is not in the library, the system enters the wait state.

Caution: SVC 3 is also used as a return from an A-transient program. The last byte of the A-transient name field determines the usage.

- X'00' - Returning from error recovery A-transients.

- X'01' - Returning from physical attention transients ($$ANERRZ, Y, 0) or post cancel by any A-transient.

- Alpha - Fetch A-transient.

When returning from an A-transient, the branch address is in general register 15. The A-transient must load one of the exit addresses from the error recovery block (ERBLOC). Refer to Figure 33.

SVC 4: Loads a phase from the core image library and returns to the user. (See the following Note.) The storage address of the phase name must be supplied in general register 1 before this SVC is issued. The user may override the link-edited load address by supplying a load address in general register 0. Upon return to the user, general register 1 contains the phase entry address adjusted for any changes in the phase's load address.

Note: Return may be either to the interrupted program or to the highest priority program ready to run.

SVC 5: Modifies the supervisor communications region. Supplies the supervisory support for the MVCOM macro. The sequence of events is:

1. MVCOM macro issues an SVC 5.

2. SVC 5 fetches $$ANERR1 by branching to the SVC 3 routine.

3. $$ANERR1 alters the supervisor communications region as specified by the MVCOM macro.

Return may be either to the interrupted program or to the highest priority program ready to run.

SVC 6: Cancels a background or foreground program. Cancel code X'23' is posted to the PIB for the program issuing the SVC 6. Refer to Figure 13 for the format of the PIB tables, to Chart 03 for General Cancel Routine, and Figures 25 and 40 for cancel codes. The next time the canceled program is selected on general exit, a branch is made to the SVC 2 routine to fetch the cancel B-transient program, $$BEOJ3.

SVC 7: Waits for I/O to complete or a timer interrupt to occur. SVC 7 supplies the supervisory support for the WAIT macro.

With MPS option, returns directly to the interrupted program if the traffic bit has been posted in the CCB or TECB. See SVC 24 in this list for an explanation of the TECB. If traffic bit is not posted:

1. Change the status of the interrupted program PIB to SVC 7 bound (not ready to run).

2. Select the highest priority program that is ready to run.

When I/O is completed or a timer interrupt occurs:

1. The traffic bit is posted in the CCB or TECB.

2. The PIB is restored to the ready-to-run status.

3. When this program is again selected at general exit, the old PSW is loaded with the address of the second instruction of the WAIT macro expansion.

Without MPS option, returns directly to the interrupted program if the traffic bit has been posted in the CCB or TECB. (See SVC 24 in this list for an explanation of the TECB.) If the traffic bit is not posted, the system enters the wait state with interrupts enabled.

SVC 8: Supplies the supervisory support to temporarily return from a B-transient program to the problem program. The B-transient area is not released. The task selection exit loads the problem program registers. An SVC 9 is used to return to the B-transient program.

SVC 9: Supplies the supervisory support for returning to the B-transient after an SVC 8 is issued. The task selection exit loads the B-transient registers.

SVC 10: Sets a timer interval. This SVC is optional, and the issuing program is canceled if supervisor is generated without the IT option. Only the timer supported program can issue an SVC 10. Others are canceled.

The time interval is specified in general register 1 by the user (SETIME macro). The system time of day (SYSTOD, X'54') is updated to the time that the next interrupt should occur (may change if another SVC 10 is issued). The system timer (SYSTIMER, X'50') is set to the specified time interval. The time interval in SYSTIMER immediately begins to lapse. Refer to IBM System/360 Principles of Operation, Form A22-6821, for information concerning the operation of SYSTIMER.

Note: Current system time of day can be obtained by shifting out the low order byte from the remaining time interval (SYSTIMER) and subtracting it from system time of day (SYSTOD). Time in SYSTOD is represented in the form, seconds x 300. Time in SYSTIMER is in the form, seconds x 300 x 256.

An SVC 10 returns directly to the timer supported program. No task selection is performed.

SVC 11: Returns from a B-transient releasing the B-transient area. SVC 11 is invalid if issued by other than a B-transient. The logical transient area is released for use by other programs or tasks. Return is to the highest priority program ready to run.

SVC 12: Supplies the supervisory support to reset flags to 0 in the linkage control byte (displacement 57 in the supervisor communications region). The user loads a mask (1 byte, hexadecimal) into general register 1. This mask is ANDed with the linkage control byte. An SVC 12 returns directly to the interrupted program. No task selection is performed.

SVC 13: Supplies the supervisory support to set flags to 1 in the linkage control byte (displacement 57 in the supervisor communications region). The user loads a mask (1 byte, hexadecimal) into general register 1. This mask is ORed with the linkage control byte. An SVC 13 returns directly to the interrupted program. No task selection is performed.

SVC 14: This is the normal end of job (EOJ). Cancel code X'10' is posted to the PIB for the program issuing the SVC 14. Refer to Figure 13 for the format of the PIB tables and to Chart 03 for the General Cancel routine. The next time the canceled program is selected on general exit, a branch is made to the SVC 2 routine to fetch the cancel B-transient program $$BEOJ3. Job Control is loaded by $$BEOJ to perform the end-of-job-step.

SVC 15: This is the same as SVC 0 (EXCP), with this exception: when the CHANQ table is full, the SVC is ignored. Return is directly to the interrupted program in this case. If the CHANQ table is not full, general register 0 is zeroed and EXCP is issued (see SVC 0 in this list).

Note: The CHANQ table is full when the free list pointer (FLPTR) equals X'FF'. Refer to Figure 12 for the format of the CHANQ table and to Figure 26 for CHANQ operation.

SVCs 16 through 21: These supervisor calls provide supervisory support for the STXIT and EXIT macros. They are optional, and the issuing program is canceled if supervisor was not generated with the applicable option.

- SVC 16 stores the address of the user's program check (PC) routine and save area address in the PC option table.

- SVC 17 provides a return from the user's PC routine to the program interrupted due to a program check.

- SVC 18 stores the address of the user's interval timer (IT) routine and save area address in the IT option table. Only the timer supported program can issue SVC 18.

- SVC 19 provides a return from the user's IT routine to the timer supported program. Only the timer supported program can issue SVC 19.

- SVC 20 stores the address of the user's operator communications (OC) routine and save area address in the OC option table.

- SVC 21 provides a return from the user's OC routine to the program interrupted by the external interrupt key.

The address of the user routine is specified in general register 0, and the address of the user's save area is specified in general register 1 in all cases. Refer to Figure 19 for the format of the option tables.

SVCs 16, 18, and 20 return directly to the interrupted program. SVCs 17, 19, and 21 return either to the interrupted program or to the highest priority program ready to run.

SVC 22: Seizes the system and provides a release from such a seizure. The SVC 22 is ignored if supervisor was generated without the MPS option. The program issuing an SVC 22 is canceled if the PSW protection key field does not equal 0. (Only job control and B-transient programs can issue an SVC 22.)

The first SVC 22 issued seizes the system and the next one issued releases the system. The last byte of register 0 replaces the system mask. If register 0 is negative, the protection key is replaced by the protection key of the PIK.

The task selection mechanism is altered by the first SVC 22 so that only supervisor or quiesce I/O tasks and the program that issued the SVC 22 can be selected. The next SVC 22 issued restores the task selection mechanism. The contents of the last byte of general register 0 are again used as the system mask.

Return from each SVC 22 is directly to the interrupted program.

Caution: There is no way to cancel a program that has seized the system.

- The program must have no pending I/O operations.

- The program cannot issue supervisor calls while the system is seized.

SVC 23: Loads phase header. Retrieves the load address for a specified phase from the core image directory. The program issuing an SVC 23 is canceled if supervisor was generated without the MPS option or if the PSW protection key does not equal 0. (Only job control and B-transient programs can issue an SVC 23.)

The user must specify the address of the core image phase name in general register 1 and the address where the load address is to be stored in general register 0. The main fetch subroutine scans the core image directory and retrieves the load address. If the phase is found in the directory, the load address (3 bytes) is stored at the user's address specified by general register 0. If the phase is not found, return is to the interrupted program.

SVC 24: Stores the address of the user's timer event control block (TECB) and sets a timer interval. This SVC is optional, and the issuing program is canceled if supervisor is generated without IT option. Only the timer supported program can issue an SVC 24. Others are canceled.

The address of the user's TECB is specified in general register 0, and the time interval is specified in general register 1.

The traffic bit is reset in the user's TECB, and the TECB address is stored in the IT option table. Refer to Figure 19 for the format of the IT option table.

Note: The TECB has the same format as a command control block (CCB), but only the traffic bit is used. The traffic bit is set when a timer interrupt occurs. Refer to Figure 27 for the format of the CCB.

The time interval is set, and the system time of day is updated as for an SVC 10. (See SVC 10.) An SVC 24 returns directly to the timer supported program. No task selection is performed.

The user causes the program to wait for the timer interrupt to occur by issuing an SVC 7. (See SVC 7 in this list.)

SVC 25: Issues halt I/O on a Teleprocessing device. If supervisor is generated without Teleprocessing option, a program issuing an SVC 25 is canceled.

The address of any command control block (CCB) containing the symbolic unit address for this device must be supplied in general register 1 before issuing this SVC.

An HIO instruction is issued to the device if:

1. It is a Teleprocessing device and

2. There is I/O pending for the device.

In this case, return is to the highest priority program ready to run. The device busy flag is reset at this time. If an SVC 25 is issued for other than a Teleprocessing device, it is ignored.

SVC 26: Validate address limits. The program issuing an SVC 26 is canceled if the PSW protection key does not equal 0. (Only job control and B-transient programs can issue an SVC 26.)

The upper address must be specified in general register 2, and the lower address must be specified in general register 1. The upper address must be within main storage, and the lower address must be higher than the end of supervisor address, or the program is canceled (ERR25). Return is to the interrupted program. No task selection is performed.

With MPS option, the PIK of the program issuing the SVC 26 must equal the storage protection key for both addresses or the program is canceled (ERR25).

With batch operation, SVC 26 is ignored unless storage protection has been specified.

SVC 27: Same as SVC 25, except the EXCP CCB is not dequeued if the CSW has been stored after a HIO command.

SVC 28: Provides return from user's stacker select routine to the MICR external interrupt routine in the supervisor. This SVC is optional and causes a cancel if issued at any point other than in a user stacker select routine with MICR devices.

SVC 29: Provides supervisory support for the WAITF macro for MICR devices only. If MICR devices are not being used, logical IOCS does not issue SVC 29. If MICR devices are not specified in the supervisor macro (SUPVR), SVC 29 causes a cancel.

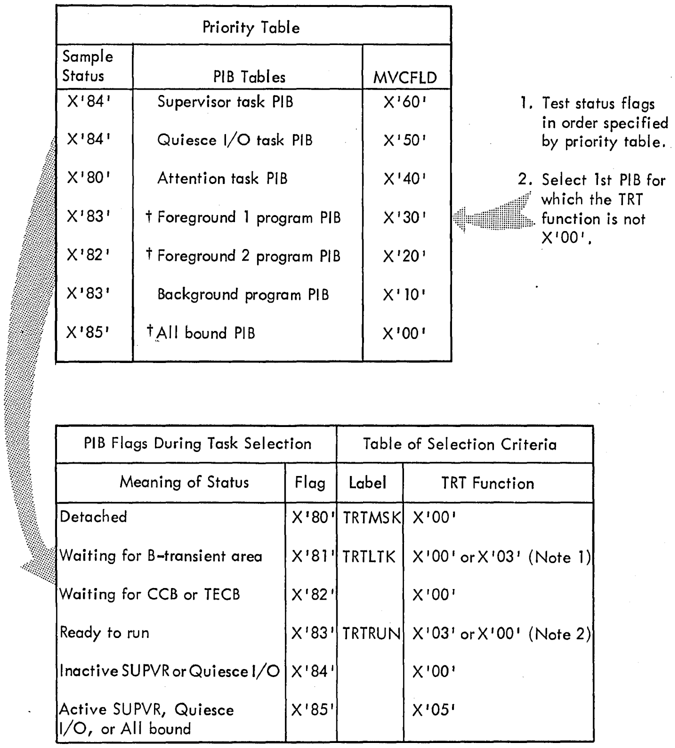
All interrupts are disabled and the CCB's are all checked for the traffic bit. When a CCB is found with the traffic bit posted, SVC 29 returns to the interrupted program.

If all CCB's are checked and no traffic bits are posted, one of two courses is taken:

- With MPS option - Cause user to become I/O-bound, disable for I/O interrupts only, and return to task selection.
- Without MPS option - Set wait bit in SVC old PSW, disable for I/O interrupts only, and return to interrupted program.

SVCs 30, 31, and 32: Reserved for QTAM. Refer to the QTAM PLM listed in the Preface.

SVCs 33 and 34: Reserved for internal macros COMRG and GETIME, respectively. Their use by other programs results in a branch to EXT01 (see Chart FA).



Priority Table		
Sample Status	PIB Tables	MVCFLD
X'84'	Supervisor task PIB	X'60'
X'84'	Quiesce I/O task PIB	X'50'
X'80'	Attention task PIB	X'40'
X'83'	† Foreground 1 program PIB	X'30'
X'82'	† Foreground 2 program PIB	X'20'
X'83'	Background program PIB	X'10'
X'85'	†All bound PIB	X'00'

1. Test status flags in order specified by priority table.

2. Select 1st PIB for which the TRT function is not X'00'.

PIB Flags During Task Selection		Table of Selection Criteria	
Meaning of Status	Flag	Label	TRT Function
Detached	X'80'	TRTMSK	X'00'
Waiting for B-transient area	X'81'	TRTLTK	X'00' or X'03' (Note 1)
Waiting for CCB or TECB	X'82'		X'00'
Ready to run	X'83'	TRTRUN	X'03' or X'00' (Note 2)
Inactive SUPVR or Quiesce I/O	X'84'		X'00'
Active SUPVR, Quiesce I/O, or All bound	X'85'		X'05'

Note 1: X'00' when the B-transient area is in use.

Note 2: X'00' when a task has seized the system. That task's status flag will equal X'84' or X'85'.

† These PIB's are generated for MPS option only.

Figure 23. Task Selection Procedure

Macro Supported	SVC	Function
EXCP	0	Execute channel programs.
FETCH	1 2 3	Fetch any phase. Fetch a logical transient (B-transient). Fetch or return from a physical transient (A-transient).
LOAD	4	Load any phase.
MVCOM	5	Modify supervisor communications region.
CANCEL	6	Cancel a problem program.
WAIT	7	Wait for a CCB or TECB.
	8	Transfer control to the problem program from a logical transient (B-transient).
LBRET	9	Return to a logical transient (B-transient) from the problem program after a SVC 8.
SETIME	10*	Set timer interval.
	11 12 13	Return from a logical transient (B-transient). Logical AND (Reset) to second Job Control byte (displacement 57 in communications region). Logical OR (Set) to second Job Control byte (displacement 57 in communications region).
EOJ	14	Cancel job and go to Job Control for end of job step.
	15	Same as SVC 0 except ignored if CHANQ table is full. (Primarily used by ERP).
STXIT (PC)	16*	Provide supervisor with linkage to user's PC routine for program check interrupts.
EXIT (PC)	17*	Return from user's PC routine.
STXIT (IT)	18*	Provide supervisor with linkage to user's IT routine for interval timer interrupts.
EXIT (IT)	19*	Return from user's IT routine.
STXIT (OC)	20*	Provide supervisor with linkage to user's OC routine for external or attention interrupts (operator communications).
EXIT (OC)	21*	Return from user's OC routine.
	22* 23* 24* 25* 26* 27*	The first SVC 22 seizes the system for the issuing program by disabling multiprogram operation. The second SVC 22 releases the system (enables multiprogram operation). Load phase header. Phase load address is stored at user's address. Provide supervisor with linkage to user's TECB and set timer interval. Issue HALT I/O on a Tele-processing device. Validate address limits. Special HIO on teleprocessing devices.
EXIT (MR)	28*	Return from user's stacker select routine (MICR devices only).
	29*	Provide return from multiple wait (WAITF) macro for MICR devices only.
	30 through 34	Reserved (See preceding text).

* = optional

Figure 24. DOS Supervisor Calls

Type	Cancel Code	Condition	Label
Logical Cancels	X'10'	Normal EOJ	ERR10
	X'20'	Program check	ERR20
	X'21'	Illegal SVC	ERR21
	X'22'	Phase not found	ERR22
	X'23'	Program request	ERR23
	X'24'	Operator intervention	ERR24
	X'25'	Invalid address limit	ERR25
	X'26'	Unassigned LUB code	ERR26
	X'27'	Invalid LUB code in CCB	ERR27
	X'28'	QTAM cancel in progress	EXT02
Logical I/O Cancels	X'30'	Reading past /& on SYSRDR or SYSIPT.	ERR30
	X'31'	Error queue overflow or no CHANQ entry available for ERP.	ERR31
	X'32'	DASD address not within JIB extents.	ERR32
	X'33'	No long seek in user's channel program.	ERR33
	X'40'	Load $$BEOJ	EXT02
	X'80'	Cancel occurred in LTA	EXT02

Figure 25. Supervisor Cancel Codes

I/O INTERRUPT

This is detected by microprogramming, which loads the I/O new PSW. Refer to the I/O Interrupt Processor on Chart 04.

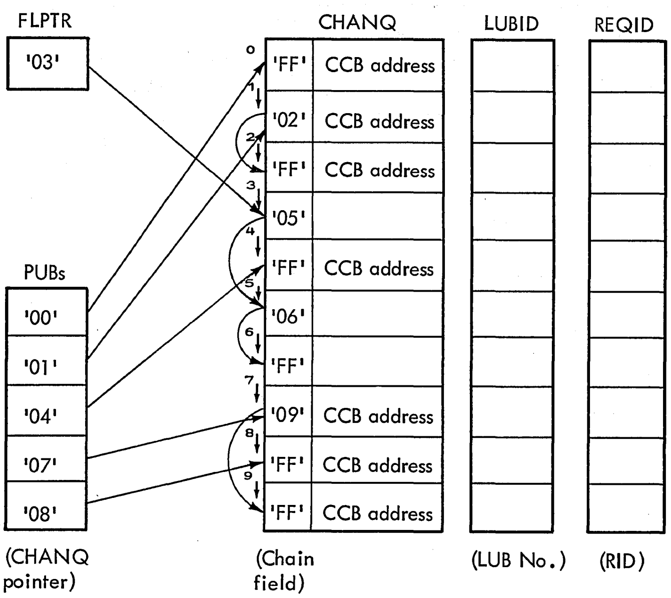


Figure 26. Example of the CHANQ Table Operation

PROGRAM CHECK INTERRUPT

This is detected by microprogramming, which loads the program check new PSW. Refer to Program Check Interrupt Processor on Chart 02.

EXTERNAL INTERRUPT

This is detected by microprogramming, which loads the external new PSW. External interrupts can be caused by:

- Timer
- External interrupt key
- Signal

Refer to External Interrupt Processor on Chart 05.

MACHINE CHECK INTERRUPT

This is detected by microprogramming, which loads the machine check new PSW. The SEREP action code (S) is stored in storage location 0001, and the system enters the wait state. Refer to Chart 01.

Physical IOCS is that portion of the resident Supervisor that:

- Builds a schedule of I/O operations for all devices on the system (CHANQ table). Refer to Channel Scheduler on Chart 04. Also, see Figure 26 for CHANQ operation.
- Starts the actual I/O operations on a device (SIO). Refer to Actual I/O on Chart 04.
- Schedules the starting of all I/O operations and monitors all events associated with I/O. Refer to I/O Interrupt Processor on Chart 04.
- Performs error recovery procedures (ERP). Refer to Unit Check, Quiesce I/O, ERP Exits, and Resident Disk Error Recovery on Chart 06. Figures 27 through 31 illustrate: Command Control Block (CCB), Channel Command Word (CCW), Program Status Word (PSW), Channel Address Word (CAW), and Channel Status Word (CSW). See Figures 32 and 33 for CSW testing and error recovery block layout, respectively.

COMMAND CONTROL BLOCK

The CCB establishes communication between the problem program and physical IOCS. The CCB is two double words in length with eight major fields, as shown in Figure 27. All data in the CCB is in the hexadecimal format. The eight fields of the CCB are listed and described as follows:

1. Count Field (bytes 0, 1): Contains the residual count, which is stored in these two bytes by PIOCS when the CCB is removed from the queue.
2. Transmission Information (bytes 2, 3): Used for communication between PIOCS and the problem program.

Note: Bytes 0 through 3 are ANDed off, by PIOCS, when the CCB is placed in the queue. Communication bits that were set on by the problem program are left on because an AND instruction is used by PIOCS for resetting bytes 0 through 3.

3. CSW Status Bits (bytes 4, 5): Contains the CSW status information, which is stored in these two bytes by PIOCS before control is returned to the problem program.

Note: The particular bits that are turned on in bytes 2 through 5 indicate the conditions that were detected by the problem program and PIOCS.

4. Symbolic Unit Address (bytes 6, 7): Contains the 2-byte hexadecimal representation of SYSnnn. This value represents the location of the logical unit in the LUB table (see Figure 27) and is placed in the CCB by the problem program.
5. Byte 8: Is not used, and must contain hexadecimal 0.
6. CCW Address (bytes 9-11): Contains the address of the CCW that is associated with this CCB. This address is placed in the CCB by the problem program.
7. Byte 12: X'80'-CCB being used by ERP. X'40'-channel appendage routine for a teleprocessing device. X'20'-sense information desired. X'10'-message writer use. X'01'-seek separation option specified.
8. CCW Address in CSW (bytes 13-15): Contains the CCW address from the CSW. This address is stored by PIOCS before control is returned to the problem program. A CCB that has been queued, by PIOCS, to service a problem program I/O request cannot be used for a second problem program I/O request until the first request has been completed.

   Note: Bytes 13-15 contain the address of the channel appendage routine when bit X'40' is set in byte 12.

9. Optional-Sense CCW (bytes 16-23): Bytes 16-23 are appended to the CCB by the CCB macro expansion when the user desires sense information to be returned on unrecoverable I/O errors. The macro expansion also turns on bit 2 (X'20') in byte 12 and bit 7 (X'01') in byte 2 of the CCB. User handles all error or exceptional conditions except program check, protection check, channel control check, and interface control check.

Figure 27. Command Control Block (CCB)

	Count	Transmission Information	CSW Status Bits	Symbolic Unit Address	Not Used Must Contain X'00'	CCW Address	Reserved for Physical IOCS	CCW Address in CSW	Optional Sense CCW
Bytes	0 1	2 3	4 5	6 7	8	9 11	12	13 15	16 23
Used For	Residual Count	Transmitting Information Between Physical IOCS and Problem Program	(Note 1)	Hexadecimal Representation of SYSnnn SYSRDR =0000 SYSIPT =0001 SYSPCH =0002 SYSLST =0003 SYSLOG =0004 SYSLNK =0005 SYSRES =0006 SYSSLB =0007 SYSRLB =0008 SYSUSE =0009 SYS000 =0100 SYS001 =0101 etc.	Not Used	Address of CCW Associated with This CCB	X'80' - CCB being used by ERP X'40' - Channel Appendage Routine Present for Teleprocessing Device X'20' - Sense Information Desired X'10' - Message Writer X'01' - Seek Separation	Address of CCW in the CSW Stored at Channel End, or Address of the Channel Appendage Routine for Teleprocessing Devices	8 Bytes Appended to the CCB when Sense Information is Desired

Byte 4 BIT DESIGNATION	Byte 5 BIT DESIGNATION
32 Attention	40 Program - controlled interruption
33 Status modifier	41 Incorrect length
34 Control unit end	42 Program check
35 Busy	43 Protection check
36 Channel end	44 Channel data check
37 Device end	45 Channel control check
38 Unit check	46 Interface control check
39 Unit exception	47 Chaining check

Byte 2

Bits	0	1	2	3	4	5	6	7
	Traffic Bit (Wait) (Note 5)	End - of - File (/* or /&) (Note 2)	Unrecoverable I/O Error	Accept Unrecoverable I/O Error	Return DASD Data Checks or 2671 errors to the User	Post at Device End (Note 5)	Return Tape Read Data Check, 2540 or 2520 Equipment Check, or DASD Data Checks on Read or Verify Command (Note 3)	User Error Routine
Set On By	PIOCS	PIOCS	PIOCS	Pr. Pr.	Pr. Pr.	Pr. Pr.	Pr. Pr.	Pr. Pr.

Byte 3

Bits	0	1	2	3	4	5	6	7
	DASD - Data Check in Count Area MICR - SCU Not Operational 1285/1287 - Data Check in Journal Tape Mode	DASD - Track Overrun MICR - Intervention Required 1285/1287 - Keyboard Correction in Journal Tape Mode	DASD - End of Cylinder MICR - (Note 4) 1285/1287 - Hopper Empty in Document Mode	2540, 2520 - Equipment Check Tape - Read Data Check DASD - Any Data Check 1285/1287 - Equipment Check	Non - Recovery Questionable Condition: Card - Unusual Command Sequence DASD - No Record Found 1285/1287 - Document Jam or Torn Tape	No - Record - Found Condition (Retry on 2311 or 2314)	Carriage Channel 9 Overflow or Verify Error for DASD 1287 Document Mode - Late Stacker Select	Command Chaining Retry from the next CCW to be executed
Set On By	PIOCS	PIOCS	PIOCS	PIOCS	PIOCS	Pr. Pr.	PIOCS	Pr. Pr.

PIOCS = Physical IOCS
Pr. Pr. = Problem Program

Note 1. Bytes 4 and 5 contain the status bytes of the Channel Status Word (Bits 32 - 47). If byte 2, bit 5 is on and device end results as a separate interrupt, device end will be ORed into CCB byte 4.

Note 2. Indicates /* or /& statement encountered on SYSRDR or SYSIPT. Byte 4, bit 7 (unit exception) is also on.

Note 3. DASD data checks on count not returned.

Note 4. For 1412/1419, Disengage. For 1419D, I/O Error in external interrupt routine (channel data check or busout check).

Note 5. The traffic bit (Byte 2, bit 0) is normally set on at channel end to signify that the I/O was completed. If byte 2, bit 5 has been set on, the traffic bit and bits 2 and 6 in byte 3 will be set on at device end. Also see Note 1.

<table>
<tr><td>Byte</td><td colspan="8">0</td><td colspan="8">1</td><td colspan="8">2</td><td colspan="8">3</td><td colspan="8">4</td><td colspan="8">5</td></tr>
<tr><td>Bit</td><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td><td>8</td><td>9</td><td>10</td><td>11</td><td>12</td><td>13</td><td>14</td><td>15</td><td>16</td><td>17</td><td>18</td><td>19</td><td>20</td><td>21</td><td>22</td><td>23</td><td>24</td><td>25</td><td>26</td><td>27</td><td>28</td><td>29</td><td>30</td><td>31</td><td>32</td><td>33</td><td>34</td><td>35</td><td>36</td><td>37</td><td>38</td><td>39</td><td>40</td><td>41</td><td>42</td><td>43</td><td>44</td><td>45</td><td>46</td><td>47</td></tr>
<tr><td>Name</td><td colspan="8">Command Code</td><td colspan="24">◄ Data Address ►</td><td colspan="5">Flags</td><td colspan="3">0 0 0</td><td colspan="8">(Ignored)</td></tr>
<tr><td>Field</td><td colspan="8">A</td><td colspan="24">B</td><td colspan="5">C</td><td colspan="3">D</td><td colspan="8">E</td></tr>
</table>

<table>
<tr><td colspan="8">6</td><td colspan="8">7</td></tr>
<tr><td>48</td><td>49</td><td>50</td><td>51</td><td>52</td><td>53</td><td>54</td><td>55</td><td>56</td><td>57</td><td>58</td><td>59</td><td>60</td><td>61</td><td>62</td><td>63</td></tr>
<tr><td colspan="16">◄ Count ►</td></tr>
<tr><td colspan="16">F</td></tr>
</table>

FIELD	NAME	DESCRIPTION
A	Command Code	Bits 0-7: Specify the operation to be performed. (See Note on Part 2 of this Figure)
B	Data Address	Bits 8-31: Specify the location of a byte in main storage. It is the first location referred to in the area designated by the CCW.
C	Flags	Bits 32-36: Specify the flag bits used in conjunction with the CCW. Bit 32- Chain-Data (CD) causes the address portion of the next CCW to be used with the current CCW. †Note Bit 33- Chain-Command (CC) causes the command code and data address of the next CCW to be used. The chain data flag (bit 32) takes precedence over this flag. Bit 34- Suppress Length Indication (SLI) causes a possible incorrect length indication to be suppressed. The chain data flag (bit 32) takes precedence over this flag. Bit 35- Skip (SKIP) suppresses the transfer of information to main storage. Bit 36- Program Control Interruption (PCI) causes the channel to generate an interrupt when the CCW is fetched.
D	Reserved	Bits 37-39: (Must contain zeros)*
E	Ignored	Bits 40-47: Not checked
F	Count	Bits 48-63: Specify the number of bytes in the operation

*The transfer in channel command (TIC) is the one exception to this statement.

† Note: Chain data cannot be done on 360/30 if a high-speed device is being used. Example- 2311,2400 mod III.

Figure 28. Channel Command Word (CCW) (Part 1 of 2)

Note,

CHANNEL COMMAND CODES

Command Code assignments are listed in the following table. The symbol X indicates that the bit position is ignored; M identifies a modifier bit.

CODE	COMMAND
MMMM 0 1 0 0	Sense
XXX X 1 0 0 0	Transfer in channel
MMMM 1 1 0 0	Read backward
MMMM MM0 1	Write
MMMM MM1 0	Read
MMMM MM1 1	Control

DASD CHANNEL COMMAND CODES (See A26-5988)

Command for CCW		Count	Multiple Track (M-T) Off			M-T On†	
			8-Bit Code 0123 4567	Hex	Dec	Hex	Dec
Control	No Op	X	0000 0011	03	03		
	Release*	X	0001 0111	17	23		
	Restore	X	0001 0011	13	19		
	Seek	6	0000 0111	07	07		
	Seek Cylinder	6	0000 1011	0B	11		
	Seek Head	6	0001 1011	1B	27		
	Sense I/O	4	0000 0100	04	04		
	Set File Mask	1	0001 1111	1F	31		
	Space Record	X	0000 1111	0F	15		
	Transfer in Channel	X	XXXX 1000	X8			
Search †	Home Address EQ	4 (usually)	0011 1001	39	57	B9	185
	Identifier EQ	5 (usually)	0011 0001	31	49	B1	177
	Identifier HI	5 (usually)	0101 0001	51	81	D1	209
	Identifier EQ or HI	5 (usually)	0111 0001	71	113	F1	241
	Key EQ	1 to 255	0010 1001	29	41	A9	169
	Key HI	1 to 255	0100 1001	49	73	C9	201
	Key EQ or HI	1 to 255	0110 1001	69	105	E9	233
	Key & Data EQ*	Note 1	0010 1101	2D	45	AD	173
	Key & Data HI*		0100 1101	4D	77	CD	205
	Key & Data EQ or HI*		0110 1101	6D	109	ED	237
Read †	Home Address	5	0001 1010	1A	26	9A	154
	Count	8	0001 0010	12	18	92	146
	Record R0	Number of bytes transferred	0001 0110	16	22	96	150
	Data		0000 0110	06	06	86	134
	Key & Data		0000 1110	0E	14	8E	142
	Count, Key & Data		0001 1110	1E	30	9E	158
Write	Home Address	5 (usually)	0001 1001	19	25		
	Record R0	8+KL+DL of R0	0001 0101	15	21		
	Count, Key & Data	8+KL+DL	0001 1101	1D	29		
	Special Count, Key & Data*	8+KL+DL	0000 0001	01	01		
	Data	DL	0000 0101	05	05		
	Key & Data	KL & DL	0000 1101	0D	13		

* Special Feature Note 1. Includes mask bytes in search argument.
† M-T On = M-T Off except, during Search and Read bit 0 = 1 in M-T On.
X = not significant; KL - Key Length DL - Data Length; EQ Equal; HI High

Device	Command for CCW		0	1	2	3	4	5	6	7	Hex	Dec
1052	Read Inquiry BCD		0	0	0	0	1	0	1	0	0A	10
	Read Reader 2 BCD		0	0	0	0	0	0	1	0	02	02
	Write BCD, Auto Carriage Return		0	0	0	0	1	0	0	1	09	09
	Write BCD, No Auto Carriage Return		0	0	0	0	0	0	0	1	01	01
	No Op		0	0	0	0	0	0	1	1	03	03
	Sense		0	0	0	0	0	1	0	0	04	04
	Alarm		0	0	0	0	1	0	1	1	0B	11
2540	Read, Feed, Select Stacker SS	Type AA	S	S	D	0	0	0	1	0		
	Read	Type AB	1	1	D	0	0	0	1	0		
	Read, Feed (1400 compatibility mode only)		1	1	D	1	0	0	1	0		
	Feed, Select Stacker SS	Type BA	S	S	1	0	0	0	1	1		
	PFR Punch, Feed, Select Stacker SS	Type BA	S	S	D	0	1	0	0	1		
	Punch, Feed, Select Stacker SS	Type BB	S	S	D	0	0	0	0	1		

SS	Stacker
00	R1
01	R2
10	RP3

D	Data Mode
0	EBCDIC
1	Column Binary

Device	Command for CCW	0	1	2	3	4	5	6	7	Hex	Dec
1442 N1	Read	M	M	M	0	0	0	1	0		
	Write	M	M	M	0	0	0	0	1		
	Control	M	M	0	0	0	0	1	1		
	No Op	0	0	0	0	0	0	1	1		
	Sense	0	0	M	M	0	1	0	0		

	M	M	M	M	
Read	0	0	X		Eject and SS1
Read	1	0	X		Eject and SS1
Read	0	1	X		Eject and SS2
Read	1	1	X		Eject and SS2
Write	0	0	X		SS1
Write	1	0	X		Eject and SS1
Write	0	1	X		SS2
Write	1	1	X		Eject and SS2
Control	1	0			Eject and SS1
Control	0	1			SS2
Control	1	1			Eject and SS2
Sense			1	1	Punch diagnostic
Sense			0	1	Read diagnostic

X = 0 means EBCDIC mode
X = 1 means Column Binary Mode

Device	Command for CCW	0	1	2	3	4	5	6	7	Hex	Dec
1403 or 1443	Write, No Space	0	0	0	0	0	0	0	1	01	01
	Write, Space 1 After Print	0	0	0	0	1	0	0	1	09	09
	Write, Space 2 After Print	0	0	0	1	0	0	Q	1	11	17
	Write, Space 3 After Print	0	0	0	1	1	0	0	1	19	25
	Write, Skip To Channel N After Print	1	C	H	A	N	0	0	1		
	Diagnostic Read	0	0	0	0	0	0	1	0	02	02
	Test I/O	0	0	0	0	0	0	0	0	00	00
	Sense	0	0	0	0	0	1	0	0	04	04
Carriage Control	Space 1 Line Immediately	0	0	0	0	1	0	1	1	0B	11
	Space 2 Line Immediately	0	0	0	1	0	0	1	1	13	19
	Space 3 Line Immediately	0	0	0	1	1	0	1	1	1B	27
	Skip To Channel N Immediately	1	C	H	A	N	0	1	1		
	No Op	0	0	0	0	0	0	1	1	03	03

C	H	A	N	Channel
0	0	0	1	1
0	0	1	0	2
0	0	1	1	3
0	1	0	0	4
0	1	0	1	5
0	1	1	0	6

C	H	A	N	Channel
0	1	1	1	7
1	0	0	0	8
1	0	0	1	9
1	0	1	0	10
1	0	1	1	11
1	1	0	0	12

Device	Command for CCW	0	1	2	3	4	5	6	7	Hex	Dec
2400 Tape*	Transfer in Channel	0	0	0	0	1	0	0	0	08	08
	Sense	0	0	0	0	0	1	0	0	04	04
	Read Backward**	0	0	0	0	1	1	0	0	0C	12
	Write	0	0	0	0	0	0	0	1	01	01
	Read	0	0	0	0	0	0	1	0	02	02
	Control	0	0	C	C	C	1	1	1		
	Mode Set	D	D	M	M	M	0	1	1		

* 9 track op. forces 800 BPI and odd parity; also, it overrides 7 track but does not reset 7 track. Load/Sys Reset forces 7 track to 500 BPI, odd parity, data converter on, translator off.

C	C	C	Control Codes	Hex	Dec
0	0	0	REW	7	7
0	0	1	RUN	0F	15
0	1	0	ERG	17	23
0	1	1	WTM	1F	31
1	0	0	BSR	27	39
1	0	1	BSF	2F	47
1	1	0	FSR	37	55
1	1	1	FSF	3F	63

D	D	7 Track Density
0	0	200
0	1	556
1	0	800
1	1	800

M M M (Mode Modifiers)	Set Density	Set Odd Parity	Set Even Parity	Data Converter On	Data Converter Off	Translator On	Translator Off	Request TIE (Track In Error)
0 0 0 No Op								
0 0 1 Not Used								
0 1 0 Reset Condition	X	X		X			X	
0 1 1 Nine-track only								X
1 0 0			X		X		X	
1 0 1	X		X		X	X		
1 1 0 Reset Condition	X	X			X		X	
1 1 1	X	X			X	X		

**Overrides Data Converter On

Figure 28. Channel Command Word (CCW) (Part 2 of 2)

<table>
<tr><td>Byte</td><td colspan="8">0</td><td colspan="8">1</td><td colspan="8">2</td><td colspan="8">3</td><td colspan="8">4</td><td colspan="8">5</td><td colspan="8">6</td><td colspan="8">7</td></tr>
<tr><td>Bit</td><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td><td>8</td><td>9</td><td>10</td><td>11</td><td>12</td><td>13</td><td>14</td><td>15</td><td>16</td><td>17</td><td>18</td><td>19</td><td>20</td><td>21</td><td>22</td><td>23</td><td>24</td><td>25</td><td>26</td><td>27</td><td>28</td><td>29</td><td>30</td><td>31</td><td>32</td><td>33</td><td>34</td><td>35</td><td>36</td><td>37</td><td>38</td><td>39</td><td>40</td><td>41</td><td>42</td><td>43</td><td>44</td><td>45</td><td>46</td><td>47</td><td>48</td><td>49</td><td>50</td><td>51</td><td>52</td><td>53</td><td>54</td><td>55</td><td>56</td><td>57</td><td>58</td><td>59</td><td>60</td><td>61</td><td>62</td><td>63</td></tr>
<tr><td>Name</td><td colspan="8">System Mask</td><td colspan="4">Key</td><td colspan="4">CPU Mask</td><td colspan="16">Interruption Code</td><td colspan="2">ILC</td><td colspan="2">CC</td><td colspan="4">Prog. Mask</td><td colspan="24">Instruction Address</td></tr>
<tr><td>Field</td><td colspan="8">A</td><td colspan="4">B</td><td colspan="4">C</td><td colspan="16">D</td><td colspan="2">E</td><td colspan="2">F</td><td colspan="4">G</td><td colspan="24">H</td></tr>
</table>

Figure 29. Program Status Word (PSW)

FIELD	NAME	DESCRIPTION
A	System Mask*	Bits 0-7: Are associated with the I/O channels and external signals as follows: Bit Interruption source 0 Multiplexor channel 1 Selector channel 1 2 Selector channel 2 3 Selector channel 3 4 Selector channel 4 5 Selector channel 5 6 Selector channel 6 Timer 7 Interrupt key External signal *A one-bit equals ON and permits an interrupt.
B	Protection Key	Bits 8-11: Form the CPU protection key. The key is matched with a storage key whenever a result is stored. If the protection feature is not implemented, bits 8-11 must be zero when loaded and are zero when stored.
C	CPU Mask (AMWP)	Bits 12-15: Form the CPU mask as follows: Bit Meaning (A) 12 If 1 - generate extended ASCII code If 0 - generate EBCDIC (M) 13 If 1 - permits machine check interrupt If 0 - prohibits machine check interrupt (W) 14 If 1 - the CPU is in the wait state If 0 - the CPU is in the running state (P) 15 If 1 - the CPU is in the problem mode If 0 - the CPU is in the supervisor mode
D	Interruption Code	Bits 16-31: Identify the cause of the interruption. (See NOTE for specific interruption codes.)

FIELD	NAME	DESCRIPTION
E	Instruction Length Code	Bits 32 and 33: Indicate the length, in halfwords, of the instruction last executed, as follows: 00 (0) Not available (unpredictable) 01 (1) 1 halfword 10 (2) 2 halfwords 11 (3) 3 halfwords
F	Condition Code	Bits 34 and 35: Indicate the last condition code setting. All instructions do not set a condition code. 00 Condition code 0 01 Condition code 1 10 Condition code 2 11 Condition code 3
G	Program Mask**	Bits 36-39: Form the program mask for the following program exceptions. Bit Exception 36 Fixed-point overflow 37 Decimal overflow 38 Exponent underflow 39 Significance **A one-bit equals ON and permits a program check interrupt for a specific exception.
H	Instruction Address	Bits 40-63: Indicate the address of the leftmost byte of the next instruction to be executed.

NOTE

SOURCE IDENTIFICATION	INTERRUPTION CODE PSW BITS 16-31	MASK BITS	ILC SET	EXE-CUTION
Input/Output (old PSW 56, new PSW 120, priority 4)				
Multiplexor channel	00000000 aaaaaaaa	0	x	complete
Selector channel 1	00000001 aaaaaaaa	1	x	complete
Selector channel 2	00000010 aaaaaaaa	2	x	complete
Selector channel 3	00000011 aaaaaaaa	3	x	complete
Selector channel 4	00000100 aaaaaaaa	4	x	complete
Selector channel 5	00000101 aaaaaaaa	5	x	complete
Selector channel 6	00000110 aaaaaaaa	6	x	complete
Program (old PSW 40, new PSW 104, priority 2)				
Operation	00000000 00000001		1,2,3	suppress
Privileged operation	00000000 00000010		1,2	suppress
Execute	00000000 00000011		2	suppress
Protection	00000000 00000100		0,2,3	suppress/ terminate
Addressing	00000000 00000101		0,1,2,3	suppress/ terminate
Specification	00000000 00000110		1,2,3	suppress
Data	00000000 00000111		2,3	terminate
Fixed-point overflow	00000000 00001000	36	1,2	complete
Fixed-point divide	00000000 00001001		1,2	suppress/ complete
Decimal overflow	00000000 00001010	37	3	complete
Decimal divide	00000000 00001011		3	suppress
Exponent overflow	00000000 00001100		1,2	terminate
Exponent underflow	00000000 00001101	38	1,2	complete
Significance	00000000 00001110	39	1,2	complete
Floating-point divide	00000000 00001111		1,2	suppress
Supervisor Call (old PSW 32, new PSW 96, priority 2)				
Instruction bits	00000000 rrrrrrrr		1	complete
External (old PSW 24, new PSW 88, priority 3)				
External signal 1	00000000 xxxxxxx1	7	x	complete
External signal 2	00000000 xxxxxx1x	7	x	complete
External signal 3	00000000 xxxxx1xx	7	x	complete
External signal 4	00000000 xxxx1xxx	7	x	complete
External signal 5	00000000 xxx1xxxx	7	x	complete
External signal 6	00000000 xx1xxxxx	7	x	complete
Interrupt key	00000000 x1xxxxxx	7	x	complete
Timer	00000000 1xxxxxxx	7	x	complete
Machine Check (old PSW 48, new PSW 112, priority 1)				
Machine malfunction	00000000 00000000	13	x	terminate

a	Device address bits
r	Bits of R_1 and R_2 field of SUPERVISOR CALL
x	Unpredictable

Mask bits 0 - 7 refer to the system mask.

Mask bits 36 - 39 refer to the program mask.

Byte	0								1								2								3							
Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
Name	Key				0	0	0	0	← Command Address →																							
Field	A				B				C																							

FIELD	NAME	DESCRIPTION
A	Protection Key	Bits 0 – 3 form the storage protection key for all commands associated with START I/O. This key is matched with a storage key whenever data is placed in storage. (Must contain zeros whenever storage protection is not implemented.)
B	Reserved	Bits 4 – 7 (Must contain zeros.)
C	Command Address	Bits 8 – 31 Designates the location of the first CCW in main storage associated with the START I/O. (The three low order bits, 29 – 31, must be zeros, specifying a CCW address on integral boundaries of a double word.)

Figure 30. Channel Address Word (CAW)

Byte	0		1	2	3	4	5	6	7
Bit	0 1 2 3	4 5 6 7	8 9 10 11 12 13 14 15	16 17 18 19 20 21 22 23	24 25 26 27 28 29 30 31	32 33 34 35 36 37 38 39	40 41 42 43 44 45 46 47	48 49 50 51 52 53 54 55	56 57 58 59 60 61 62 63
Name	Key	0 0 0 0	← Command Address →			← Status →		← Count →	
Field	A	B	C			D		E	

FIELD	NAME	DESCRIPTION
A	Protection Key	Bits 0-3 form the storage protection key used in the chain of operations at the subchannel.
B	Reserved	(Must be zeros.)
C	Command Address	Bits 8-31 form an address that is eight higher than the address of the last CCW used. *
D	Status	Bits 32-47 identify the conditions in the device and channel that caused the CSW to be stored. Bits 32-39 are obtained over the I/O Interface and indicate conditions detected by the device or the control unit. Bits 40-47 are provided by the channel and indicate conditions associated with the subchannel. Each status bit represents one type of condition as follows:
E	Count	Bits 48-63 form the residual count for the last CCW used.

DEVICE OR CONTROL UNIT	
Bit Position	Designated Condition
32	Attention
33	Status Modifier
34	Control Unit End
35	Busy
36	Channel End
37	Device End
38	Unit Check
39	Unit Exception

CHANNEL/SUBCHANNEL	
Bit Position	Designated Condition
40	Program-Controlled Interrupt
41	Incorrect Length
42	Program Check
43	Protection Check
44	Channel Data Check
45	Channel Control Check
46	Interface Control Check
47	Chaining Check

* This address is not 8 higher on a command reject.

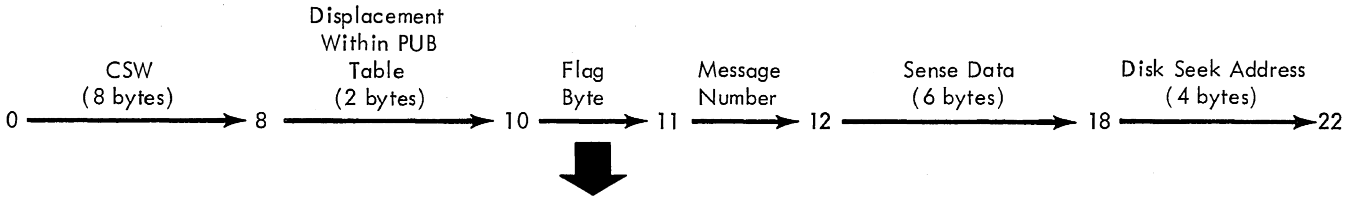
Figure 31. Channel Status Word (CSW)

Status Bit	Status Condition	Action
45 46	Channel control check Interface control check	Enter wait state with all interrupts masked off.
38 42 43 44 47	Unit check Program check Protection check Channel data check Channel chaining check	Exit to unit check on Chart 06 for error recovery.
32	Attention	For attention from a 1052, include attention task in task selection and take general exit (EXT03). Attention interrupts are ignored if: 1. System reallocation or condense is in operation. 2. Attention is not from a 1052.
35	Device busy	Skip channel end test.
36	Channel end	See Chart CL for actions taken. Attempts to reschedule the channel (No attempt is made for the multiplex channel unless this is a burst-multiplex device).
37 34	Device end Control unit end	See Chart CK for actions taken. Attempts to reschedule the channel (If the multiplex channel is being rescheduled. If the device on the multiplex channel is a burst-multiplex device, both channel and device are rescheduled).
33 and 35	Control unit busy	Reset device to available. The status is not tested unless neither channel end, device end, nor control unit end has occurred.

Figure 32. CSW Testing in I/O Interrupt Processor

Displacement from ERBLOC in Decimal	Length in Bytes	Label	Description
-2	2	ERRQ	Address of first error queue entry in table (ERQUE).
0	2	ERBLOC	Address of retry ERP exit (EXRTY).
2	2		Address of ignore ERP exit (EXIGN).
4	2		Address of DISWHY (retry) ERP exit (EXWHY).
6	2	YCHANQ	Address of the channel queue table (CHANQ).
8	2		Address of cancel ERP exit (EXCAN).
10	2	FULQUE	Address of last entry in error queue table.
12	2	ERQPTR	Address of last entry queued to table (initially ERQUE -22).
14	2	RIK	Requestor I/O key (RIK).
16	2		Address of cancel attention exit (ATNCNL).
18	2		Address of attention dequeue exit (PUBDEQ).
20	2	ATNEXT	Address of attention exit (EXT02).
22	8	SVC3NM	A-transient phase name field ($$ANERRb̸).
30	5 x 22	ERQUE	Five 22 bytes error queue entries. (See below)

Each Error Queue Entry (22 bytes)



Bit	Designation
0	Unused
1	Intervention required
2	Passback *
3	Allow ignore
4	DASD device in error
5	Allow retry
6	No CCB available
7	Unused

* Put on by device ERP when user wants control returned on error.

Figure 33. Error Recovery Block (ERBLOC)

PHYSICAL TRANSIENT PROGRAMS (CHARTS 07-09)

Physical transient programs are commonly referred to as A-transients. These infrequently-used sections of the supervisor reside in the core image library and are fetched by the resident supervisor (SVC 3) only when needed. Each program phase name begins with the prefix characters $$A. These phases are loaded singly into the A-transient area. See Figure 5 for Supervisor storage organization. The A-transients functions within DOS are:

1. Provide device-dependent Error Recovery Procedures (ERP).

2. Issue messages associated with ERP operations, Message Writer.

3. Process 1052 attention requests, Physical Attention Routines.

Figure 34 illustrates each A-transient in terms of phase name, function, and program level chart identification.

ERP

To understand the error recovery procedures detailed in the flowcharts, the reader should be familiar with the sense information that corresponds to the individual I/O devices supported by this system.

Figure 35 illustrates the unit record equipment supported by ERP and also indicates the sense bits associated with each device. This figure is followed by ERP descriptions with their corresponding messages.

Caution: Although the disk error recovery procedure is not an A-transient, the sense data and action-taken information is included here to consolidate the sense data in this section of the manual. The disk ERP is part of the supervisor nucleus. See Chart 06.

MESSAGE WRITER

The message writer is a group of seven A-transients that build error messages, issue the message, analyze operator responses, and select the proper exit. See Figure 36 for a listing of the error messages.

PHYSICAL ATTENTION ROUTINES

The physical attention routines are three A-transients fetched by the supervisor when an attention interrupt has been determined. The attention key signals operator communication with the system. If the operator chooses to initiate a foreground program or to use the nonresident attention routine facilities (other B-transients) the physical attention transients get the $$BATTNA root phase. If the operator is satisfying an operator intervention condition or canceling the job, the physical attention transients process the attention interrupt. When the physical attention routines are processing the interrupt, they perform parameter passing by using a common area called the interphase communications area. Figure 37 illustrates this area and its relationship to the entire A-transient area.

Phase Name	Function	Program Level Chart ID
$$ANERRA	Error Recovery Monitor	07
$$ANERRB		07
$$ANERRC		07
$$ANERRD	Tape (2400) Error Recovery	07
$$ANERRE		07
$$ANERRL		07
$$ANERRF		07
$$ANERRG	Data Cell (2321) Error Recovery	07
$$ANERRH		07
$$ANERRI		07
$$ANERRJ		07
$$ANERRK		07
$$ANERRM	Message Writer	08
$$ANERRN		08
$$ANERRO		08
$$ANERRP		08
$$ANERRQ		08
$$ANERRR		08
$$ANERRS		08
$$ANERRT	1412 and 1419 (Single Address Adapter) Error Recovery	08
$$ANERRU	Unit Record Error Recovery	07
$$ANERRV		07
$$ANERRW	1419 (Dual Address Adapter) Error Recovery	08
$$ANERRX	Paper Tape Error Recovery	07
$$ANERR9	Optical Reader Error Recovery	07
$$ANERRZ	Physical Attention	09
$$ANERRY		09
$$ANERR0		09
$$ANERR1	Modify Communications Region	None (See Chart MW)

Figure 34. A-Transient Programs

	DASD	Tapes	Reader Punch	Printers	Printer Keyboard	Paper Tape Reader	Optical Reader	MICR
Device → Sense Bytes	2311, 2314, 2321	2400T7 2400T9	1442, 2501, 2520, 2540	1403, 1404, 1443	1052	2671	1285 1287	1412 1419
Byte 0 Bit 0	Command Reject ├	—	—	—	—	—	→	①
1	Intervention Required ├	—	—	—	—	—	—	→
2	Busout Check ├	—	—	—	—	—	—	→
3	Equipment Check ├	—	—	—	—	—	—	→
4	Data Check ├	—	—	→	N/A	Data Check ├	—	→
5	Overrun ├	—	→	②	N/A ├	→	Overrun	N/A
6	Track Condition Check	N/A	③	N/A ├	—	→	Non-Recovery	N/A
7	Seek Check	Data Converter Check	N/A	Channel 9 Overflow	N/A	N/A	Keyboard Correction	Batch Numbering Switch Off
Byte 1 Bit 0	Data Check in Count	N/A ├	—	—	—	→	④	N/A
1	Track Overrun	N/A ├	—	—	—	—	—	→
2	End-of-Cylinder	N/A ├	—	—	—	—	—	→
3	N/A ├	—	—	—	—	—	—	→
4	No Record Found	At Load Point	N/A ├	—	—	→	Invalid Font ⑤	N/A
5	File Protected	N/A ├	—	—	—	—	—	→
6	Missing Address Marker	N/A ├	—	—	—	—	—	→
7	N/A	Tape Not Compatible	N/A ├	—	—	—	—	→

N/A = Not Applicable

① Command Reject or Disengage Failure.

② UCB Parity Check (1403 only).

③ Unusual Command Sequence (2540 read only).

④ Applies for 1287 to indicate tape (set to 1) or document (set to 0) mode.

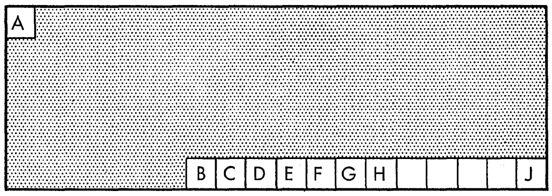
⑤ Applies for 1287 in the document mode only.

Figure 35. Sense Information for Devices Supported by Device Error Recovery

MESSAGE CODE (IN HEX)	10-CHARACTER MESSAGE	ERROR
08	C'INTERV REQ'	OPERATOR INTERVENTION REQUIRED
09	C'BUSOUT CHK'	BUS OUT CHECK
10	C'EQUIP CHK '	EQUIPMENT CHECK
11	C'DATA CHECK'	DATA CHECK
12	C'VERIFY CHK'	VERIFY CHECK
13	C'ADDR MRKER'	MISSING ADDRESS MARKER
14	C'OVERRUN '	OVERRUN
15	C'SEEK CHECK'	SEEK CHECK
16	C'DTA CHK CT'	DATA CHECK IN COUNT FIELD
17	C'FILE PROT '	VIOLATED FILE PROTECTION
18	C'COMM REJCT'	COMMAND REJECT
19	C'UNDETR ERR'	UNDETERMINED ERROR
20	C'ERR ON REC'	ERROR DURING RECOVERY ATTEMPT
21	C'NRF-MADDMK'	NO RECORD FOUND & MISSING ADDRESS MARKER
22	C'BALST CELL'	BALLAST CELL ACCESSED ON 2321
23	C'BLNK STRIP'	ACCESSED A PREVIOUSLY UNUSED STRIP
24	C'PROG CHECK'	I/O PROGRAM CHECK
25	C'PROT CHECK'	STORAGE PROTECTION CHECK
26	C'INVAL SEEK'	SEEK ADDRESS NOT VALID
27	C'UNKNWN DEV'	DEVICE IN ERROR NOT RECOGNIZED
28	C'CHAN DTCHK'	CHANNEL DATA CHECK
29	C'BK INTO LP'	BACKSPACE INTO LOADPOINT
30	C'CONVRT CHK'	TAPE CONVERT CHECK
31	C'DVC NOT OP'	DEVICE NOT OPERATIONAL
32	C'NON COMPAT'	NON-COMPATIBLE TAPE ON DRIVE
33	C'UCB PARITY'	PARITY ERROR IN UNIVERSAL CHARACTER BUFFER
34	C'BCH NM OFF'	BATCH NUMERING SWITCH OFF ON MICR
35	C'NON RECOV '	NON-RECOVERY ON 1285
36	C'NO REC FND'	NO RECORD FOUND
37	C'INVLD FONT' C'DISEN FAIL'	INVALID FONT ON 1287 IN DOCUMENT MODE DISENGAGE FAILURE ON MICR

Figure 36. Physical Transients Error Messages

504-Byte Physical Transient Area



The labels which are associated with these bytes are as designated below. Byte A is the first byte of the Physical Transient Area, Byte J is the last. Bytes B through H constitute the interphase communications area. When phases Z, Y and 0 are fetched or refetched, these bytes (B through H) are not overlaid and remain with information for the other phases.

Byte	Label	Phase
A	IJBPAR1	Z
	IJBPAR2	Y
	IJBPAR3	0
B	PARLTK	Z
	PARCOMM-1	
C	IJBPAR1+493	
	PARCOMM	Z
	IJBPAR2+493	
	PARCOMM2	Y
	PARCOMMC	Z
	PARCOMMD	Y
F	PARCOMMI	Z
	PARCOMMJ	Y
	PARCOMMC+3	
	PARCOMMD+3	
D, E, G, H	Addressed by incrementing or decrementing one of these labels.	

Note: Bytes C, D and E are used to indicate the program(s) F1, F2 or BG, to be canceled.

Bytes F, G and H indicate the programs which use devices which require operator intervention.

Byte B indicates if a canceled program has fetched a logical transient.

Figure 37. Interphase Communication Area (For A-Transients $$ANERRZ, Y, and 0)

I/O ERROR RECOVERY PROCEDURES AND SENSE DATA

2400 TAPE ERROR RECOVERY

CSW Bit 44--Channel Data Check
Action: Initial Selection--eight retries without repositioning. Read data transfer--no retries. Write data transfer--eight retries with repositioning. After stated number of retries, take equipment error exit (cancel).

Message: 0P28 CHAN DTCHK.

Byte 0, Bit 2--Bus Out Check
Action: If retry count is greater than seven (eight retries), take equipment error exit (cancel). If initial selection, take retry exit. Otherwise, perform repositioning and take retry exit.

Message: 0P09 BUSOUT CHK.

Byte 0, Bit 3--Equipment Check
Action: Take equipment error exit (cancel).

Message: 0P10 EQUIP CHK.

Byte 0, Bit 1--Intervention Required
Action: Check for Rewind and Unload (intervention required at device end). If yes, take continue exit; otherwise, take operator intervention exit.

Message: 0P08 INTERV REQ.

Byte 0, Bit 5--Overrun
Action: Allow eight retries, repositioning the tape. After eight retries, take equipment error exit (cancel).

Message: 0P14 OVERRUN.

Byte 0, Bit 4--Data Check
Action:

1. Read Commands--CCB option. If the record length is less than twelve and Byte 1, Bit 0 (noise) is off, take retry exit. Otherwise, retry 100 times with repositioning (back space/forward space) performing CRC correction. Perform tape cleaning every eight retries. Tape cleaning consists of five backspaces and four forward spaces. For a read backward, tape cleaning is done by five forward spaces and four backspaces. Detection of load-point causes termination of the backspacing sequence. After 100 retries, take equipment error exit (cancel, ignore).

2. Write and WTM Commands--Backspace erase and retry fifteen times, then take equipment error exit (cancel). For write commands, if unit exception is present in the CSW, post it to the CCB (Byte 4, Bit 7).

3. Erase Gap Commands--After fifteen retries, without repositioning, take equipment error exit (cancel).

Message: 0P11 DATA CHECK.

Byte 0, Bit 7--Data Converter Check
Action: Take equipment error exit (cancel).

Message: 0P30 CONVRT CHK.

Byte 0, Bit 0--Command Reject
Action: Take program check exit.

Message: 0P18 COMM REJCT.

Byte 1, Bit 4--Load Point
and
Byte 3, Bit 6--Backward Status
Action: Take program check exit.

Message: 0P29 BK INTO LP (Backward Command into Load Point).

Byte 1, Bit 7--Not Compatible
Action: Issue a rewind and unload command to the unit and then take operator intervention exit.

Message: 0P32 NOT COMPAT.

CSW Bit 47--Chaining Check
Action: Allow eight retries, repositioning the tape. After eight retries, take equipment error exit (cancel).

Message: 0P14 OVERRUN.

Notes: If an I/O error occurs during tape repositioning (other than backspace into load point on tape cleaning), equipment error exit (cancel) is taken with the message: 0P20 ERR ON REC (Error During Recovery).

To achieve data check error recovery on write tape mark and erase gap commands, they must be command-chained to a no-op because the command code is not available for analysis when the error occurs (device end).

1052 ERROR RECOVERY

CSW Bit 44--Channel Data Check
Action: One retry, equipment error exit (cancel, retry, ignore).

Message: 0P28 CHAN DTCHK.

Byte 0, Bit 3--Equipment Check
Action: One retry, equipment error exit (cancel, retry, ignore).

Message: 0P10 EQUIP CHK.

Byte 0, Bit 1--Intervention Required
Action: Execute audible alarm command and take operator intervention exit.

Message: 0P08 INTERV REQ.

Byte 0, Bit 2--Bus Out Check
Action: One retry, equipment error exit (cancel, retry, ignore).

Message: 0P09 BUSOUT CHK.

Byte 0, Bit 0--Command Reject
Action: Take program check exit.

Message: 0P18 COMM REJCT.

1403-1443 ERROR RECOVERY

CSW Bit 44--Channel Data Check
Action: If initial selection, one retry--take equipment error exit (initial selection: cancel, retry; channel end: cancel, retry, ignore).

Message: 0P28 CHAN DTCHK.

Byte 0, Bit 3--Equipment Check
Action: Take equipment error exit (cancel, ignore).

Message: 0P10 EQUIP CHK.

Byte 0, Bit 5--Code General Storage Parity Error (1403 only)
Action: Take equipment error exit (cancel). UCS buffer must be reloaded.

Message: 0P33 UCB PARITY.

Byte 0, Bit 1--Intervention Required
Action: Take operator intervention exit.

Message: 0P08 INTERV REQ.

Byte 0, Bit 2--Bus Out Check
Action: If initial selection, one retry; otherwise, take equipment error exit. (Initial selection: cancel, retry; channel end: cancel, retry, ignore).

Message: 0P09 BUSOUT CHK.

Byte 0, Bit 7--Channel 9
<u>Action</u>: Post CCB, take continue exit.

<u>Note</u>: This test is main storage resident.

Byte 0, Bit 0--Command Reject
<u>Action</u>: If command code is UCS enable or inhibit data check, take continue exit; otherwise, take program check exit. This procedure allows UCS-oriented programs to operate on non-UCS hardware.

<u>Message</u>: 0P18 COMM REJCT.

Byte 0, Bit 4--Data Check (1403 Only)
<u>Action</u>: Take equipment error exit (cancel, ignore).

<u>Message</u>: 0P11 DATA CHECK.

1442 ERROR RECOVERY

CSW Bit 44--Channel Data Check
<u>Action</u>: If initial selection, one retry; then equipment error exit (cancel, retry). If data transfer, take operator intervention exit.

<u>Message</u>: 0P28 CHAN DTCHK.

Byte 0, Bit 3--Equipment Check
<u>Action</u>: Take operator intervention exit.

<u>Message</u>: 0P10 EQUIP CHK.

Byte 0, Bit 1--Intervention Required
<u>Action</u>: Take operator intervention exit.

<u>Message</u>: 0P08 INTERV REQ.

Byte 0, Bit 2--Bus Out Check
<u>Action</u>: If initial selection, do one retry; then take equipment error exit (cancel, retry). If data transfer, take operator intervention exit.

<u>Message</u>: 0P09 BUSOUT CHK.

Byte 0, Bit 4--Data Check
<u>Action</u>: Take operator intervention exit.

<u>Message</u>: 0P11 DATA CHECK.

Byte 0, Bit 5--Overrun
<u>Action</u>: Take operator intervention exit.

<u>Message</u>: 0P14 OVERRUN.

Byte 0, Bit 0--Command Reject
<u>Action</u>: Take program check exit.

<u>Message</u>: 0P18 COMM REJCT.

CSW Bit 47--Chaining Check
<u>Action</u>: Take operator intervention exit.

<u>Message</u>: 0P14 OVERRUN.

2501, 2520, 2540 ERROR RECOVERY

CSW Bit 44--Channel Data Check
<u>Action</u>: If initial selection, one retry; then equipment error exit (cancel, retry). If read data transfer, take operator intervention exit. If punch data transfer, one retry; then equipment error exit (cancel, retry).

<u>Message</u>: 0P28 CHAN DTCHK.

Byte 0, Bit 3--Equipment Check
<u>Action</u>: Reader--Take operator intervention exit. Punch--CCB option. Take equipment error exit (cancel, ignore). For 2520, Byte 0, Bit 7 indicates punch check.

<u>Message</u>: 0P10 EQUIP CHK.

Byte 0, Bit 1--Intervention Required
<u>Action</u>: Take operator intervention exit.

<u>Message</u>: 0P08 INTERV REQ.

Byte 0, Bit 2--Bus Out Check
<u>Action</u>: Do one retry; then take equipment error exit (cancel, retry). If the device is a 2520, do not retry if this is not initial selection (cancel, retry).

<u>Message</u>: 0P09 BUSOUT CHK.

Byte 0, Bit 4--Data Check (Cannot occur on a 2520 punch)
<u>Action</u>: Take operator intervention exit.

<u>Message</u>: 0P11 DATA CHECK.

Byte 0, Bit 5--Overrun (Cannot occur on 2540 or 2520 punch)
<u>Action</u>: Take operator intervention exit.

<u>Message</u>: 0P14 OVERRUN.

Byte 0, Bit 0--Command Reject
Action: Take program check exit.

Message: 0P18 COMM REJCT.

Byte 0, Bit 6--Unusual Command Sequence (2540 read only)
Action: Post CCB--take continue exit.

CSW Bit 47--Chaining Check (2501, 2520 read only)
Action: Take operator intervention exit.

Message: 0P14 OVERRUN.

2671 ERROR RECOVERY

CSW Bit 44--Channel Data Check
Action: If initial selection, do one retry. Take equipment error exit (cancel).

Message: 0P28 CHAN DTCHK.

Byte 0, Bit 3--Equipment Check
Action: Test CCB for ignore option (byte 2, bit 4) and if on, turn on byte 3, bit 1 of the CCB and take equipment error exit (cancel, ignore, retry). Otherwise, take operator intervention exit.

Message: 0P10 EQUIP CHK.

Note: When an equipment check occurs, the operator must reposition the paper tape to the beginning of the record in error to perform the retry operation. The device must not be readied until this repositioning has been performed. If the ignore option is available to the operator, he can exercise this option by repositioning the tape to the beginning of the next record on the tape and then responding ignore on the 1052 keyboard. The ignore option is available to the operator whenever the user specifies any of the DTFPT ERROPT entry options.

Byte 0, Bit 1--Intervention Required
Action: Take operator intervention exit.

Message: 0P08 INTERV REQ.

Byte 0, Bit 2--Bus Out Check
Action: Do one retry; if error persists, take equipment error exit (cancel, retry).

Message: 0P09 BUSOUT CHK.

Byte 0, Bit 4--Data Check
Action: Test CCB for ignore option (byte 2, bit 4) and if on, turn on byte 3, bit 3 of the CCB and take equipment error exit (cancel, ignore, retry). Otherwise, take operator intervention exit.

Message: 0P11 DATA CHECK.

Notes: When a data check occurs, the user's CCW is modified by the error routine to allow rereading of the last character. The data address is the last character read (character in error), and the byte count is decreased by the number of valid characters read. If the CCB ignore option is chosen and the operator responds ignore, the I/O operation is dequeued and posted with the unrecoverable error bit on (CCB byte 2, bit 2) and 2671 data-check bit on (CCB byte 3, bit 3).

To read the rest of the record, the problem program (logical IOCS) should add one to the CCW data address and subtract one from the byte count to adjust for not rereading the bad character. It should then reissue the EXCP. The operator must backspace the tape two characters for retry (option retry or on the A-type message when ignore is not allowed). If the operator chooses the ignore option (the character in error is not to be reread), he must backspace the tape one character if the load key was pressed to free the tape or if the character preceding the character under the read head is an EOR (End-of-Record). Otherwise, no manual intervention is required for the ignore option. The ignore option is available to the operator whenever the user specifies any of the DTFPT ERROPT entry options.

Byte 0, Bit 0--Command Reject
Action: Take program check exit.

Message: 0P18 COMM REJCT.

Note: A record may not be partly on one tape and partly on another.

2311-2314 DASD ERROR RECOVERY

CSW Bit 44--Channel Data Check
Action: One retry; then equipment error exit (cancel, retry).

Message: 0P28 CHAN DTCHK.

Byte 0, Bit 3 - Equipment Check
Action: Take equipment error exit (cancel, retry).

Message: 0P10 EQUIP CHK.

Byte 1, Bit 4 - No Record Found
Action: Test for byte 1, bit 6 (Missing Address Marker). If present, execute restore command and take retry exit. After ten retries, take equipment error exit (cancel, retry). If not present, read Home Address and compare to user's Seek Address. If equal, post No Record Found to the CCB and take continue exit. If not equal, treat as a Seek Check.

Messages: 0P21 NRF - MADDMK (No Record Found/Missing Address Marker).
0P15 SEEK CHECK (Home Address unequal to Seek Address).

Note: Home Address is read, and the track address is provided for the error message. For other errors, the track address is obtained from the user seek address if error occurs during channel program execution.

Byte 0, Bit 7--Seek Check
Action: If byte 0, bit 0 (command reject) is on, take program check exit. Otherwise, execute restore command and take retry exit. After ten retries, take equipment error exit (cancel, retry).

Messages: 0P26 INVAL SEEK (Seek Check/Command Reject)
0P15 SEEK CHECK.

Byte 0, Bit 1--Intervention Required
Action: Take operator intervention exit.

Message: 0P08 INTERV REQ.

Byte 0, Bit 2--Bus Out Check
Action: If retry count is greater than nine, take equipment error exit (cancel, retry); otherwise, take retry exit.

Message: 0P09 BUSOUT CHK.

Byte 0, Bit 4 - Data Check
Action: CCB options (all data checks, data check on read or verify). If retry count is greater than nine, take equipment error exit (cancel, retry); otherwise, take retry exit. After nine retries, post data check on count to CCB, if present; otherwise, post data check. If command code is verify, post verify error to CCB.

Messages: 0P12 VERIFY CHK (Data Check on Verify Command).
0P11 DATA CHECK (Data Check/not Data Check on Count or Verify).
0P16 DTA CHK CT (Data Check on Count).

Note: Home Address is read, and the track address is provided for the error message. For other errors, the track address is obtained from the user seek address if error occurs during channel program execution.

Byte 0, Bit 5--Overrun
Action: If retry count is greater than nine, take equipment error exit (cancel, retry); otherwise, take retry exit.

Message: 0P14 OVERRUN.

Byte 1, Bit 6--Missing Address Markers
Action: If retry count is greater than nine, take equipment error exit (cancel, retry); otherwise, take retry exit.

Message: 0P13 ADDR MRKER.

Note: Home Address is read, and the track address is provided for the error message. For other errors, the track address is obtained from the user seek address if error occurs during channel program execution.

Byte 0, Bit 0 - Command Reject
Action: Check for Byte 1, Bit 5 (File Protect); in either case, take program check exit.

Messages: 0P18 COMM REJCT.
0P17 FILE PROT.

Byte 0, Bit 6--Track Condition Check
Action:
1. Read Home Address and R0 in the error recovery routine and move CCHH from R0 to Seek command executed below.

2. If alternate track: update seek address to the next track address. If the track address equals 10, treat it as End of Cylinder; otherwise, proceed to step 3.

3. Set up the channel program: Seek, Read Home Address (with skip bit on), TIC to CSW address minus eight. Execute this channel program in error recovery. At channel end, exit to channel scheduler CSW processing routine. If DASD file protection is present, set the appropriate file mask following Seek.

Byte 1, Bit 1--Track Overrun
Action: Post track overrun to the CCB and take continue exit.

Byte 1, Bit 2--End of Cylinder
Action: Post End of Cylinder to the CCB and take continue exit.

Byte 1, Bit 5--File Protect
Action: Take program check exit.

Message: 0P17 FILE PROT.

CSW Bit 47--Chaining Check
Action: If retry count is greater than nine, take equipment error exit (cancel, retry); otherwise, take retry exit.

Message: 0P14 OVERRUN.

Note: If the error routine gets an error while trying to execute a Restore command or Read Home Address or R0, equipment error exit is taken with retry and cancel options with the message: 0P20 ERR ON REC (Error During Recovery).

2321 DASD ERROR RECOVERY

CSW Bit 44--Channel Data Check
Action: One retry; then equipment error exit (cancel, retry).

Message: 0P28 CHAN DTCHK.

Byte 0, Bit 3--Equipment Check
Action: Take equipment error exit (cancel, retry).

Message: 0P10 EQUIP CHK.

Byte 1, Bit 4--No Record Found
Action:
1. If Byte 1, Bit 6 (missing Address Markers) is present, go to step 2. Otherwise, go to step 6.

2. If retry count is less than 3, issue a Restore command and go to step 5.

3. If retry count is equal to 3, issue a Read Home Address to the first and last tracks of the cylinder. If neither is successful (unit checks), take equipment error exit (cancel, retry). Otherwise, go to step 4.

4. If retry count is equal to 15, take equipment error exit (cancel, retry). Otherwise, go to step 5.

5. Increment retry count and take retry exit.

6. Issue a Read R0 and compare CCH to user's Seek Address. If equal, post No Record Found to the CCB and take continue exit. Otherwise, go to routine for Seek Check (alone).

Messages: 0P15 SEEK CHECK (No Record Found/R0 unequal to Seek Address).
0P23, BLNK STRIP (Step 3, cannot read Home Address).
0P21 NRF - MADDMK (Step 4, 15 retries).

Byte 0, Bit 7--Seek Check
Action: If Byte 0, Bit 0 (command reject) is present, take program check exit. If Byte 1, Bit 6 (missing Address Markers) is present, take operator intervention exit. Otherwise, issue a Seek to BB1111, a Seek to BB2222, and take retry exit. After ten retries, take equipment error exit (cancel, retry).

Messages: 0P26 INVAL SEEK (Seek Check/Command Reject).
0P22 BALST CELL (Seek Check/Missing Address Markers).
0P15 SEEK CHECK (Seek Check alone).

Byte 0, Bit 1--Intervention Required
Action: Take operator intervention exit.

Message: 0P08 INTERV REQ.

Byte 0, Bit 2--Bus Out Check
Action: Take retry exit. After 15 retries, take equipment error exit (cancel, retry).

Message: 0P09 BUSOUT CHK.

Byte 0, Bit 4--Data Check
Action:

1. If retry count is less than eight, go to step 5.

2. If retry count is equal to 226, take equipment error exit (cancel, retry).

3. If retry count is an even number, issue a Seek to X-X-X-4-19 (last track of strip) and a Seek to X-X-X-0-0 (first track of strip). Perform this operation eight times. Then proceed to step 4.

4. If retry count is any multiple of 32 (32, 64, 96, ...), issue a Seek to next lower strip. (If this is the lowest strip - 00000 - seek the next higher strip.) Proceed to step 5.

5. Increment retry count and take retry exit.

Messages: 0P11 DATA CHECK (Data Check/not Data Check on Count or Verify).
0P12 VERIFY CHK (Data Check on Verify Command).
0P16 DTA CHK CT (Data Check on Count).

Note: Home Address is read, and the track address is provided for the error message. For other conditions, the track address is obtained from the user's initial Seek address if the error occurs during channel program execution.

Byte 0, Bit 5--Overrun
Action: Take retry exit. After 15 retries, take equipment error exit (cancel, retry).

Message: 0P14 OVERRUN.

Byte 1, Bit 6--Missing Address Markers
Action: Perform action indicated under Data Check just described.

Message: 0P13 ADDR MRKER.

Note: Home Address is read, and the track address is provided for the error message. For other conditions, the track address is obtained from the user's initial Seek address if the error occurs during channel program execution.

Byte 0, Bit 0--Command Reject
Action: Check for byte 1, bit 5 (file protect); in either case, take program check exit.

Messages: 0P17 FILE PROT (Command Reject/File Protect).
0P18 COMM REJCT (Command Reject alone).

Byte 0, Bit 6--Track Condition Check
Action:

1. Read Home Address and R0 and move CCHH from R0 to Seek command executed below.

2. If alternate track: Update Seek Address to the next track address. If track address equals 20, treat it as End of Cylinder; otherwise, proceed to step 3.

3. Set up the channel program: Seek, Read Home Address (with skip bit on), TIC to CSW command address minus eight (last CCW executed). Execute this channel program in error recovery. At channel end, exit to channel scheduler CSW processing routine. If DASD file protection is present, set file mask (inhibit long Seeks) following the seek.

Byte 1, Bit 1--Track Overrun
Action: Post track overrun to the CCB and take continue exit.

Byte 1, Bit 2--End of Cylinder
Action: Post End of Cylinder to the CCB and take continue exit.

Byte 1, Bit 5--File Protect
Action: Take program check exit.

Message: 0P17 FILE PROT.

CSW Bit 47--Chaining Check
Action: Take retry exit. After 15 retries, take equipment error exit (cancel, retry).

Message: 0P14 OVERRUN.

Note: If the 2321 Error Routine gets an error while trying to execute a Restore command, a Seek command (data-check procedure), or a Read Home Address or a Read R0, equipment error exit is taken with retry and cancel options with the message: 0P20 ERR ON REC (Error During Recovery).

1285-1287 ERROR RECOVERY

CSW Bit 44--Channel Data Check
Action: One retry; then take equipment error exit (retry, cancel).

Message: 0P28 CHAN DTCHK.

Byte 0, Bit 3--Equipment Check
Action: Post byte 3 of CCB and then continue exit.

Note: Data Check and Equipment Check, which indicate unreadable character and unreadable line, respectively, are retried by Logical IOCS in an attempt to correct the error.

Byte 0, Bit 1--Intervention Required
Action: Test for byte 1, bit 6 (Non-recovery)--if present, post byte 3, bit 4 of the CCB. This indicates that the error is passed back to the problem program. Exit via equipment error.

Message: 0P35 NON RECOV. If byte 0, bit 6 is not present, take operator intervention exit.
0P08 INTERV REQ.

Byte 0, Bit 6--Nonrecovery
Action: Post byte 3, bit 4, of CCB and take continue exit.

Byte 0, Bit 2--Busout Check
Action: One retry; then equipment error exit (retry, cancel).

Message: 0P09 BUSOUT CHK.

Byte 0, Bit 4--Data Check
Action: Post byte 3, bit 0, of CCB and take continue exit.

Note: Data Check and Equipment Check, which indicate unreadable character and unreadable line, respectively, are retried by Logical IOCS in an attempt to correct the error.

Byte 0, Bit 5--Overrun
Action: Four retries; then equipment error exit (retry, cancel).

Message: 0P14 OVERRUN.

Byte 0, Bit 0--Command Reject
Action: Take program check exit.

Message: 0P18 COMM REJCT.

CSW Bit 47--Chaining Check
Action: Four retries; then equipment error exit (retry, cancel).

Message: 0P14 OVERRUN.

Byte 0, Bit 7--Keyboard Correction
Action: Post byte 3, bit 1, of CCB and take continue exit.

Byte 1, Bit 4--Invalid Font
Action: Take program check exit.

Message: 0P37 INVLD FONT.

Note: Byte 1, bit 4 applies only to the 1287 in document mode.

1412-1419 ERROR RECOVERY

CSW Bit 44--Channel Data Check
Action: Post unrecoverable I/O error to CCB byte 2, bit 2. Turn on passback bit in error queue entry byte 10, bit 2 for return to user for error recovery.

Message: 0P28 CHAN DTCHK.

Byte 0, Bit 0--Command Reject
Action: Check command code of CCW causing interrupt with X'E1' for 'Disengage Failed'. If 'Disengage Failed', post intervention required to CCB byte 3, bit 1; if not, post unrecoverable I/O error to CCB byte 2, bit 2. In either case, turn on passback in error queue entry of ERBLOC byte 10, bit 2, for return to user for error recovery.

Message: 0P18 COMM REJCT or 0P37 DISEN FAIL.

Byte 0, Bit 1--Intervention Required
Action: Post "Intervention Required" to the CCB (byte 3, bit 1), and take IGNORE exit. No message is printed.

Message: 0P08 INTERV REQ.

Note: The problem program should process all documents in the input buffer, note the Intervention Required and perform any print out necessary for operator recovery, and issue an Engage-Read to the device to continue processing documents. If the Intervention Required is due to a batch numbering update failure, the operator must update the batch number as part of manual recovery.

Byte 0, Bit 2--Bus Out Check

Action: Post unrecoverable I/O error to CCB byte 2, bit 2, and turn on passback bit in error queue entry of ERBLOC Byte 10, bit 2, for return to user for error recovery.

Message: 0P09 BUSOUT CHK.

Byte 0, Bit 3--Should not occur

Action: Post Unrecoverable I/O error to the CCB and provide informational message to the operator.

Message: 0P19 UNDETR ERR

Byte 0, Bit 7--Batch Numbering Switch Off

Action: Post document buffer byte 0, bits 0 and 1 and insert reject code X'CF' in byte 5. Turn off retry and turn on ignore bits in ERRFLG of error queue entry of ERBLOC.

Message: OP34 BCH NM OFF.

Note: CSW Bit 47 and the Sense bits 4, 5, 6, and 7 will not cause an I/O Interrupt. If CSW bit 44 or sense bit 1, 2, or 3 is not present for an I/O interrupt, the action and message for Sense bit 3 will be generated.

B-transient programs are infrequently-used routines; therefore, they are not resident in main storage and are fetched or loaded from the core image library when needed. The B-transients occupy an area of 1200 bytes, referred to as the Logical Transient Area (LTA).

An SVC 2 instruction loads and executes a B-transient phase. A prefix of $$B to the name of a phase identifies it as a B-transient. The normal return to supervisor nucleus control is an SVC 11, but some of the transient programs exit by fetching another B-transient with an SVC 2. In the latter case, the calling B-transient is overlaid by the transient being fetched.

Register 1 is loaded with the address of the transient name before the SVC 2 is issued. The fetch or load routine, then, has access to the name for searching the disk directories or tape records for the desired transient.

B-TRANSIENT GROUPING

The supervisor B-transient programs can be grouped by the various functions performed. These functions are: transient attention routine, program initiator, and program terminator.

TRANSIENT ATTENTION ROUTINES (CHARTS 10-12)

This group of B-Transients consists of $$BATTNA-$$BATTNH and $$BATTNN.

Attention commands are submitted when the operator presses the request key on the 1052 keyboard. The system's attention transient routine ($$BATTNA) is loaded, and issues the message READY FOR COMMUNICATIONS. It then reads input statement information and selects the appropriate statement processor. Commands accepted by the nonresident attention routines are:

- PAUSE: Indicates job control pauses for operator communication at the end of the current job step in the specified partition or, optionally, at end-of-job of the current program.
- CANCEL: Indicates one of the programs in the system is to be canceled. See Figure 38 for cancel code information.
- MAP: Provides a map of main-storage utilization. See Figure 39.
- ALLOC: Permits the operator to allocate storage among foreground and background programs.
- MSG: Causes control to be given to a foreground program operator communications routine previously activated by a STXIT command.
- TIMER: Causes interval timer support to be given to the program specified.
- START: Indicates the foreground initiation function has begun.
- BATCH: Initiates a dormant background or batched foreground area.
- LOG, NOLOG: Permits or suppresses logging of job control and single program statements on SYSLOG.
- IGNORE: Permits input from SYSRDR after a READ is issued.

Cancel-code in HEX	MSG-Code	Descriptive Part of Message
10		Normal EOJ
19	0P74	I/O Operator Option
1A	0P73	I/O Error
20	0S03 or 0S11	Program Check
21	0S04 or 0S09	Illegal SVC
22	0S05 or 0S06	Phase Not Found
23	0S02	Program Request
24	0S01	Operator Intervention
25	0P77	Invalid CCB-Address
26	0P71	Device Not Assigned
27	0P70	Undefined Logical Unit
30	0P72	Reading Past /& Statement
31	0P75	I/O Error Queue Overflow
32	0P76	Invalid DASD Address (Disk Only) Irrecoverable I/O Error (Tape Only)
33	0P79	No Long Seek (Disk Only)
FF	0P78	Unrecognized CANCEL Code

All cancel-codes except in connection with DUMP-macro (code = X'00' - not a true cancel-condition) initially have a value X'40' higher than indicated above, but the X'40' bit is stripped by the SUPVR before fetching the Terminator.

In addition to recognizing the cancel-codes above, the Terminator also recognizes the same codes with the X'80' bit on. The X'80' bit indicates that the cancellation occurred in a Logical Transient routine and it is tested for by $$BEOJ and subsequently reset.

Figure 38. Cancel Code Messages

SP		upper limit	
BG	size	upper limit	NAME
F2	size	upper limit	NAME
F1 T	size	upper limit	NAME
field 1	field 2	field 3	field 4

Field 1 - area identification

SP - supervisor
BG - background area
F2 - foreground area 2
F1 - foreground area 1
T - indicates which program has interval timer support.

Field 2 - length of area

The number of bytes allocated to the corresponding area of storage, where 1K equals 1024 bytes of storage.

Field 3 - area upper storage limit

The highest storage address allocated to the corresponding area in decimal.

Field 4 - user name

BG - background job name
F2 - foreground 2 program name
F1 - foreground 1 program name

Absence of a name indicates there is no active program in the area.

Figure 39. MAP Output

PROGRAM INITIATOR (CHARTS 10, 11, 13, AND 14)

This group of B-Transients consists of $$BATTNA, $$BATTNC, $$BATTNI-$$BATTNM, $$BATTNO, and $$BATTNP.

Single foreground programs are initiated by the operator through the 1052 assigned to SYSLOG. The operator may initiate a single program whenever an allocated foreground area does not contain a program.

The operator initiates a single program by pressing the 1052 request key. The attention interrupt causes control to be given to the system's Attention routine.

Note: If the transient area is in use by a routine other than the Attention routine, the attention interrupt is posted and serviced when the transient area becomes available.

The Attention routine reads a command from the operator. The command START (F1 or F2) indicates a single program is to be initiated. The Attention routine determines if the area specified is allocated and does not contain a program. If so, it transfers control to the single program initiator; otherwise, the operator is notified that an invalid command has been given.

The single program initiator reads subsequent commands required to initiate the program. These commands are used primarily to specify I/O assignments and label information. When an I/O assignment is attempted, the following verification is made:

1. The symbolic unit is a valid logical unit.

2. The symbolic unit is contained within the number specified for the area at system generation.

3. If the symbolic unit is to be assigned to a non-DASD, the device must not be in use by the other foreground program nor can it be assigned to a background job either as a standard, temporary, or alternate unit.

Figure 40 illustrates a LISTIO example.

The label information for each file in the job is written on SYSRES as a label information block for later retrieval and processing by the data management routines. A main storage area for label information is required under the same conditions as for background jobs, and is calculated and reserved by the initiator for self-relocating foreground programs. For non-self-relocating foreground programs, the label information area is determined by the LBLTYP statement.

When the EXEC statement is encountered, the initiator directs the supervisor to provide loading information for the program to be invoked. If the program has not been cataloged, the operator is notified by the initiator. He may correct the command (for example, if the name was misspelled) or cancel the initiation.

After the loading information is received, the initiator checks the load address to determine if it is zero, which indicates that a self-relocating program is to be loaded. The initiator sets up the load address so that the program will be loaded following the label information area. It also calculates the entry point to the program by adding the address at which it will be loaded to the previously-calculated entry point (derived when the program was linkage edited and cataloged onto the system). If a non-self-relocating program is loaded, the information used is that derived when the program was cataloged.

Diagnostics, such as the program being outside the limits of the foreground area, are not performed by the initiator, but are performed by the Supervisor when the program is loaded. The supervisor then causes the program to be terminated.

When initial control is given to the user's foreground program, register 2 contains the address of the uppermost byte of storage available to this program. This may be used to calculate the total storage available to the program. A foreground program can dynamically determine the storage available to it by storing the contents of this register for later reference.

Note that a program capable of either foreground or background operation (with proper linkage editing) can utilize the same programming to determine its storage allocation independently of its actual area assignment.

TERMINATOR (CHARTS 15-19)

A single program is terminated under its own control by issuing an EOJ, DUMP, or CANCEL macro or through operator action or a program error or certain I/O failures. When a single program is terminated, the following actions are taken:

1. All I/O operations that the program has requested are allowed to quiesce.

2. Tape error statistics for all tape drives assigned to the program being terminated, and on which an error has occurred, are logged out on SYSLOG. The statistics are then reset. This feature is a system generation option.

3. DASD extents used by this program for DASD file protection are dequeued. This feature is a system generation option.

4. All I/O assignments made for the program are canceled so that these devices will be available to subsequent programs. The assignments

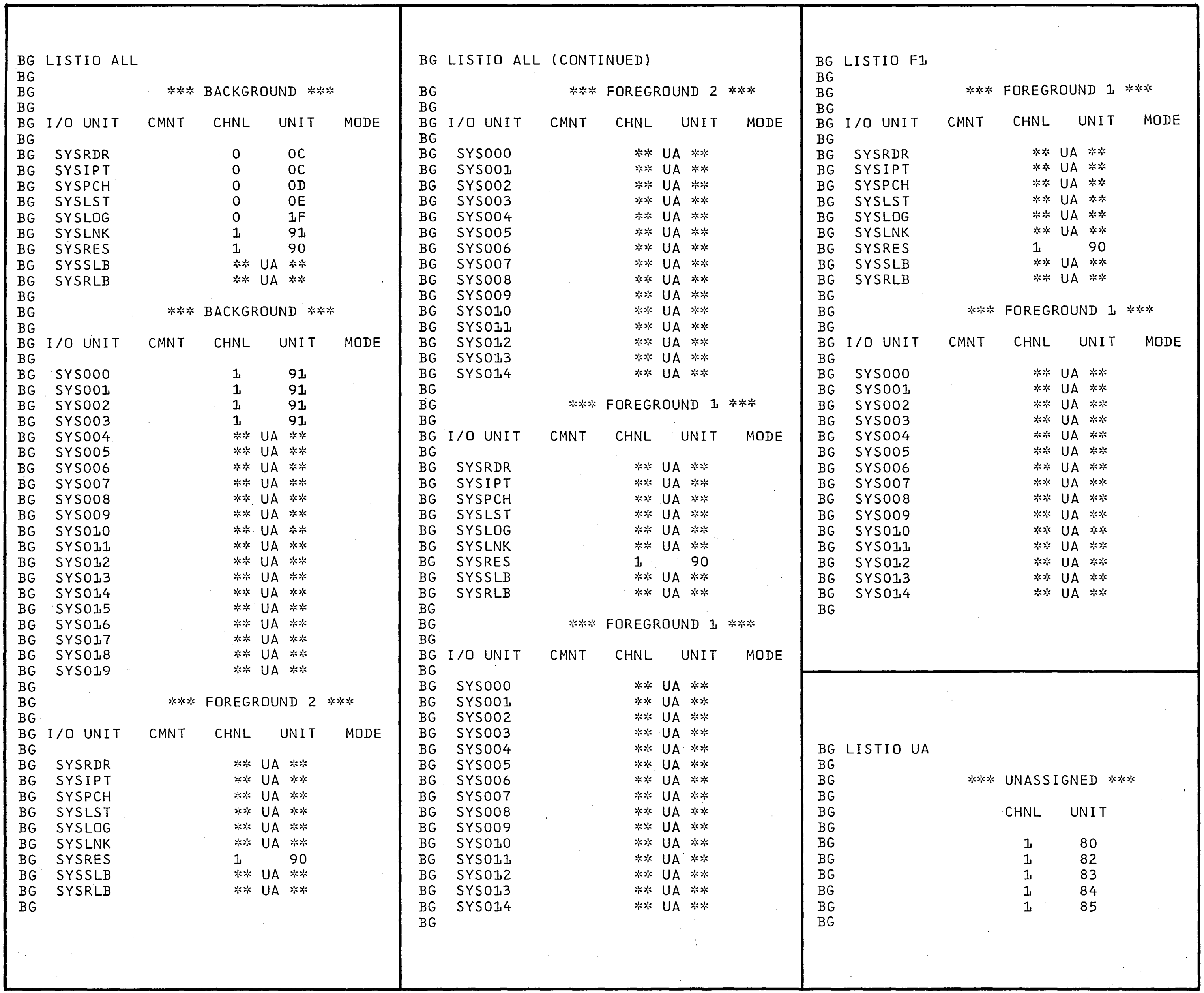
are not canceled if they are to be held across jobs by the HOLD command.

5. The operator is notified that the program is completed. The storage used by the program remains allocated for the foreground area.

6. The program is detached from the system's task selection mechanism.

See Figure 41 for an overall view of the terminator phases.

After a foreground program is completed, the operator may initiate another program for the area by pressing the SYSLOG request key and continuing with the initiation procedure previously described.



```
BG LISTIO ALL
BG
BG                *** BACKGROUND ***
BG
BG I/O UNIT    CMNT    CHNL    UNIT    MODE
BG
BG  SYSRDR               0      0C
BG  SYSIPT               0      0C
BG  SYSPCH               0      0D
BG  SYSLST               0      0E
BG  SYSLOG               0      1F
BG  SYSLNK               1      91
BG  SYSRES               1      90
BG  SYSSLB              ** UA **
BG  SYSRLB              ** UA **
BG
BG                *** BACKGROUND ***
BG
BG I/O UNIT    CMNT    CHNL    UNIT    MODE
BG
BG  SYS000               1      91
BG  SYS001               1      91
BG  SYS002               1      91
BG  SYS003               1      91
BG  SYS004              ** UA **
BG  SYS005              ** UA **
BG  SYS006              ** UA **
BG  SYS007              ** UA **
BG  SYS008              ** UA **
BG  SYS009              ** UA **
BG  SYS010              ** UA **
BG  SYS011              ** UA **
BG  SYS012              ** UA **
BG  SYS013              ** UA **
BG  SYS014              ** UA **
BG  SYS015              ** UA **
BG  SYS016              ** UA **
BG  SYS017              ** UA **
BG  SYS018              ** UA **
BG  SYS019              ** UA **
BG
BG                *** FOREGROUND 2 ***
BG
BG I/O UNIT    CMNT    CHNL    UNIT    MODE
BG
BG  SYSRDR              ** UA **
BG  SYSIPT              ** UA **
BG  SYSPCH              ** UA **
BG  SYSLST              ** UA **
BG  SYSLOG              ** UA **
BG  SYSLNK              ** UA **
BG  SYSRES              1      90
BG  SYSSLB              ** UA **
BG  SYSRLB              ** UA **
BG
```

```
BG LISTIO ALL (CONTINUED)
BG
BG                *** FOREGROUND 2 ***
BG
BG I/O UNIT    CMNT    CHNL    UNIT    MODE
BG
BG  SYS000              ** UA **
BG  SYS001              ** UA **
BG  SYS002              ** UA **
BG  SYS003              ** UA **
BG  SYS004              ** UA **
BG  SYS005              ** UA **
BG  SYS006              ** UA **
BG  SYS007              ** UA **
BG  SYS008              ** UA **
BG  SYS009              ** UA **
BG  SYS010              ** UA **
BG  SYS011              ** UA **
BG  SYS012              ** UA **
BG  SYS013              ** UA **
BG  SYS014              ** UA **
BG
BG                *** FOREGROUND 1 ***
BG
BG I/O UNIT    CMNT    CHNL    UNIT    MODE
BG
BG  SYSRDR              ** UA **
BG  SYSIPT              ** UA **
BG  SYSPCH              ** UA **
BG  SYSLST              ** UA **
BG  SYSLOG              ** UA **
BG  SYSLNK              ** UA **
BG  SYSRES              1      90
BG  SYSSLB              ** UA **
BG  SYSRLB              ** UA **
BG
BG                *** FOREGROUND 1 ***
BG
BG I/O UNIT    CMNT    CHNL    UNIT    MODE
BG
BG  SYS000              ** UA **
BG  SYS001              ** UA **
BG  SYS002              ** UA **
BG  SYS003              ** UA **
BG  SYS004              ** UA **
BG  SYS005              ** UA **
BG  SYS006              ** UA **
BG  SYS007              ** UA **
BG  SYS008              ** UA **
BG  SYS009              ** UA **
BG  SYS010              ** UA **
BG  SYS011              ** UA **
BG  SYS012              ** UA **
BG  SYS013              ** UA **
BG  SYS014              ** UA **
BG
```

```
BG LISTIO F1
BG
BG                *** FOREGROUND 1 ***
BG
BG I/O UNIT    CMNT    CHNL    UNIT    MODE
BG
BG  SYSRDR              ** UA **
BG  SYSIPT              ** UA **
BG  SYSPCH              ** UA **
BG  SYSLST              ** UA **
BG  SYSLOG              ** UA **
BG  SYSLNK              ** UA **
BG  SYSRES              1      90
BG  SYSSLB              ** UA **
BG  SYSRLB              ** UA **
BG
BG                *** FOREGROUND 1 ***
BG
BG I/O UNIT    CMNT    CHNL    UNIT    MODE
BG
BG  SYS000              ** UA **
BG  SYS001              ** UA **
BG  SYS002              ** UA **
BG  SYS003              ** UA **
BG  SYS004              ** UA **
BG  SYS005              ** UA **
BG  SYS006              ** UA **
BG  SYS007              ** UA **
BG  SYS008              ** UA **
BG  SYS009              ** UA **
BG  SYS010              ** UA **
BG  SYS011              ** UA **
BG  SYS012              ** UA **
BG  SYS013              ** UA **
BG  SYS014              ** UA **
BG
```

```
BG LISTIO UA
BG
BG                *** UNASSIGNED ***
BG
BG                    CHNL   UNIT
BG
BG                     1      80
BG                     1      82
BG                     1      83
BG                     1      84
BG                     1      85
BG
```

Figure 40. List I/O Examples

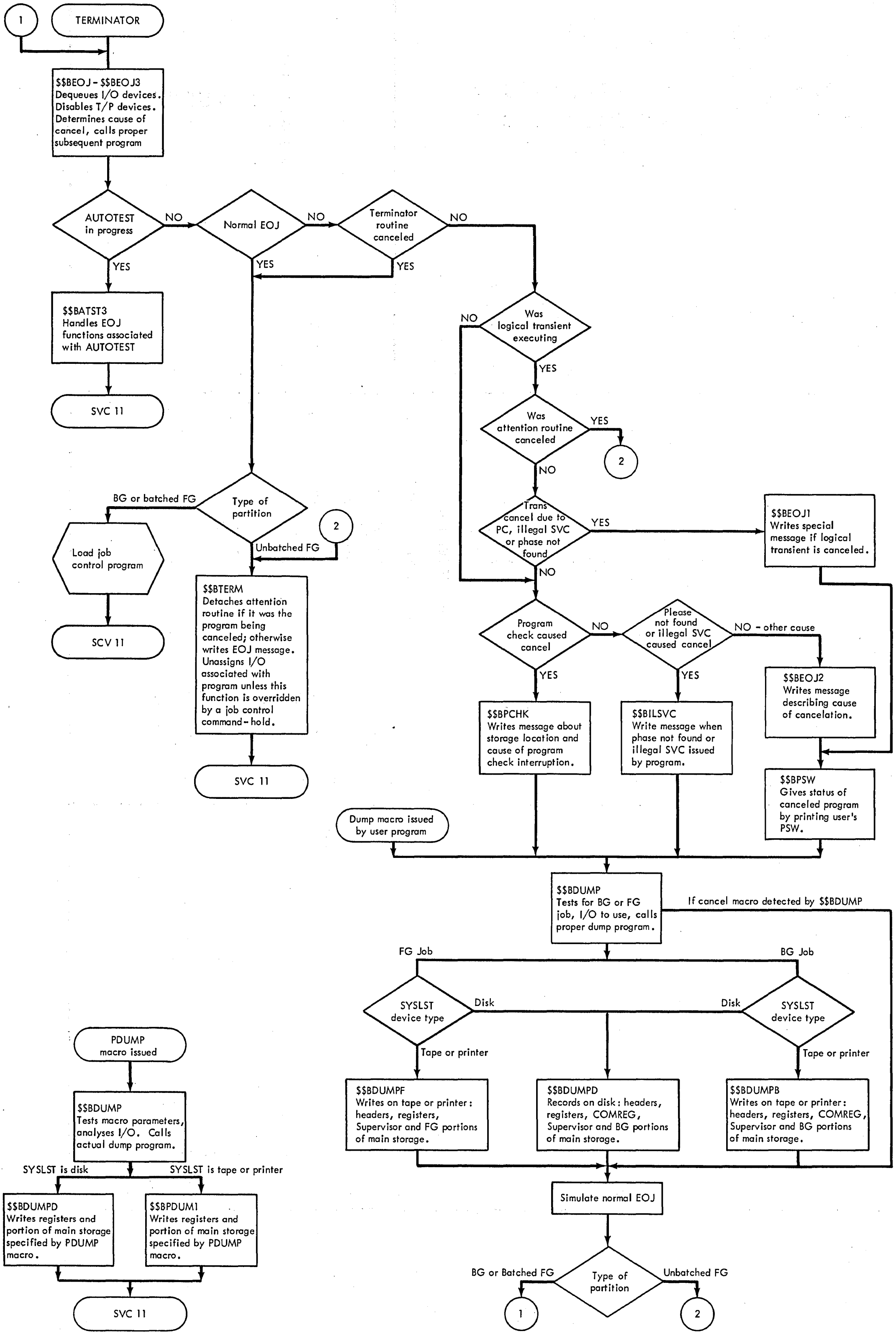
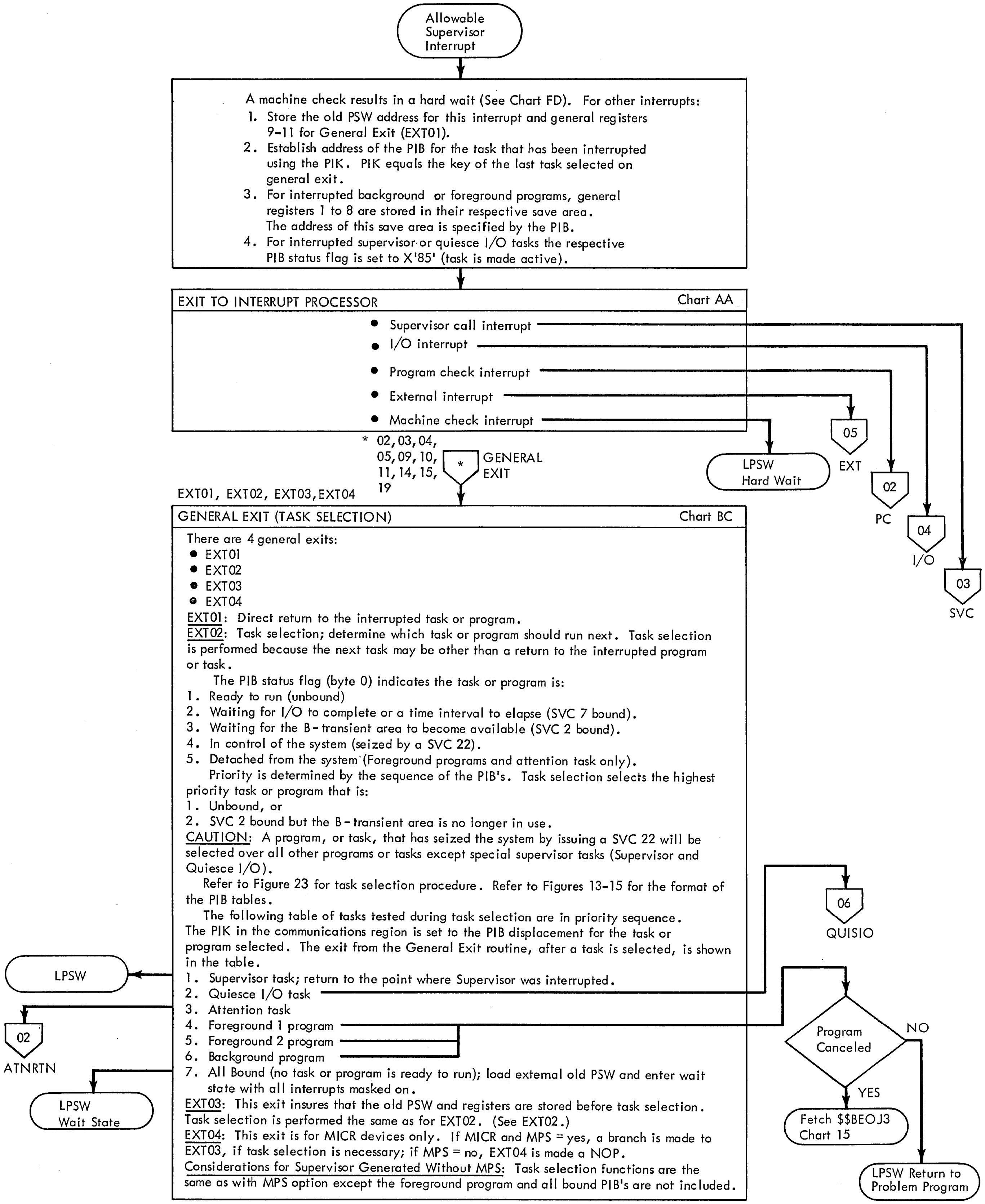

1
TERMINATOR
$$BEOJ - $$BEOJ3 Dequeues I/O devices. Disables T/P devices. Determines cause of cancel, calls proper subsequent program
AUTOTEST in progress
NO
YES
$$BATST3 Handles EOJ functions associated with AUTOTEST
SVC 11
Normal EOJ
NO
YES
Terminator routine canceled
NO
YES
Type of partition
BG or batched FG
Unbatched FG
2
Load job control program
SCV 11
$$BTERM Detaches attention routine if it was the program being canceled; otherwise writes EOJ message. Unassigns I/O associated with program unless this function is overridden by a job control command-hold.
SVC 11
Was logical transient executing
NO
YES
Was attention routine canceled
YES
2
NO
Trans cancel due to PC, illegal SVC or phase not found
YES
NO
$$BEOJ1 Writes special message if logical transient is canceled.
Program check caused cancel
NO
YES
Please not found or illegal SVC caused cancel
NO - other cause
YES
$$BEOJ2 Writes message describing cause of cancelation.
$$BPCHK Writes message about storage location and cause of program check interruption.
$$BILSVC Write message when phase not found or illegal SVC issued by program.
$$BPSW Gives status of canceled program by printing user's PSW.
Dump macro issued by user program
$$BDUMP Tests for BG or FG job, I/O to use, calls proper dump program.
If cancel macro detected by $$BDUMP
FG Job
BG Job
SYSLST device type
Disk
Tape or printer
Disk
SYSLST device type
Tape or printer
$$BDUMPF Writes on tape or printer: headers, registers, Supervisor and FG portions of main storage.
$$BDUMPD Records on disk: headers, registers, COMREG, Supervisor and BG portions of main storage.
$$BDUMPB Writes on tape or printer: headers, registers, COMREG, Supervisor and BG portions of main storage.
Simulate normal EOJ
Type of partition
BG or Batched FG
Unbatched FG
1
2
PDUMP macro issued
$$BDUMP Tests macro parameters, analyses I/O. Calls actual dump program.
SYSLST is disk
SYSLST is tape or printer
$$BDUMPD Writes registers and portion of main storage specified by PDUMP macro.
$$BPDUM1 Writes registers and portion of main storage specified by PDUMP macro.
SVC 11

Figure 41. Terminator Phases

Chart 01. Supervisor General Entry, General Exit, and Processor Exit



A machine check results in a hard wait (See Chart FD). For other interrupts:

1. Store the old PSW address for this interrupt and general registers 9-11 for General Exit (EXT01).
2. Establish address of the PIB for the task that has been interrupted using the PIK. PIK equals the key of the last task selected on general exit.
3. For interrupted background or foreground programs, general registers 1 to 8 are stored in their respective save area. The address of this save area is specified by the PIB.
4. For interrupted supervisor or quiesce I/O tasks the respective PIB status flag is set to X'85' (task is made active).

EXIT TO INTERRUPT PROCESSOR Chart AA

- Supervisor call interrupt
- I/O interrupt
- Program check interrupt
- External interrupt
- Machine check interrupt

GENERAL EXIT (TASK SELECTION) Chart BC

There are 4 general exits:

- EXT01
- EXT02
- EXT03
- EXT04

EXT01: Direct return to the interrupted task or program.

EXT02: Task selection; determine which task or program should run next. Task selection is performed because the next task may be other than a return to the interrupted program or task.

The PIB status flag (byte 0) indicates the task or program is:

1. Ready to run (unbound)
2. Waiting for I/O to complete or a time interval to elapse (SVC 7 bound).
3. Waiting for the B-transient area to become available (SVC 2 bound).
4. In control of the system (seized by a SVC 22).
5. Detached from the system (Foreground programs and attention task only).

Priority is determined by the sequence of the PIB's. Task selection selects the highest priority task or program that is:

1. Unbound, or
2. SVC 2 bound but the B-transient area is no longer in use.

CAUTION: A program, or task, that has seized the system by issuing a SVC 22 will be selected over all other programs or tasks except special supervisor tasks (Supervisor and Quiesce I/O).

Refer to Figure 23 for task selection procedure. Refer to Figures 13-15 for the format of the PIB tables.

The following table of tasks tested during task selection are in priority sequence. The PIK in the communications region is set to the PIB displacement for the task or program selected. The exit from the General Exit routine, after a task is selected, is shown in the table.

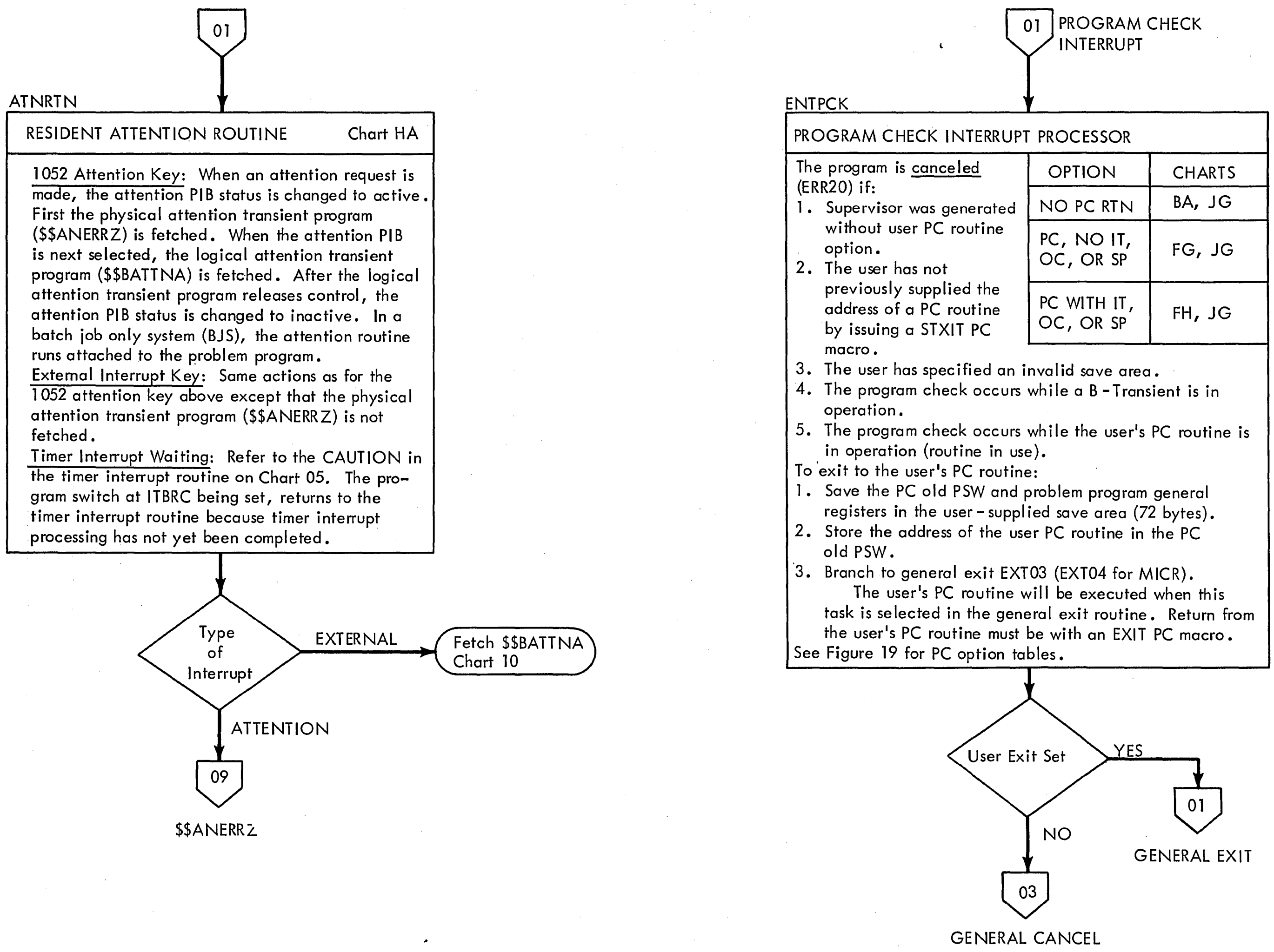
1. Supervisor task; return to the point where Supervisor was interrupted.
2. Quiesce I/O task
3. Attention task
4. Foreground 1 program
5. Foreground 2 program
6. Background program
7. All Bound (no task or program is ready to run); load external old PSW and enter wait state with all interrupts masked on.

EXT03: This exit insures that the old PSW and registers are stored before task selection. Task selection is performed the same as for EXT02. (See EXT02.)

EXT04: This exit is for MICR devices only. If MICR and MPS = yes, a branch is made to EXT03, if task selection is necessary; if MPS = no, EXT04 is made a NOP.

Considerations for Supervisor Generated Without MPS: Task selection functions are the same as with MPS option except the foreground program and all bound PIB's are not included.

Chart 02. Resident Attention Routine and Program Check Interrupt Routine



RESIDENT ATTENTION ROUTINE — Chart HA

1052 Attention Key: When an attention request is made, the attention PIB status is changed to active. First the physical attention transient program ($$ANERRZ) is fetched. When the attention PIB is next selected, the logical attention transient program ($$BATTNA) is fetched. After the logical attention transient program releases control, the attention PIB status is changed to inactive. In a batch job only system (BJS), the attention routine runs attached to the problem program.

External Interrupt Key: Same actions as for the 1052 attention key above except that the physical attention transient program ($$ANERRZ) is not fetched.

Timer Interrupt Waiting: Refer to the CAUTION in the timer interrupt routine on Chart 05. The program switch at ITBRC being set, returns to the timer interrupt routine because timer interrupt processing has not yet been completed.

PROGRAM CHECK INTERRUPT PROCESSOR

The program is canceled (ERR20) if:

1. Supervisor was generated without user PC routine option.
2. The user has not previously supplied the address of a PC routine by issuing a STXIT PC macro.
3. The user has specified an invalid save area.
4. The program check occurs while a B-Transient is in operation.
5. The program check occurs while the user's PC routine is in operation (routine in use).

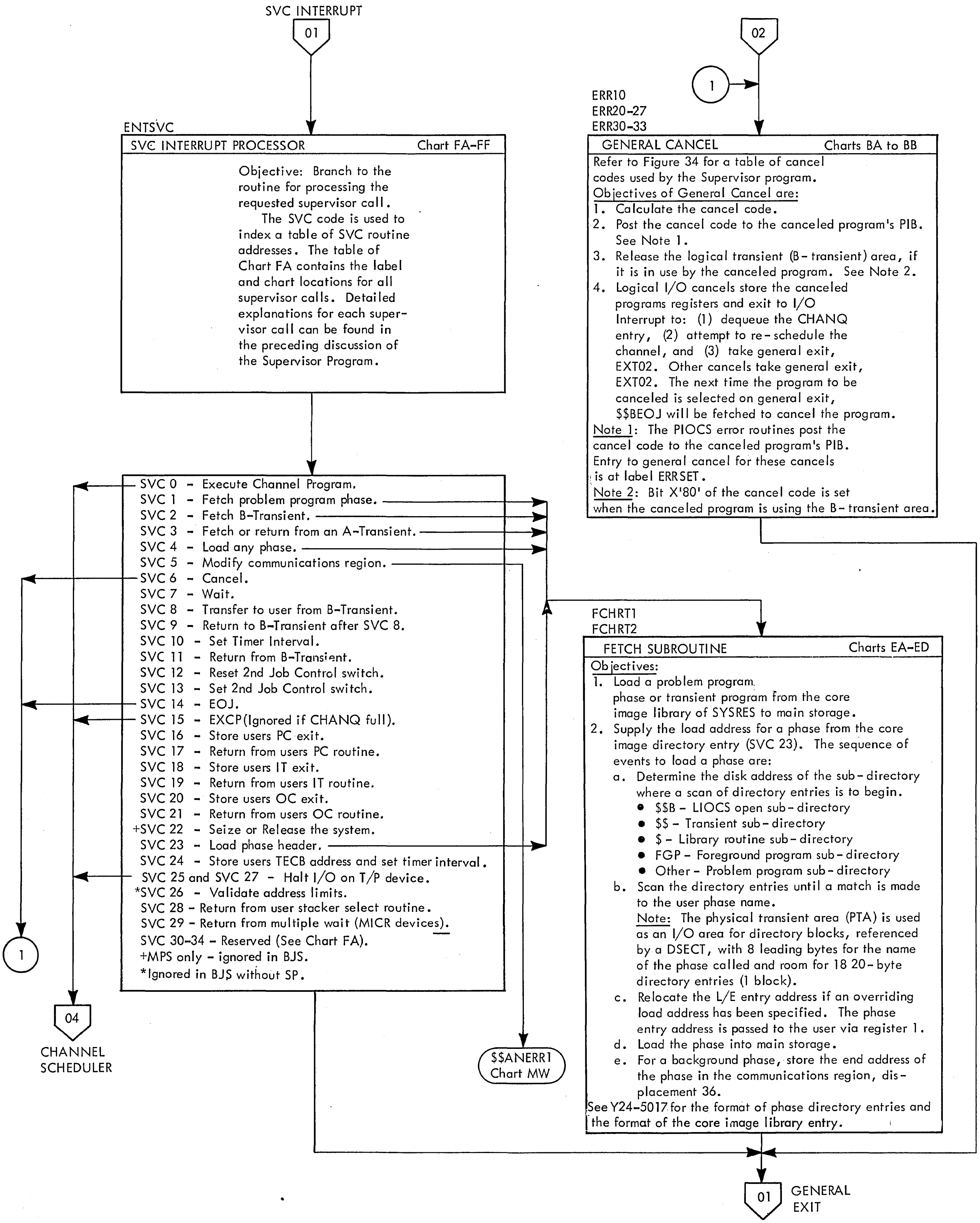
OPTION	CHARTS
NO PC RTN	BA, JG
PC, NO IT, OC, OR SP	FG, JG
PC WITH IT, OC, OR SP	FH, JG

To exit to the user's PC routine:

1. Save the PC old PSW and problem program general registers in the user-supplied save area (72 bytes).
2. Store the address of the user PC routine in the PC old PSW.
3. Branch to general exit EXT03 (EXT04 for MICR). The user's PC routine will be executed when this task is selected in the general exit routine. Return from the user's PC routine must be with an EXIT PC macro.

See Figure 19 for PC option tables.

Chart 03. SVC Interrupt Processor, General Cancel, and Fetch



SVC INTERRUPT PROCESSOR Chart FA-FF

Objective: Branch to the routine for processing the requested supervisor call.

The SVC code is used to index a table of SVC routine addresses. The table of Chart FA contains the label and chart locations for all supervisor calls. Detailed explanations for each supervisor call can be found in the preceding discussion of the Supervisor Program.

GENERAL CANCEL Charts BA to BB

Refer to Figure 34 for a table of cancel codes used by the Supervisor program.

Objectives of General Cancel are:

1. Calculate the cancel code.
2. Post the cancel code to the canceled program's PIB. See Note 1.
3. Release the logical transient (B-transient) area, if it is in use by the canceled program. See Note 2.
4. Logical I/O cancels store the canceled programs registers and exit to I/O Interrupt to: (1) dequeue the CHANQ entry, (2) attempt to re-schedule the channel, and (3) take general exit, EXT02. Other cancels take general exit, EXT02. The next time the program to be canceled is selected on general exit, $$BEOJ will be fetched to cancel the program.

Note 1: The PIOCS error routines post the cancel code to the canceled program's PIB. Entry to general cancel for these cancels is at label ERRSET.

Note 2: Bit X'80' of the cancel code is set when the canceled program is using the B-transient area.

SVC 0 - Execute Channel Program.
SVC 1 - Fetch problem program phase.
SVC 2 - Fetch B-Transient.
SVC 3 - Fetch or return from an A-Transient.
SVC 4 - Load any phase.
SVC 5 - Modify communications region.
SVC 6 - Cancel.
SVC 7 - Wait.
SVC 8 - Transfer to user from B-Transient.
SVC 9 - Return to B-Transient after SVC 8.
SVC 10 - Set Timer Interval.
SVC 11 - Return from B-Transient.
SVC 12 - Reset 2nd Job Control switch.
SVC 13 - Set 2nd Job Control switch.
SVC 14 - EOJ.
SVC 15 - EXCP(Ignored if CHANQ full).
SVC 16 - Store users PC exit.
SVC 17 - Return from users PC routine.
SVC 18 - Store users IT exit.
SVC 19 - Return from users IT routine.
SVC 20 - Store users OC exit.
SVC 21 - Return from users OC routine.
+SVC 22 - Seize or Release the system.
SVC 23 - Load phase header.
SVC 24 - Store users TECB address and set timer interval.
SVC 25 and SVC 27 - Halt I/O on T/P device.
*SVC 26 - Validate address limits.
SVC 28 - Return from user stacker select routine.
SVC 29 - Return from multiple wait (MICR devices).
SVC 30-34 - Reserved (See Chart FA).
+MPS only - ignored in BJS.
*Ignored in BJS without SP.

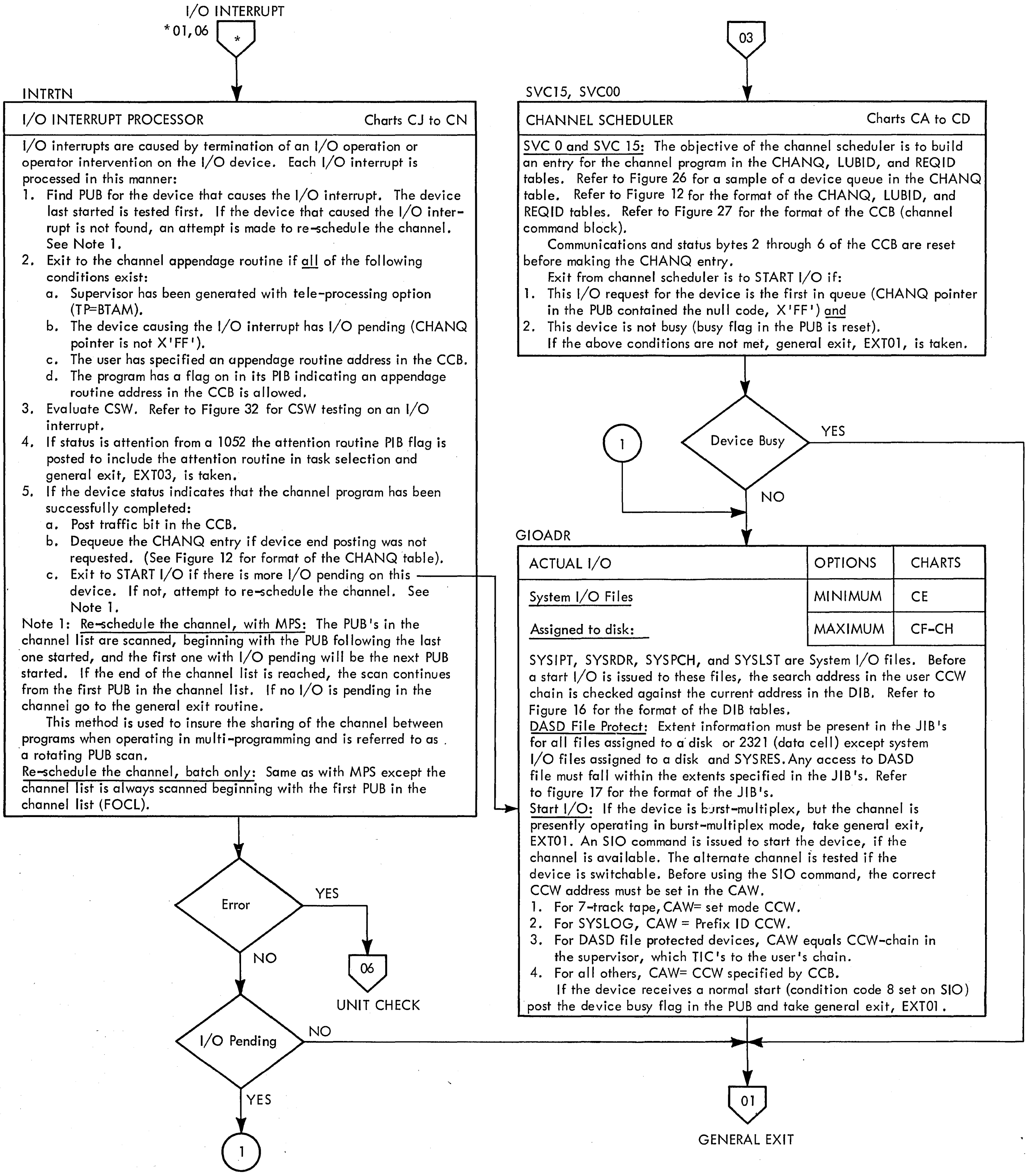
FETCH SUBROUTINE Charts EA-ED

Objectives:

1. Load a problem program phase or transient program from the core image library of SYSRES to main storage.
2. Supply the load address for a phase from the core image directory entry (SVC 23). The sequence of events to load a phase are:
   a. Determine the disk address of the sub-directory where a scan of directory entries is to begin.
      - $$B - LIOCS open sub-directory
      - $$ - Transient sub-directory
      - $ - Library routine sub-directory
      - FGP - Foreground program sub-directory
      - Other - Problem program sub-directory
   b. Scan the directory entries until a match is made to the user phase name.
      Note: The physical transient area (PTA) is used as an I/O area for directory blocks, referenced by a DSECT, with 8 leading bytes for the name of the phase called and room for 18 20-byte directory entries (1 block).
   c. Relocate the L/E entry address if an overriding load address has been specified. The phase entry address is passed to the user via register 1.
   d. Load the phase into main storage.
   e. For a background phase, store the end address of the phase in the communications region, displacement 36.

See Y24-5017 for the format of phase directory entries and the format of the core image library entry.

Chart 04. I/O Interrupt Processor and Channel Scheduler



I/O INTERRUPT PROCESSOR — Charts CJ to CN

I/O interrupts are caused by termination of an I/O operation or operator intervention on the I/O device. Each I/O interrupt is processed in this manner:

1. Find PUB for the device that causes the I/O interrupt. The device last started is tested first. If the device that caused the I/O interrupt is not found, an attempt is made to re-schedule the channel. See Note 1.
2. Exit to the channel appendage routine if <u>all</u> of the following conditions exist:
   a. Supervisor has been generated with tele-processing option (TP=BTAM).
   b. The device causing the I/O interrupt has I/O pending (CHANQ pointer is not X'FF').
   c. The user has specified an appendage routine address in the CCB.
   d. The program has a flag on in its PIB indicating an appendage routine address in the CCB is allowed.
3. Evaluate CSW. Refer to Figure 32 for CSW testing on an I/O interrupt.
4. If status is attention from a 1052 the attention routine PIB flag is posted to include the attention routine in task selection and general exit, EXT03, is taken.
5. If the device status indicates that the channel program has been successfully completed:
   a. Post traffic bit in the CCB.
   b. Dequeue the CHANQ entry if device end posting was not requested. (See Figure 12 for format of the CHANQ table).
   c. Exit to START I/O if there is more I/O pending on this device. If not, attempt to re-schedule the channel. See Note 1.

Note 1: <u>Re-schedule the channel, with MPS:</u> The PUB's in the channel list are scanned, beginning with the PUB following the last one started, and the first one with I/O pending will be the next PUB started. If the end of the channel list is reached, the scan continues from the first PUB in the channel list. If no I/O is pending in the channel go to the general exit routine.

This method is used to insure the sharing of the channel between programs when operating in multi-programming and is referred to as a rotating PUB scan.

<u>Re-schedule the channel, batch only:</u> Same as with MPS except the channel list is always scanned beginning with the first PUB in the channel list (FOCL).

CHANNEL SCHEDULER — Charts CA to CD

<u>SVC 0 and SVC 15:</u> The objective of the channel scheduler is to build an entry for the channel program in the CHANQ, LUBID, and REQID tables. Refer to Figure 26 for a sample of a device queue in the CHANQ table. Refer to Figure 12 for the format of the CHANQ, LUBID, and REQID tables. Refer to Figure 27 for the format of the CCB (channel command block).

Communications and status bytes 2 through 6 of the CCB are reset before making the CHANQ entry.

Exit from channel scheduler is to START I/O if:

1. This I/O request for the device is the first in queue (CHANQ pointer in the PUB contained the null code, X'FF') <u>and</u>
2. This device is not busy (busy flag in the PUB is reset).

If the above conditions are not met, general exit, EXT01, is taken.

ACTUAL I/O	OPTIONS	CHARTS
<u>System I/O Files</u>	MINIMUM	CE
<u>Assigned to disk:</u>	MAXIMUM	CF-CH

SYSIPT, SYSRDR, SYSPCH, and SYSLST are System I/O files. Before a start I/O is issued to these files, the search address in the user CCW chain is checked against the current address in the DIB. Refer to Figure 16 for the format of the DIB tables.

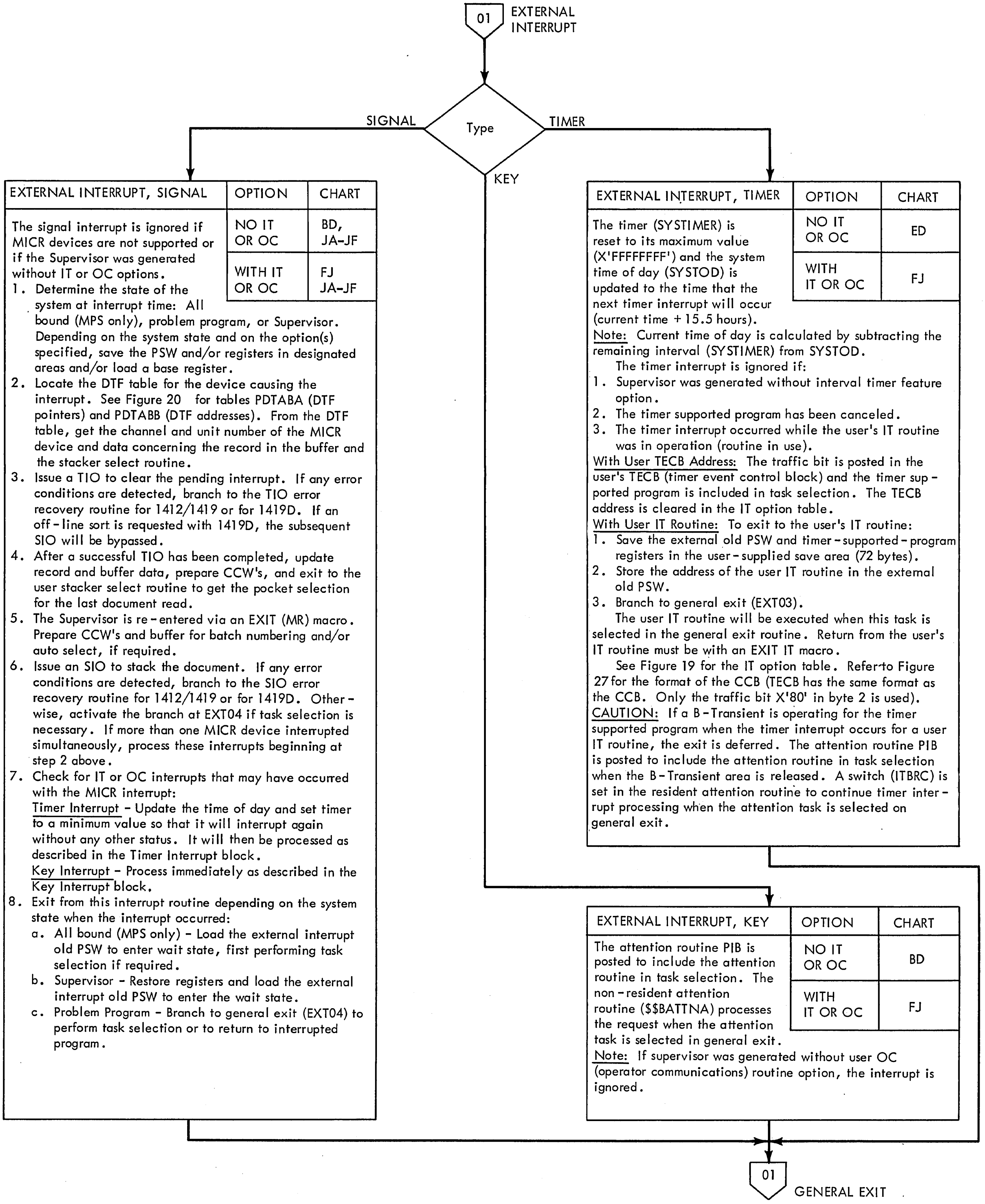
<u>DASD File Protect:</u> Extent information must be present in the JIB's for all files assigned to a disk or 2321 (data cell) except system I/O files assigned to a disk and SYSRES. Any access to DASD file must fall within the extents specified in the JIB's. Refer to figure 17 for the format of the JIB's.

<u>Start I/O:</u> If the device is burst-multiplex, but the channel is presently operating in burst-multiplex mode, take general exit, EXT01. An SIO command is issued to start the device, if the channel is available. The alternate channel is tested if the device is switchable. Before using the SIO command, the correct CCW address must be set in the CAW.

1. For 7-track tape, CAW= set mode CCW.
2. For SYSLOG, CAW = Prefix ID CCW.
3. For DASD file protected devices, CAW equals CCW-chain in the supervisor, which TIC's to the user's chain.
4. For all others, CAW= CCW specified by CCB.

If the device receives a normal start (condition code 8 set on SIO) post the device busy flag in the PUB and take general exit, EXT01.

Chart 05. External Interrupt Routines



EXTERNAL INTERRUPT, SIGNAL	OPTION	CHART
	NO IT OR OC	BD, JA-JF
	WITH IT OR OC	FJ JA-JF

The signal interrupt is ignored if MICR devices are not supported or if the Supervisor was generated without IT or OC options.

1. Determine the state of the system at interrupt time: All bound (MPS only), problem program, or Supervisor. Depending on the system state and on the option(s) specified, save the PSW and/or registers in designated areas and/or load a base register.
2. Locate the DTF table for the device causing the interrupt. See Figure 20 for tables PDTABA (DTF pointers) and PDTABB (DTF addresses). From the DTF table, get the channel and unit number of the MICR device and data concerning the record in the buffer and the stacker select routine.
3. Issue a TIO to clear the pending interrupt. If any error conditions are detected, branch to the TIO error recovery routine for 1412/1419 or for 1419D. If an off-line sort is requested with 1419D, the subsequent SIO will be bypassed.
4. After a successful TIO has been completed, update record and buffer data, prepare CCW's, and exit to the user stacker select routine to get the pocket selection for the last document read.
5. The Supervisor is re-entered via an EXIT (MR) macro. Prepare CCW's and buffer for batch numbering and/or auto select, if required.
6. Issue an SIO to stack the document. If any error conditions are detected, branch to the SIO error recovery routine for 1412/1419 or for 1419D. Otherwise, activate the branch at EXT04 if task selection is necessary. If more than one MICR device interrupted simultaneously, process these interrupts beginning at step 2 above.
7. Check for IT or OC interrupts that may have occurred with the MICR interrupt:
   Timer Interrupt - Update the time of day and set timer to a minimum value so that it will interrupt again without any other status. It will then be processed as described in the Timer Interrupt block.
   Key Interrupt - Process immediately as described in the Key Interrupt block.
8. Exit from this interrupt routine depending on the system state when the interrupt occurred:
   a. All bound (MPS only) - Load the external interrupt old PSW to enter wait state, first performing task selection if required.
   b. Supervisor - Restore registers and load the external interrupt old PSW to enter the wait state.
   c. Problem Program - Branch to general exit (EXT04) to perform task selection or to return to interrupted program.

EXTERNAL INTERRUPT, TIMER	OPTION	CHART
	NO IT OR OC	ED
	WITH IT OR OC	FJ

The timer (SYSTIMER) is reset to its maximum value (X'FFFFFFFF') and the system time of day (SYSTOD) is updated to the time that the next timer interrupt will occur (current time + 15.5 hours).

Note: Current time of day is calculated by subtracting the remaining interval (SYSTIMER) from SYSTOD.

The timer interrupt is ignored if:

1. Supervisor was generated without interval timer feature option.
2. The timer supported program has been canceled.
3. The timer interrupt occurred while the user's IT routine was in operation (routine in use).

With User TECB Address: The traffic bit is posted in the user's TECB (timer event control block) and the timer supported program is included in task selection. The TECB address is cleared in the IT option table.

With User IT Routine: To exit to the user's IT routine:

1. Save the external old PSW and timer-supported-program registers in the user-supplied save area (72 bytes).
2. Store the address of the user IT routine in the external old PSW.
3. Branch to general exit (EXT03).

The user IT routine will be executed when this task is selected in the general exit routine. Return from the user's IT routine must be with an EXIT IT macro.

See Figure 19 for the IT option table. Refer to Figure 27 for the format of the CCB (TECB has the same format as the CCB. Only the traffic bit X'80' in byte 2 is used).

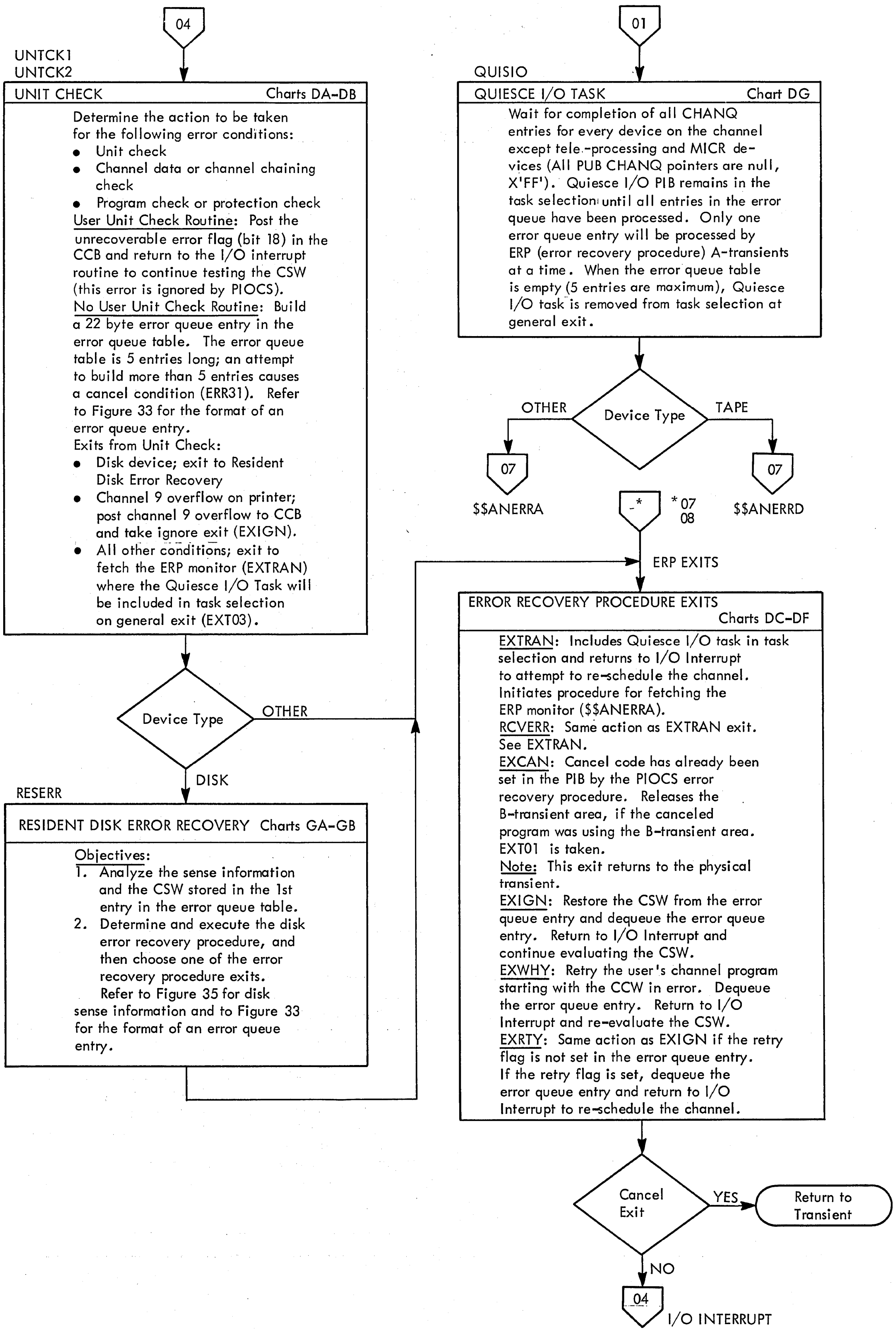
CAUTION: If a B-Transient is operating for the timer supported program when the timer interrupt occurs for a user IT routine, the exit is deferred. The attention routine PIB is posted to include the attention routine in task selection when the B-Transient area is released. A switch (ITBRC) is set in the resident attention routine to continue timer interrupt processing when the attention task is selected on general exit.

EXTERNAL INTERRUPT, KEY	OPTION	CHART
	NO IT OR OC	BD
	WITH IT OR OC	FJ

The attention routine PIB is posted to include the attention routine in task selection. The non-resident attention routine ($$BATTNA) processes the request when the attention task is selected in general exit.

Note: If supervisor was generated without user OC (operator communications) routine option, the interrupt is ignored.

Chart 06. Unit Check, Resident ERP, and Quiesce I/O Routines



UNIT CHECK

Determine the action to be taken for the following error conditions:

- Unit check
- Channel data or channel chaining check
- Program check or protection check

User Unit Check Routine: Post the unrecoverable error flag (bit 18) in the CCB and return to the I/O interrupt routine to continue testing the CSW (this error is ignored by PIOCS).
No User Unit Check Routine: Build a 22 byte error queue entry in the error queue table. The error queue table is 5 entries long; an attempt to build more than 5 entries causes a cancel condition (ERR31). Refer to Figure 33 for the format of an error queue entry.
Exits from Unit Check:

- Disk device; exit to Resident Disk Error Recovery
- Channel 9 overflow on printer; post channel 9 overflow to CCB and take ignore exit (EXIGN).
- All other conditions; exit to fetch the ERP monitor (EXTRAN) where the Quiesce I/O Task will be included in task selection on general exit (EXT03).

QUIESCE I/O TASK

Wait for completion of all CHANQ entries for every device on the channel except tele-processing and MICR devices (All PUB CHANQ pointers are null, X'FF'). Quiesce I/O PIB remains in the task selection until all entries in the error queue have been processed. Only one error queue entry will be processed by ERP (error recovery procedure) A-transients at a time. When the error queue table is empty (5 entries are maximum), Quiesce I/O task is removed from task selection at general exit.

ERROR RECOVERY PROCEDURE EXITS

EXTRAN: Includes Quiesce I/O task in task selection and returns to I/O Interrupt to attempt to re-schedule the channel. Initiates procedure for fetching the ERP monitor ($$ANERRA).
RCVERR: Same action as EXTRAN exit. See EXTRAN.
EXCAN: Cancel code has already been set in the PIB by the PIOCS error recovery procedure. Releases the B-transient area, if the canceled program was using the B-transient area. EXT01 is taken.
Note: This exit returns to the physical transient.
EXIGN: Restore the CSW from the error queue entry and dequeue the error queue entry. Return to I/O Interrupt and continue evaluating the CSW.
EXWHY: Retry the user's channel program starting with the CCW in error. Dequeue the error queue entry. Return to I/O Interrupt and re-evaluate the CSW.
EXRTY: Same action as EXIGN if the retry flag is not set in the error queue entry. If the retry flag is set, dequeue the error queue entry and return to I/O Interrupt to re-schedule the channel.

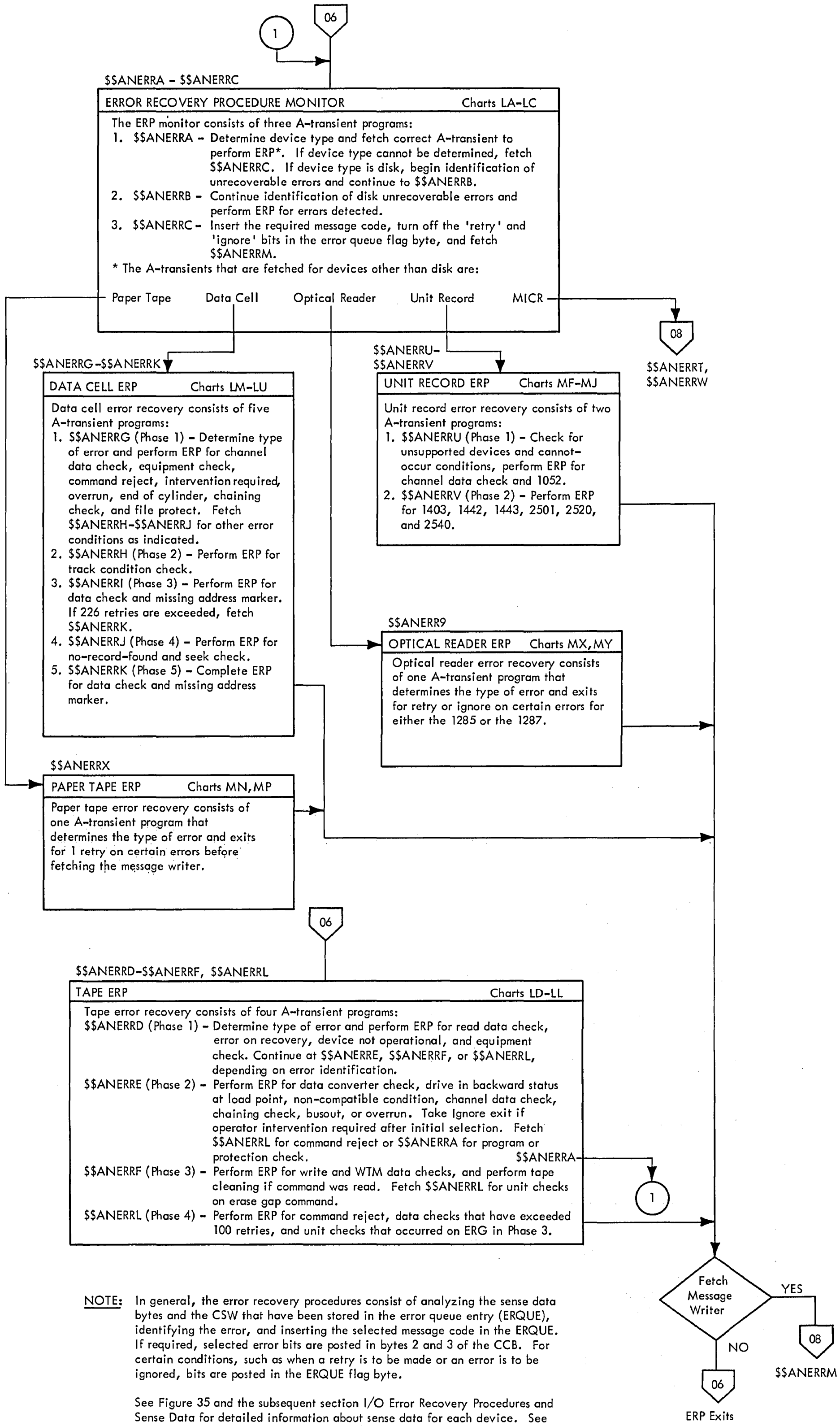
RESIDENT DISK ERROR RECOVERY

Objectives:

1. Analyze the sense information and the CSW stored in the 1st entry in the error queue table.
2. Determine and execute the disk error recovery procedure, and then choose one of the error recovery procedure exits.

Refer to Figure 35 for disk sense information and to Figure 33 for the format of an error queue entry.

Chart 07. Physical Transients (Part 1 of 2)



$$ANERRA - $$ANERRC

ERROR RECOVERY PROCEDURE MONITOR Charts LA-LC

The ERP monitor consists of three A-transient programs:

1. $$ANERRA - Determine device type and fetch correct A-transient to perform ERP*. If device type cannot be determined, fetch $$ANERRC. If device type is disk, begin identification of unrecoverable errors and continue to $$ANERRB.
2. $$ANERRB - Continue identification of disk unrecoverable errors and perform ERP for errors detected.
3. $$ANERRC - Insert the required message code, turn off the 'retry' and 'ignore' bits in the error queue flag byte, and fetch $$ANERRM.

* The A-transients that are fetched for devices other than disk are:

Paper Tape Data Cell Optical Reader Unit Record MICR

$$ANERRG-$$ANERRK

DATA CELL ERP Charts LM-LU

Data cell error recovery consists of five A-transient programs:

1. $$ANERRG (Phase 1) - Determine type of error and perform ERP for channel data check, equipment check, command reject, intervention required, overrun, end of cylinder, chaining check, and file protect. Fetch $$ANERRH-$$ANERRJ for other error conditions as indicated.
2. $$ANERRH (Phase 2) - Perform ERP for track condition check.
3. $$ANERRI (Phase 3) - Perform ERP for data check and missing address marker. If 226 retries are exceeded, fetch $$ANERRK.
4. $$ANERRJ (Phase 4) - Perform ERP for no-record-found and seek check.
5. $$ANERRK (Phase 5) - Complete ERP for data check and missing address marker.

$$ANERRU-$$ANERRV

UNIT RECORD ERP Charts MF-MJ

Unit record error recovery consists of two A-transient programs:

1. $$ANERRU (Phase 1) - Check for unsupported devices and cannot-occur conditions, perform ERP for channel data check and 1052.
2. $$ANERRV (Phase 2) - Perform ERP for 1403, 1442, 1443, 2501, 2520, and 2540.

$$ANERR9

OPTICAL READER ERP Charts MX,MY

Optical reader error recovery consists of one A-transient program that determines the type of error and exits for retry or ignore on certain errors for either the 1285 or the 1287.

$$ANERRX

PAPER TAPE ERP Charts MN,MP

Paper tape error recovery consists of one A-transient program that determines the type of error and exits for 1 retry on certain errors before fetching the message writer.

$$ANERRD-$$ANERRF, $$ANERRL

TAPE ERP Charts LD-LL

Tape error recovery consists of four A-transient programs:

$$ANERRD (Phase 1) - Determine type of error and perform ERP for read data check, error on recovery, device not operational, and equipment check. Continue at $$ANERRE, $$ANERRF, or $$ANERRL, depending on error identification.

$$ANERRE (Phase 2) - Perform ERP for data converter check, drive in backward status at load point, non-compatible condition, channel data check, chaining check, busout, or overrun. Take Ignore exit if operator intervention required after initial selection. Fetch $$ANERRL for command reject or $$ANERRA for program or protection check. $$ANERRA

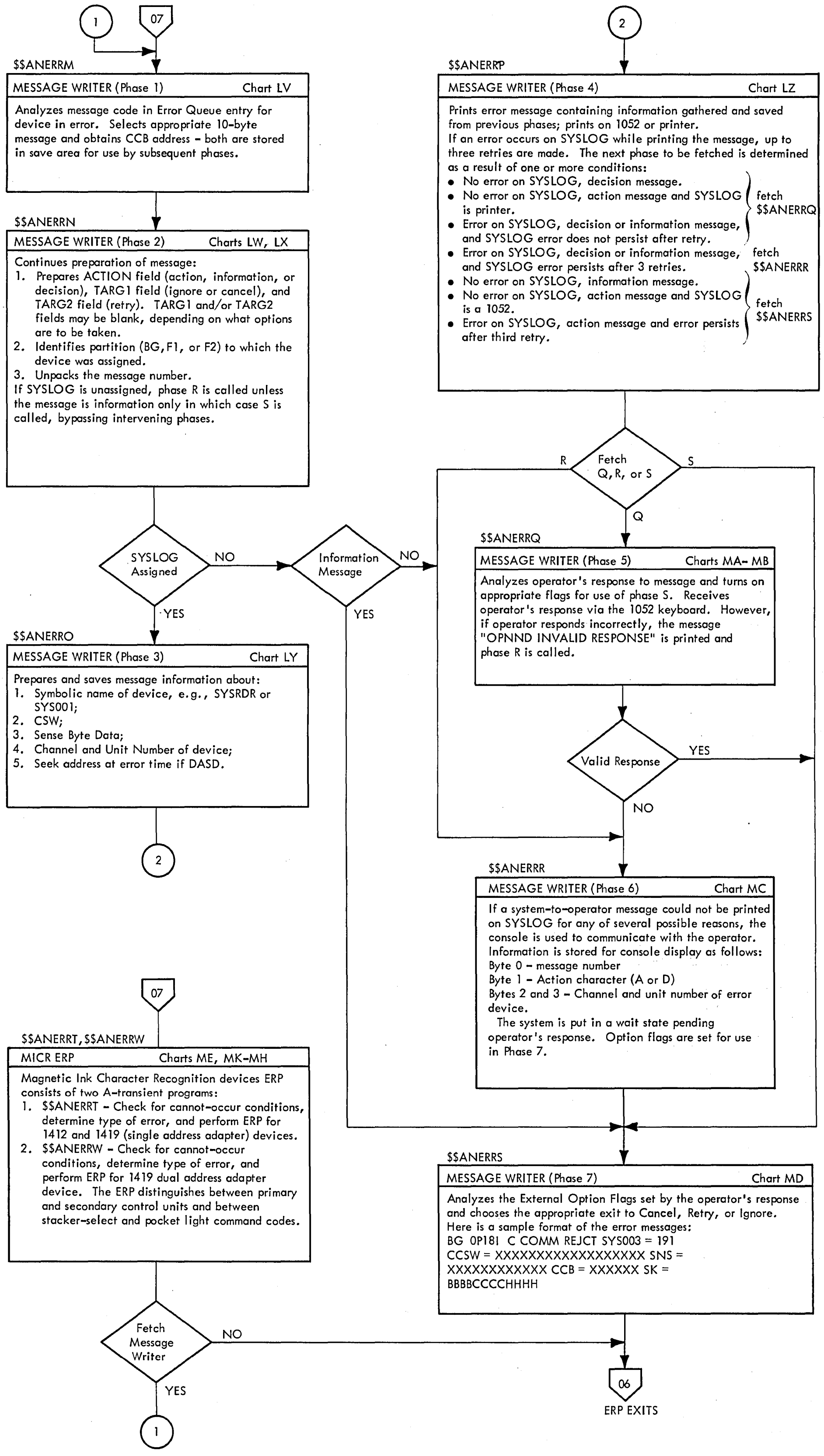
$$ANERRF (Phase 3) - Perform ERP for write and WTM data checks, and perform tape cleaning if command was read. Fetch $$ANERRL for unit checks on erase gap command.

$$ANERRL (Phase 4) - Perform ERP for command reject, data checks that have exceeded 100 retries, and unit checks that occurred on ERG in Phase 3.

NOTE: In general, the error recovery procedures consist of analyzing the sense data bytes and the CSW that have been stored in the error queue entry (ERQUE), identifying the error, and inserting the selected message code in the ERQUE. If required, selected error bits are posted in bytes 2 and 3 of the CCB. For certain conditions, such as when a retry is to be made or an error is to be ignored, bits are posted in the ERQUE flag byte.

See Figure 35 and the subsequent section I/O Error Recovery Procedures and Sense Data for detailed information about sense data for each device. See Figure 33 for an illustration of the error recovery block (ERBLOC), including the error queue entry (ERQUE). Figure 36 is a listing of the hexadecimal message codes with the error message for each code. See Figures 31 and 27 for the condition and error bits in the CSW and CCB, respectively.

Chart 08. Physical Transients (Part 2 of 2)



$$ANERRM

MESSAGE WRITER (Phase 1) Chart LV

Analyzes message code in Error Queue entry for device in error. Selects appropriate 10-byte message and obtains CCB address - both are stored in save area for use by subsequent phases.

$$ANERRN

MESSAGE WRITER (Phase 2) Charts LW, LX

Continues preparation of message:

1. Prepares ACTION field (action, information, or decision), TARG1 field (ignore or cancel), and TARG2 field (retry). TARG1 and/or TARG2 fields may be blank, depending on what options are to be taken.
2. Identifies partition (BG, F1, or F2) to which the device was assigned.
3. Unpacks the message number.

If SYSLOG is unassigned, phase R is called unless the message is information only in which case S is called, bypassing intervening phases.

$$ANERRO

MESSAGE WRITER (Phase 3) Chart LY

Prepares and saves message information about:

1. Symbolic name of device, e.g., SYSRDR or SYS001;
2. CSW;
3. Sense Byte Data;
4. Channel and Unit Number of device;
5. Seek address at error time if DASD.

$$ANERRT, $$ANERRW

MICR ERP Charts ME, MK-MH

Magnetic Ink Character Recognition devices ERP consists of two A-transient programs:

1. $$ANERRT - Check for cannot-occur conditions, determine type of error, and perform ERP for 1412 and 1419 (single address adapter) devices.
2. $$ANERRW - Check for cannot-occur conditions, determine type of error, and perform ERP for 1419 dual address adapter device. The ERP distinguishes between primary and secondary control units and between stacker-select and pocket light command codes.

$$ANERRP

MESSAGE WRITER (Phase 4) Chart LZ

Prints error message containing information gathered and saved from previous phases; prints on 1052 or printer.

If an error occurs on SYSLOG while printing the message, up to three retries are made. The next phase to be fetched is determined as a result of one or more conditions:

- No error on SYSLOG, decision message.
- No error on SYSLOG, action message and SYSLOG is printer.
- Error on SYSLOG, decision or information message, and SYSLOG error does not persist after retry.

} fetch $$ANERRQ

- Error on SYSLOG, decision or information message, and SYSLOG error persists after 3 retries.

fetch $$ANERRR

- No error on SYSLOG, information message.
- No error on SYSLOG, action message and SYSLOG is a 1052.
- Error on SYSLOG, action message and error persists after third retry.

} fetch $$ANERRS

$$ANERRQ

MESSAGE WRITER (Phase 5) Charts MA- MB

Analyzes operator's response to message and turns on appropriate flags for use of phase S. Receives operator's response via the 1052 keyboard. However, if operator responds incorrectly, the message "OPNND INVALID RESPONSE" is printed and phase R is called.

$$ANERRR

MESSAGE WRITER (Phase 6) Chart MC

If a system-to-operator message could not be printed on SYSLOG for any of several possible reasons, the console is used to communicate with the operator. Information is stored for console display as follows:

Byte 0 - message number

Byte 1 - Action character (A or D)

Bytes 2 and 3 - Channel and unit number of error device.

The system is put in a wait state pending operator's response. Option flags are set for use in Phase 7.

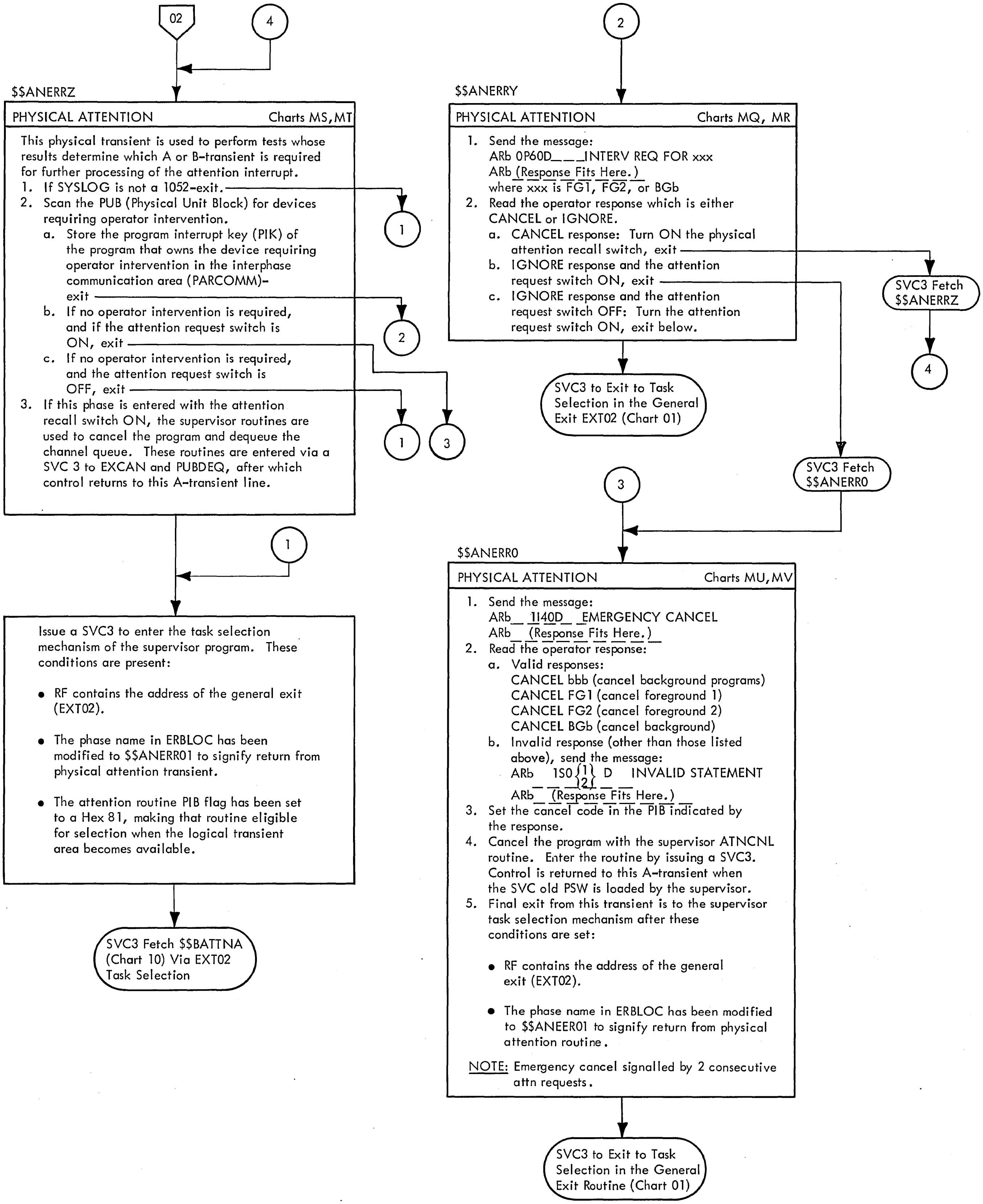
$$ANERRS

MESSAGE WRITER (Phase 7) Chart MD

Analyzes the External Option Flags set by the operator's response and chooses the appropriate exit to Cancel, Retry, or Ignore. Here is a sample format of the error messages:

BG 0P181 C COMM REJCT SYS003 = 191
CCSW = XXXXXXXXXXXXXXXXXX SNS =
XXXXXXXXXXXX CCB = XXXXXX SK =
BBBBCCCCHHHH

Chart 09. Physical Attention Transients


02
4
2
1
2
1
3
3
1
4
SVC3 Fetch $$ANERRZ
SVC3 Fetch $$ANERRO
SVC3 to Exit to Task Selection in the General Exit EXT02 (Chart 01)
SVC3 Fetch $$BATTNA (Chart 10) Via EXT02 Task Selection
SVC3 to Exit to Task Selection in the General Exit Routine (Chart 01)

$$ANERRZ

PHYSICAL ATTENTION Charts MS,MT

This physical transient is used to perform tests whose results determine which A or B-transient is required for further processing of the attention interrupt.

1. If SYSLOG is not a 1052-exit.
2. Scan the PUB (Physical Unit Block) for devices requiring operator intervention.
   a. Store the program interrupt key (PIK) of the program that owns the device requiring operator intervention in the interphase communication area (PARCOMM)-exit
   b. If no operator intervention is required, and if the attention request switch is ON, exit
   c. If no operator intervention is required, and the attention request switch is OFF, exit
3. If this phase is entered with the attention recall switch ON, the supervisor routines are used to cancel the program and dequeue the channel queue. These routines are entered via a SVC 3 to EXCAN and PUBDEQ, after which control returns to this A-transient line.

Issue a SVC3 to enter the task selection mechanism of the supervisor program. These conditions are present:

- RF contains the address of the general exit (EXT02).
- The phase name in ERBLOC has been modified to $$ANERR01 to signify return from physical attention transient.
- The attention routine PIB flag has been set to a Hex 81, making that routine eligible for selection when the logical transient area becomes available.

$$ANERRY

PHYSICAL ATTENTION Charts MQ, MR

1. Send the message:
   ARb 0P60D___INTERV REQ FOR xxx
   ARb (Response Fits Here.)
   where xxx is FG1, FG2, or BGb
2. Read the operator response which is either CANCEL or IGNORE.
   a. CANCEL response: Turn ON the physical attention recall switch, exit
   b. IGNORE response and the attention request switch ON, exit
   c. IGNORE response and the attention request switch OFF: Turn the attention request switch ON, exit below.

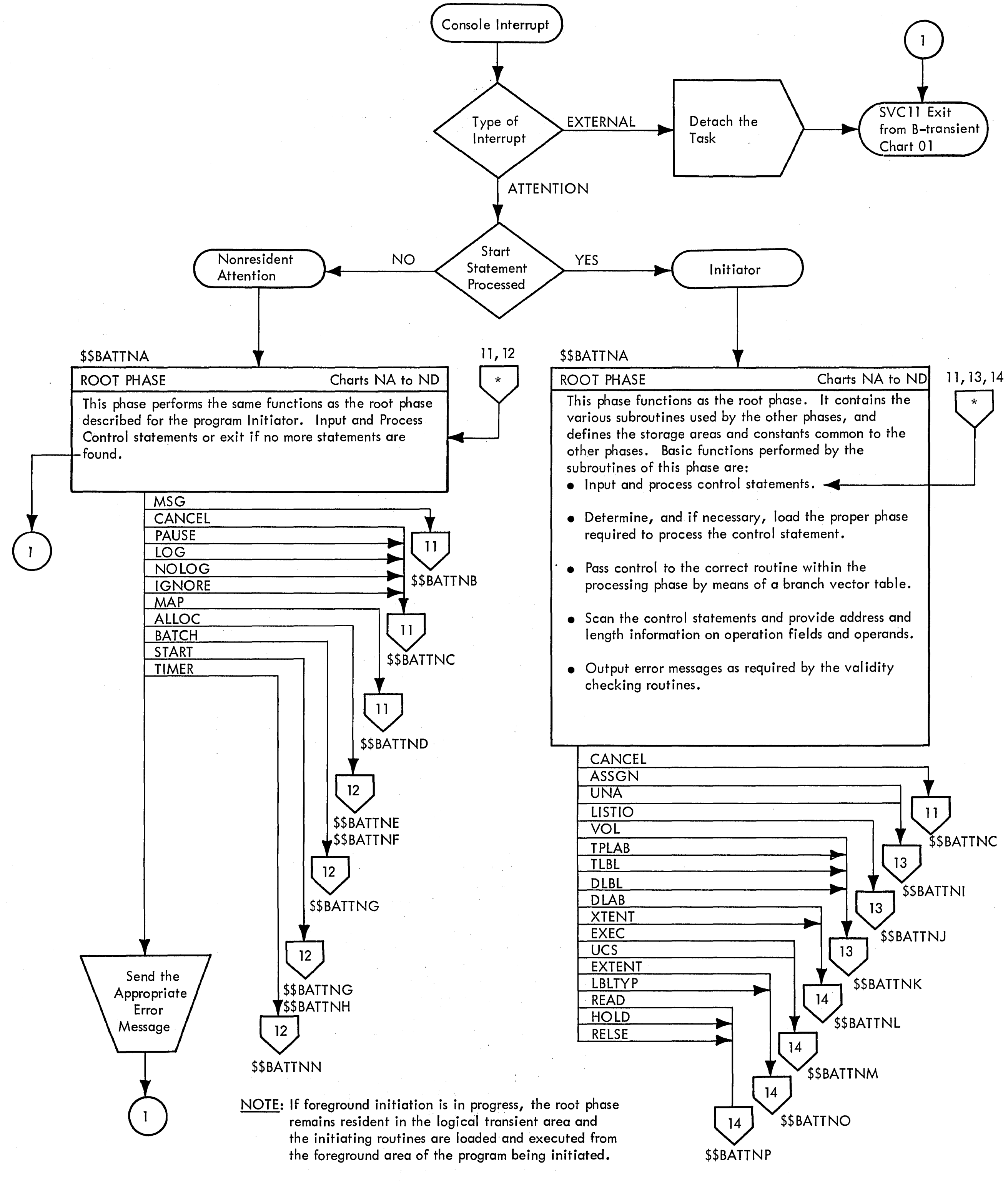
$$ANERRO

PHYSICAL ATTENTION Charts MU,MV

1. Send the message:
   ARb__ 1I40D__ _EMERGENCY CANCEL
   ARb__ (Response Fits Here.)__
2. Read the operator response:
   a. Valid responses:
      CANCEL bbb (cancel background programs)
      CANCEL FG1 (cancel foreground 1)
      CANCEL FG2 (cancel foreground 2)
      CANCEL BGb (cancel background)
   b. Invalid response (other than those listed above), send the message:
      ARb 1S0$\left\{\begin{matrix}1\\2\end{matrix}\right\}$ D INVALID STATEMENT
      ARb__ (Response Fits Here.)__
3. Set the cancel code in the PIB indicated by the response.
4. Cancel the program with the supervisor ATNCNL routine. Enter the routine by issuing a SVC3. Control is returned to this A-transient when the SVC old PSW is loaded by the supervisor.
5. Final exit from this transient is to the supervisor task selection mechanism after these conditions are set:
   - RF contains the address of the general exit (EXT02).
   - The phase name in ERBLOC has been modified to $$ANEER01 to signify return from physical attention routine.

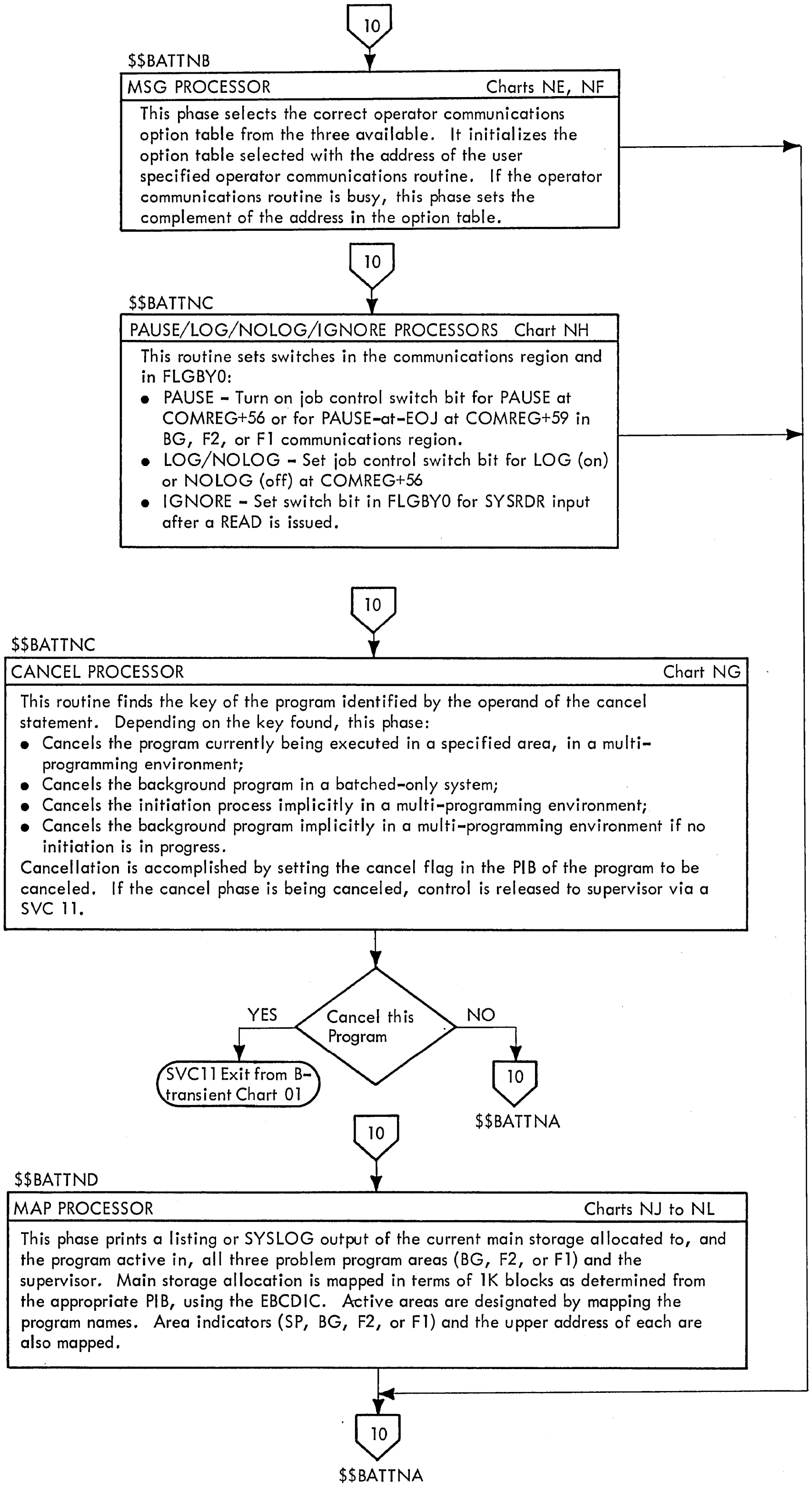
NOTE: Emergency cancel signalled by 2 consecutive attn requests.

Chart 10. Logical Transient Root Phase


Console Interrupt
Type of Interrupt
EXTERNAL
Detach the Task
1
SVC11 Exit from B-transient Chart 01
ATTENTION
Start Statement Processed
NO
YES
Nonresident Attention
Initiator
$$BATTNA
ROOT PHASE
Charts NA to ND
This phase performs the same functions as the root phase described for the program Initiator. Input and Process Control statements or exit if no more statements are found.
11, 12
*
1
MSG
CANCEL
PAUSE
LOG
NOLOG
IGNORE
MAP
ALLOC
BATCH
START
TIMER
11
$$BATTNB
11
$$BATTNC
11
$$BATTND
12
$$BATTNE
$$BATTNF
12
$$BATTNG
12
$$BATTNG
$$BATTNH
12
$$BATTNN
Send the Appropriate Error Message
1
$$BATTNA
ROOT PHASE
Charts NA to ND
This phase functions as the root phase. It contains the various subroutines used by the other phases, and defines the storage areas and constants common to the other phases. Basic functions performed by the subroutines of this phase are:
• Input and process control statements.
• Determine, and if necessary, load the proper phase required to process the control statement.
• Pass control to the correct routine within the processing phase by means of a branch vector table.
• Scan the control statements and provide address and length information on operation fields and operands.
• Output error messages as required by the validity checking routines.
11, 13, 14
*
CANCEL
ASSGN
UNA
LISTIO
VOL
TPLAB
TLBL
DLBL
DLAB
XTENT
EXEC
UCS
EXTENT
LBLTYP
READ
HOLD
RELSE
11
$$BATTNC
13
$$BATTNI
13
$$BATTNJ
13
$$BATTNK
14
$$BATTNL
14
$$BATTNM
14
$$BATTNO
14
$$BATTNP

NOTE: If foreground initiation is in progress, the root phase remains resident in the logical transient area and the initiating routines are loaded and executed from the foreground area of the program being initiated.

Chart 11. Logical Transient Attention Routines (Part 1 of 2)



$$BATTNB

MSG PROCESSOR Charts NE, NF

This phase selects the correct operator communications option table from the three available. It initializes the option table selected with the address of the user specified operator communications routine. If the operator communications routine is busy, this phase sets the complement of the address in the option table.

$$BATTNC

PAUSE/LOG/NOLOG/IGNORE PROCESSORS Chart NH

This routine sets switches in the communications region and in FLGBY0:

- PAUSE - Turn on job control switch bit for PAUSE at COMREG+56 or for PAUSE-at-EOJ at COMREG+59 in BG, F2, or F1 communications region.
- LOG/NOLOG - Set job control switch bit for LOG (on) or NOLOG (off) at COMREG+56
- IGNORE - Set switch bit in FLGBY0 for SYSRDR input after a READ is issued.

$$BATTNC

CANCEL PROCESSOR Chart NG

This routine finds the key of the program identified by the operand of the cancel statement. Depending on the key found, this phase:

- Cancels the program currently being executed in a specified area, in a multiprogramming environment;
- Cancels the background program in a batched-only system;
- Cancels the initiation process implicitly in a multi-programming environment;
- Cancels the background program implicitly in a multi-programming environment if no initiation is in progress.

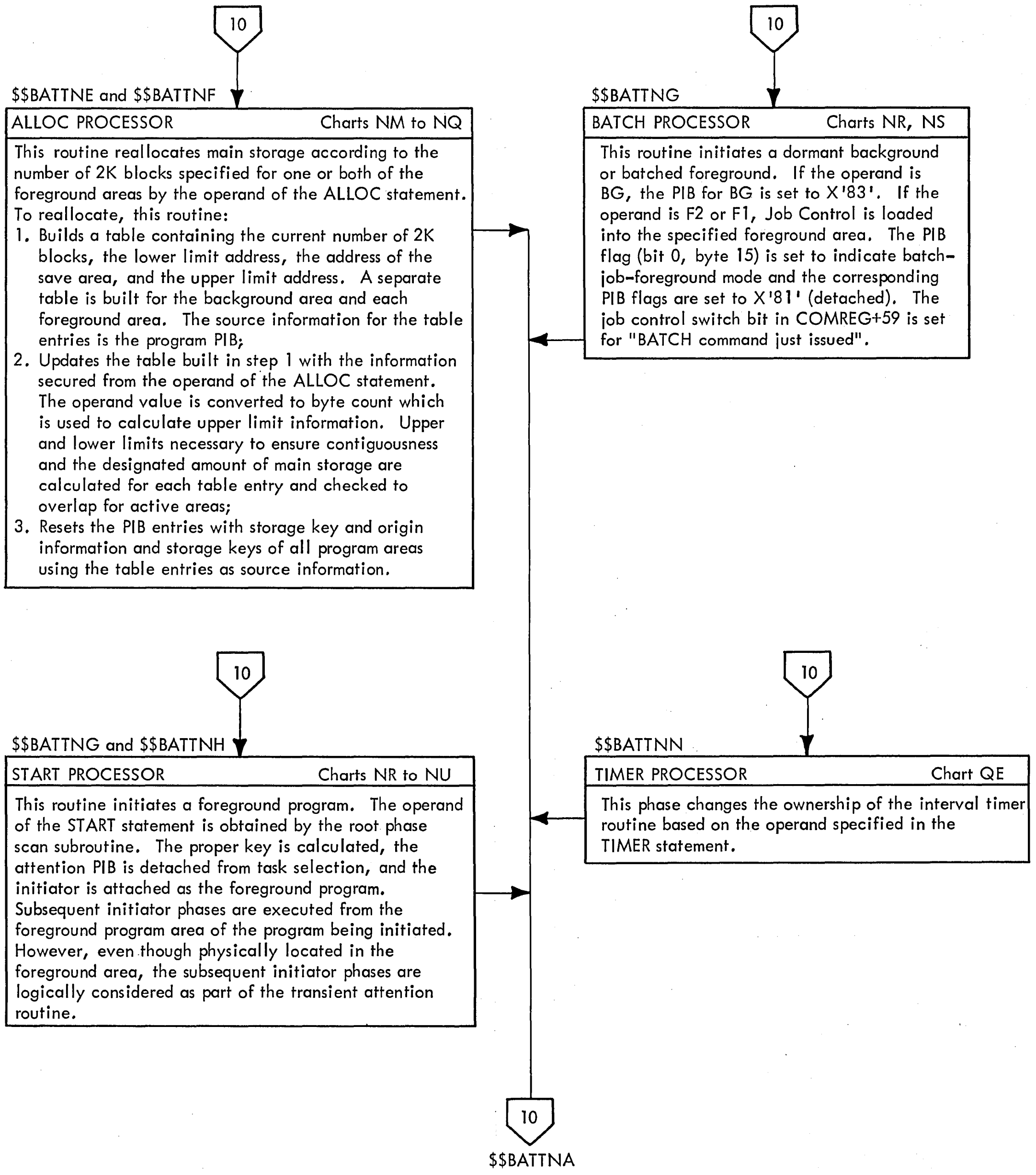
Cancellation is accomplished by setting the cancel flag in the PIB of the program to be canceled. If the cancel phase is being canceled, control is released to supervisor via a SVC 11.

$$BATTND

MAP PROCESSOR Charts NJ to NL

This phase prints a listing or SYSLOG output of the current main storage allocated to, and the program active in, all three problem program areas (BG, F2, or F1) and the supervisor. Main storage allocation is mapped in terms of 1K blocks as determined from the appropriate PIB, using the EBCDIC. Active areas are designated by mapping the program names. Area indicators (SP, BG, F2, or F1) and the upper address of each are also mapped.

Chart 12. Logical Transient Attention Routines (Part 2 of 2)



$$BATTNE and $$BATTNF

ALLOC PROCESSOR Charts NM to NQ

This routine reallocates main storage according to the number of 2K blocks specified for one or both of the foreground areas by the operand of the ALLOC statement. To reallocate, this routine:

1. Builds a table containing the current number of 2K blocks, the lower limit address, the address of the save area, and the upper limit address. A separate table is built for the background area and each foreground area. The source information for the table entries is the program PIB;
2. Updates the table built in step 1 with the information secured from the operand of the ALLOC statement. The operand value is converted to byte count which is used to calculate upper limit information. Upper and lower limits necessary to ensure contiguousness and the designated amount of main storage are calculated for each table entry and checked to overlap for active areas;
3. Resets the PIB entries with storage key and origin information and storage keys of all program areas using the table entries as source information.

$$BATTNG

BATCH PROCESSOR Charts NR, NS

This routine initiates a dormant background or batched foreground. If the operand is BG, the PIB for BG is set to X'83'. If the operand is F2 or F1, Job Control is loaded into the specified foreground area. The PIB flag (bit 0, byte 15) is set to indicate batch-job-foreground mode and the corresponding PIB flags are set to X'81' (detached). The job control switch bit in COMREG+59 is set for "BATCH command just issued".

$$BATTNG and $$BATTNH

START PROCESSOR Charts NR to NU

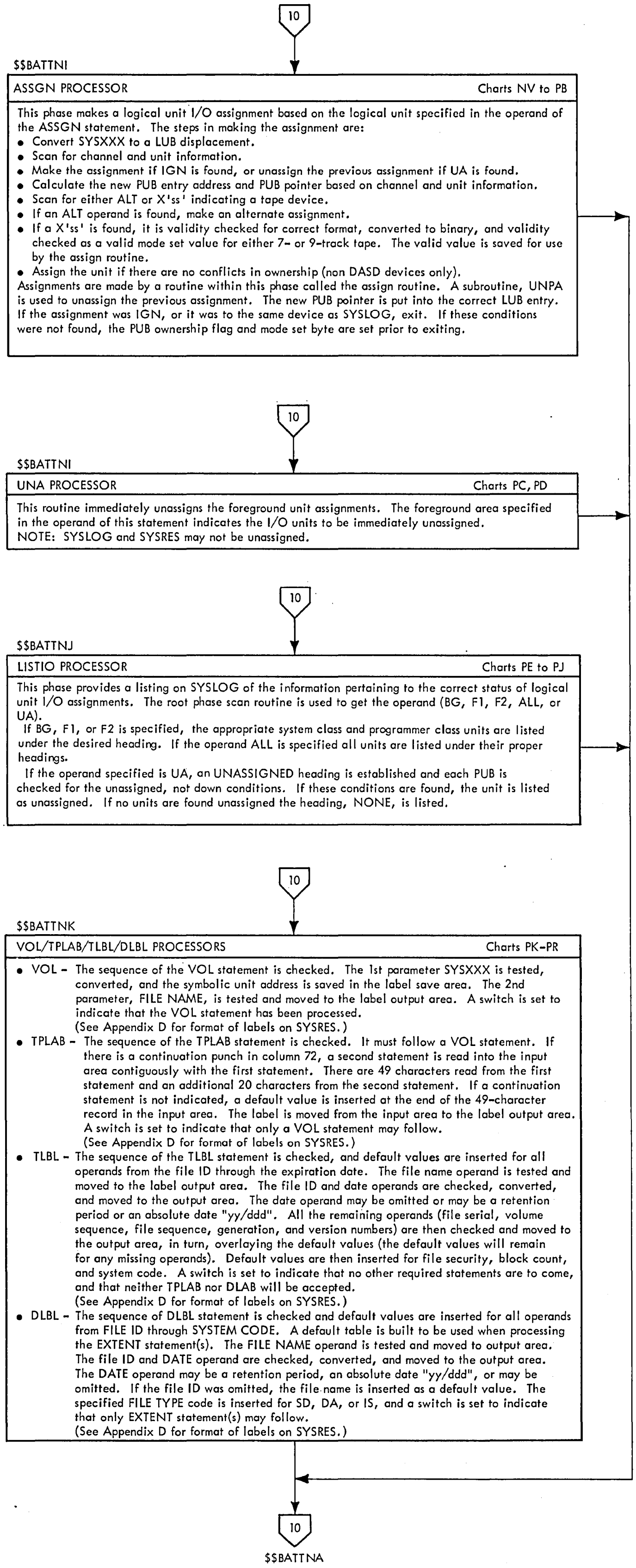
This routine initiates a foreground program. The operand of the START statement is obtained by the root phase scan subroutine. The proper key is calculated, the attention PIB is detached from task selection, and the initiator is attached as the foreground program. Subsequent initiator phases are executed from the foreground program area of the program being initiated. However, even though physically located in the foreground area, the subsequent initiator phases are logically considered as part of the transient attention routine.

$$BATTNN

TIMER PROCESSOR Chart QE

This phase changes the ownership of the interval timer routine based on the operand specified in the TIMER statement.

Chart 13. Logical Transient Initiator (Part 1 of 2)



ASSGN PROCESSOR Charts NV to PB

This phase makes a logical unit I/O assignment based on the logical unit specified in the operand of the ASSGN statement. The steps in making the assignment are:

- Convert SYSXXX to a LUB displacement.
- Scan for channel and unit information.
- Make the assignment if IGN is found, or unassign the previous assignment if UA is found.
- Calculate the new PUB entry address and PUB pointer based on channel and unit information.
- Scan for either ALT or X'ss' indicating a tape device.
- If an ALT operand is found, make an alternate assignment.
- If a X'ss' is found, it is validity checked for correct format, converted to binary, and validity checked as a valid mode set value for either 7- or 9-track tape. The valid value is saved for use by the assign routine.
- Assign the unit if there are no conflicts in ownership (non DASD devices only).

Assignments are made by a routine within this phase called the assign routine. A subroutine, UNPA is used to unassign the previous assignment. The new PUB pointer is put into the correct LUB entry. If the assignment was IGN, or it was to the same device as SYSLOG, exit. If these conditions were not found, the PUB ownership flag and mode set byte are set prior to exiting.

UNA PROCESSOR Charts PC, PD

This routine immediately unassigns the foreground unit assignments. The foreground area specified in the operand of this statement indicates the I/O units to be immediately unassigned.
NOTE: SYSLOG and SYSRES may not be unassigned.

LISTIO PROCESSOR Charts PE to PJ

This phase provides a listing on SYSLOG of the information pertaining to the correct status of logical unit I/O assignments. The root phase scan routine is used to get the operand (BG, F1, F2, ALL, or UA).

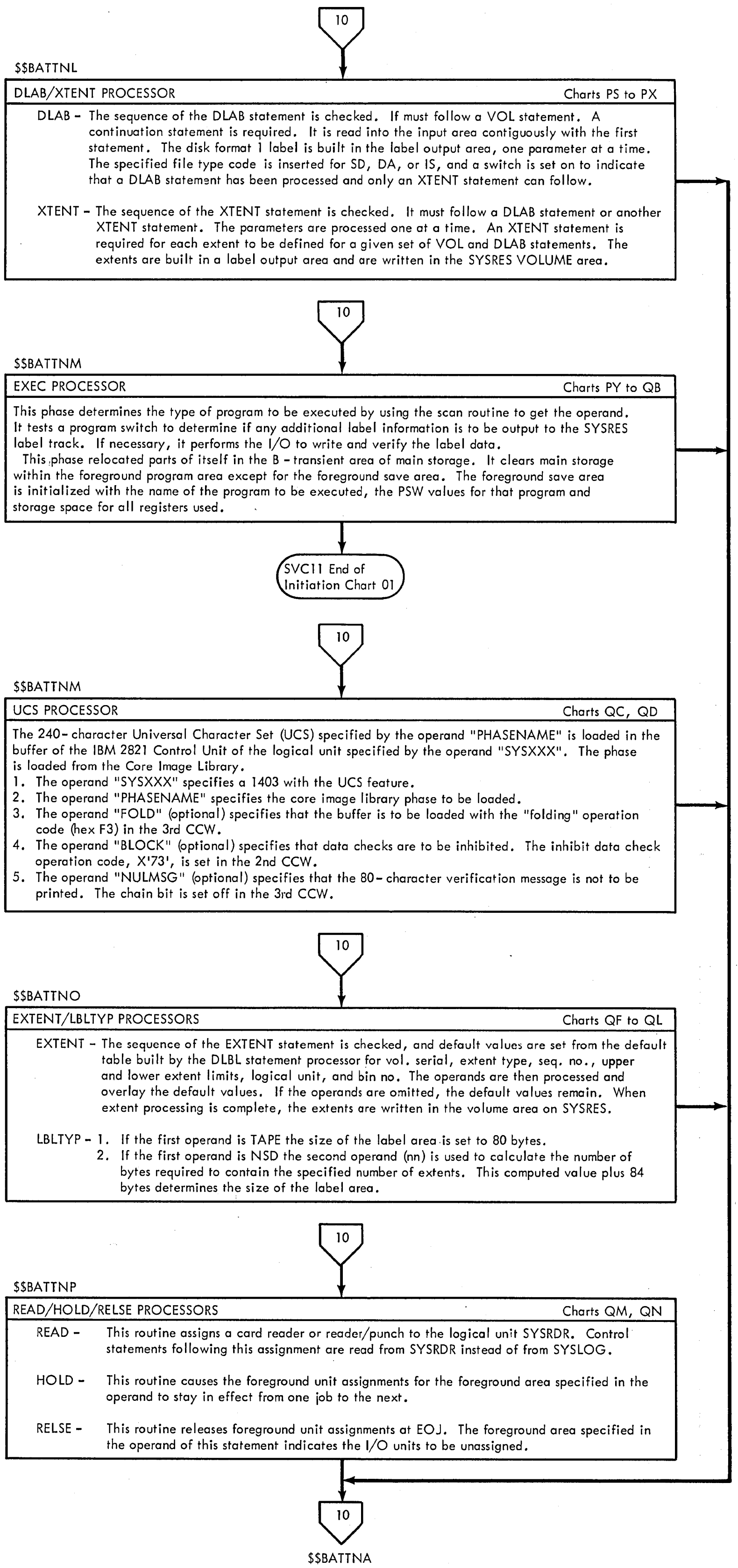
If BG, F1, or F2 is specified, the appropriate system class and programmer class units are listed under the desired heading. If the operand ALL is specified all units are listed under their proper headings.

If the operand specified is UA, an UNASSIGNED heading is established and each PUB is checked for the unassigned, not down conditions. If these conditions are found, the unit is listed as unassigned. If no units are found unassigned the heading, NONE, is listed.

VOL/TPLAB/TLBL/DLBL PROCESSORS Charts PK-PR

- VOL - The sequence of the VOL statement is checked. The 1st parameter SYSXXX is tested, converted, and the symbolic unit address is saved in the label save area. The 2nd parameter, FILE NAME, is tested and moved to the label output area. A switch is set to indicate that the VOL statement has been processed.
(See Appendix D for format of labels on SYSRES.)
- TPLAB - The sequence of the TPLAB statement is checked. It must follow a VOL statement. If there is a continuation punch in column 72, a second statement is read into the input area contiguously with the first statement. There are 49 characters read from the first statement and an additional 20 characters from the second statement. If a continuation statement is not indicated, a default value is inserted at the end of the 49-character record in the input area. The label is moved from the input area to the label output area. A switch is set to indicate that only a VOL statement may follow.
(See Appendix D for format of labels on SYSRES.)
- TLBL - The sequence of the TLBL statement is checked, and default values are inserted for all operands from the file ID through the expiration date. The file name operand is tested and moved to the label output area. The file ID and date operands are checked, converted, and moved to the output area. The date operand may be omitted or may be a retention period or an absolute date "yy/ddd". All the remaining operands (file serial, volume sequence, file sequence, generation, and version numbers) are then checked and moved to the output area, in turn, overlaying the default values (the default values will remain for any missing operands). Default values are then inserted for file security, block count, and system code. A switch is set to indicate that no other required statements are to come, and that neither TPLAB nor DLAB will be accepted.
(See Appendix D for format of labels on SYSRES.)
- DLBL - The sequence of DLBL statement is checked and default values are inserted for all operands from FILE ID through SYSTEM CODE. A default table is built to be used when processing the EXTENT statement(s). The FILE NAME operand is tested and moved to output area. The file ID and DATE operand are checked, converted, and moved to the output area. The DATE operand may be a retention period, an absolute date "yy/ddd", or may be omitted. If the file ID was omitted, the file name is inserted as a default value. The specified FILE TYPE code is inserted for SD, DA, or IS, and a switch is set to indicate that only EXTENT statement(s) may follow.
(See Appendix D for format of labels on SYSRES.)

Chart 14. Logical Transient Initiator (Part 2 of 2)



DLAB/XTENT PROCESSOR Charts PS to PX

DLAB - The sequence of the DLAB statement is checked. If must follow a VOL statement. A continuation statement is required. It is read into the input area contiguously with the first statement. The disk format 1 label is built in the label output area, one parameter at a time. The specified file type code is inserted for SD, DA, or IS, and a switch is set on to indicate that a DLAB statement has been processed and only an XTENT statement can follow.

XTENT - The sequence of the XTENT statement is checked. It must follow a DLAB statement or another XTENT statement. The parameters are processed one at a time. An XTENT statement is required for each extent to be defined for a given set of VOL and DLAB statements. The extents are built in a label output area and are written in the SYSRES VOLUME area.

EXEC PROCESSOR Charts PY to QB

This phase determines the type of program to be executed by using the scan routine to get the operand. It tests a program switch to determine if any additional label information is to be output to the SYSRES label track. If necessary, it performs the I/O to write and verify the label data.

This phase relocated parts of itself in the B - transient area of main storage. It clears main storage within the foreground program area except for the foreground save area. The foreground save area is initialized with the name of the program to be executed, the PSW values for that program and storage space for all registers used.

UCS PROCESSOR Charts QC, QD

The 240- character Universal Character Set (UCS) specified by the operand "PHASENAME" is loaded in the buffer of the IBM 2821 Control Unit of the logical unit specified by the operand "SYSXXX". The phase is loaded from the Core Image Library.

1. The operand "SYSXXX" specifies a 1403 with the UCS feature.
2. The operand "PHASENAME" specifies the core image library phase to be loaded.
3. The operand "FOLD" (optional) specifies that the buffer is to be loaded with the "folding" operation code (hex F3) in the 3rd CCW.
4. The operand "BLOCK" (optional) specifies that data checks are to be inhibited. The inhibit data check operation code, X'73', is set in the 2nd CCW.
5. The operand "NULMSG" (optional) specifies that the 80- character verification message is not to be printed. The chain bit is set off in the 3rd CCW.

EXTENT/LBLTYP PROCESSORS Charts QF to QL

EXTENT - The sequence of the EXTENT statement is checked, and default values are set from the default table built by the DLBL statement processor for vol. serial, extent type, seq. no., upper and lower extent limits, logical unit, and bin no. The operands are then processed and overlay the default values. If the operands are omitted, the default values remain. When extent processing is complete, the extents are written in the volume area on SYSRES.

LBLTYP - 1. If the first operand is TAPE the size of the label area is set to 80 bytes.
2. If the first operand is NSD the second operand (nn) is used to calculate the number of bytes required to contain the specified number of extents. This computed value plus 84 bytes determines the size of the label area.

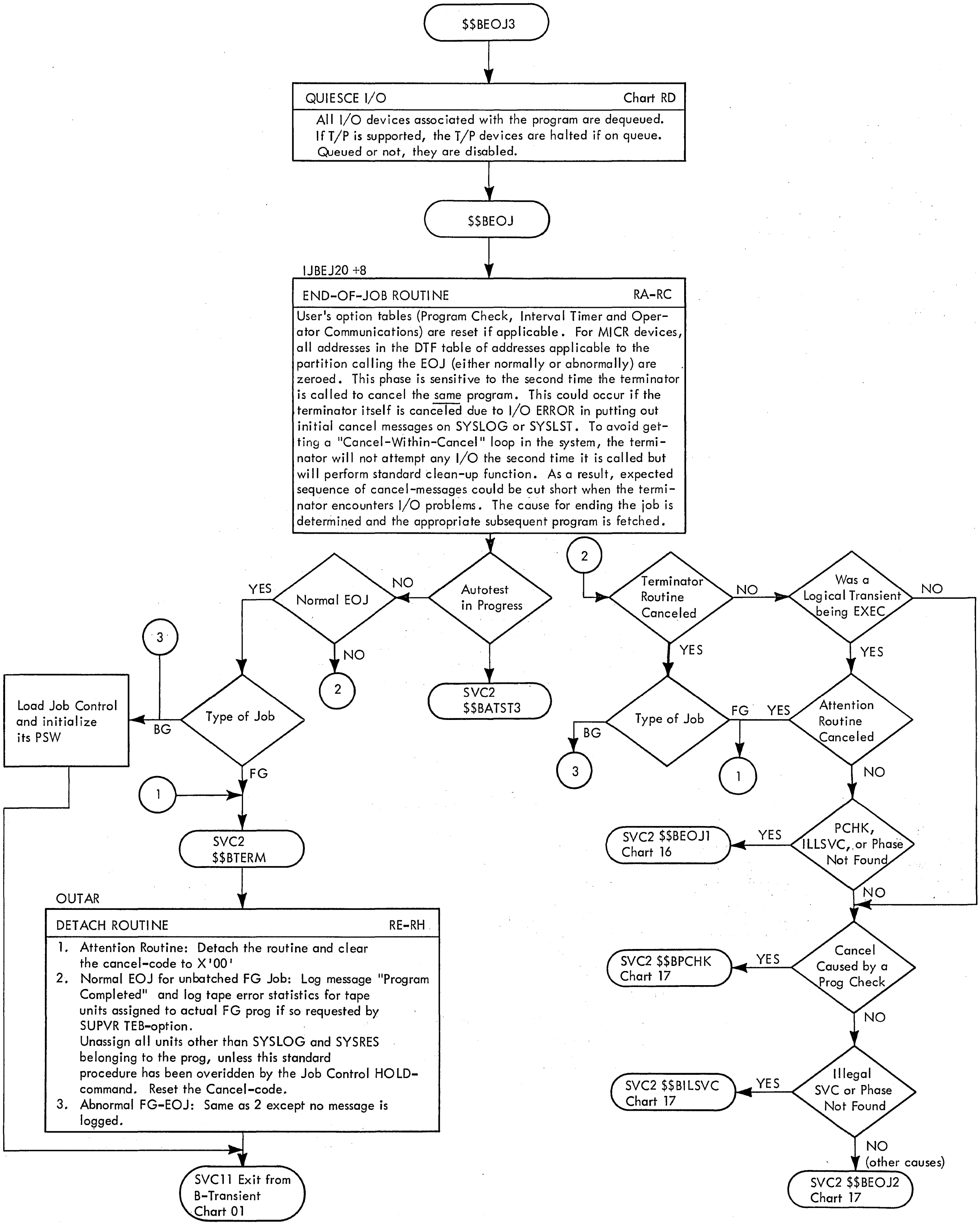
READ/HOLD/RELSE PROCESSORS Charts QM, QN

READ - This routine assigns a card reader or reader/punch to the logical unit SYSRDR. Control statements following this assignment are read from SYSRDR instead of from SYSLOG.

HOLD - This routine causes the foreground unit assignments for the foreground area specified in the operand to stay in effect from one job to the next.

RELSE - This routine releases foreground unit assignments at EOJ. The foreground area specified in the operand of this statement indicates the I/O units to be unassigned.

Chart 15. Logical Transient Terminator (Part 1 of 5)



QUIESCE I/O: All I/O devices associated with the program are dequeued. If T/P is supported, the T/P devices are halted if on queue. Queued or not, they are disabled.

END-OF-JOB ROUTINE: User's option tables (Program Check, Interval Timer and Operator Communications) are reset if applicable. For MICR devices, all addresses in the DTF table of addresses applicable to the partition calling the EOJ (either normally or abnormally) are zeroed. This phase is sensitive to the second time the terminator is called to cancel the same program. This could occur if the terminator itself is canceled due to I/O ERROR in putting out initial cancel messages on SYSLOG or SYSLST. To avoid getting a "Cancel-Within-Cancel" loop in the system, the terminator will not attempt any I/O the second time it is called but will perform standard clean-up function. As a result, expected sequence of cancel-messages could be cut short when the terminator encounters I/O problems. The cause for ending the job is determined and the appropriate subsequent program is fetched.

DETACH ROUTINE:

1. Attention Routine: Detach the routine and clear the cancel-code to X'00'
2. Normal EOJ for unbatched FG Job: Log message "Program Completed" and log tape error statistics for tape units assigned to actual FG prog if so requested by SUPVR TEB-option. Unassign all units other than SYSLOG and SYSRES belonging to the prog, unless this standard procedure has been overidden by the Job Control HOLD-command. Reset the Cancel-code.
3. Abnormal FG-EOJ: Same as 2 except no message is logged.

Chart 16. Logical Transient Terminator (Part 2 of 5)

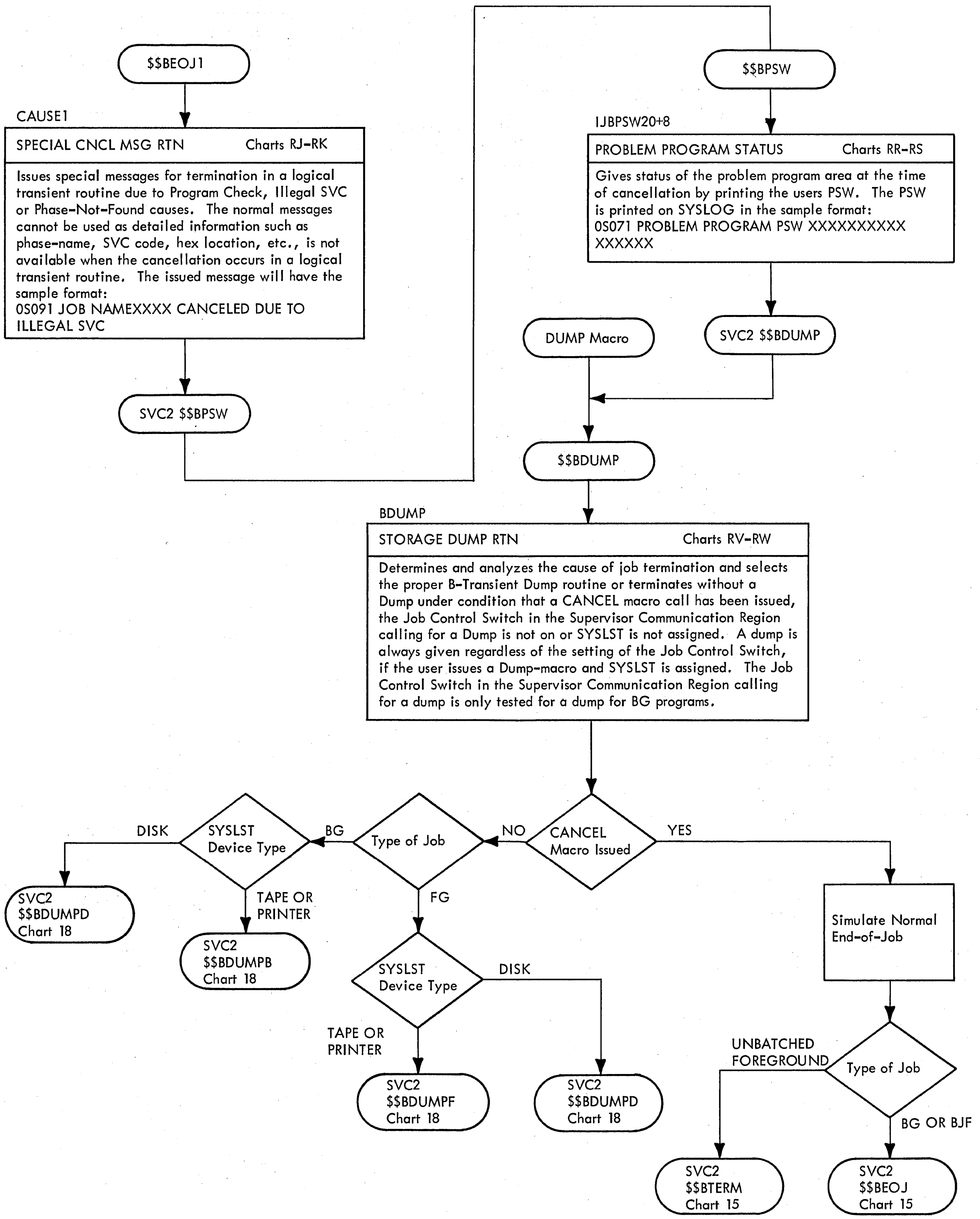
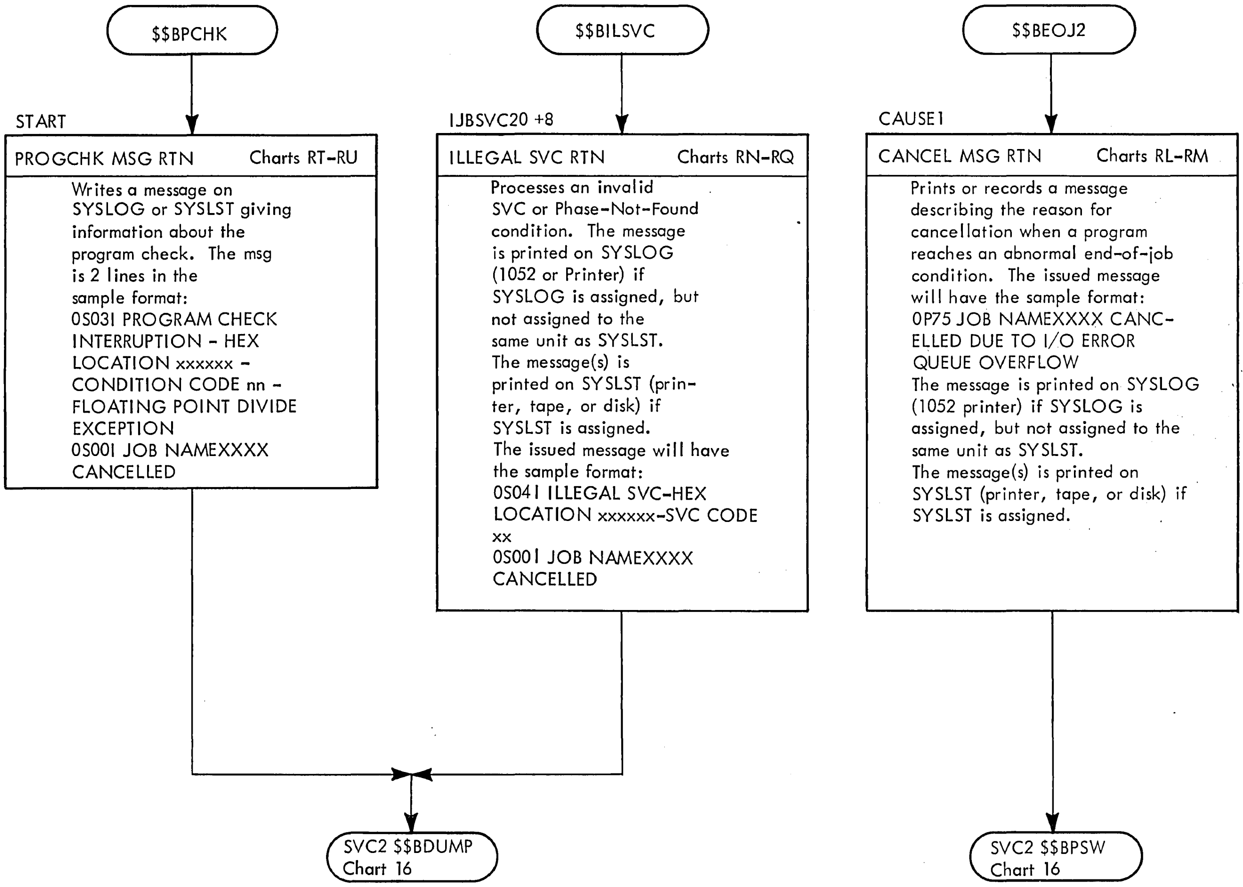

$$BEOJ1
CAUSE1
SPECIAL CNCL MSG RTN
Charts RJ-RK
Issues special messages for termination in a logical transient routine due to Program Check, Illegal SVC or Phase-Not-Found causes. The normal messages cannot be used as detailed information such as phase-name, SVC code, hex location, etc., is not available when the cancellation occurs in a logical transient routine. The issued message will have the sample format:
0S091 JOB NAMEXXXX CANCELED DUE TO ILLEGAL SVC
SVC2 $$BPSW
$$BPSW
IJBPSW20+8
PROBLEM PROGRAM STATUS
Charts RR-RS
Gives status of the problem program area at the time of cancellation by printing the users PSW. The PSW is printed on SYSLOG in the sample format:
0S071 PROBLEM PROGRAM PSW XXXXXXXXXX XXXXXX
DUMP Macro
SVC2 $$BDUMP
$$BDUMP
BDUMP
STORAGE DUMP RTN
Charts RV-RW
Determines and analyzes the cause of job termination and selects the proper B-Transient Dump routine or terminates without a Dump under condition that a CANCEL macro call has been issued, the Job Control Switch in the Supervisor Communication Region calling for a Dump is not on or SYSLST is not assigned. A dump is always given regardless of the setting of the Job Control Switch, if the user issues a Dump-macro and SYSLST is assigned. The Job Control Switch in the Supervisor Communication Region calling for a dump is only tested for a dump for BG programs.
CANCEL Macro Issued
NO
YES
Type of Job
BG
FG
SYSLST Device Type
DISK
TAPE OR PRINTER
SVC2 $$BDUMPD Chart 18
SVC2 $$BDUMPB Chart 18
SYSLST Device Type
DISK
TAPE OR PRINTER
SVC2 $$BDUMPF Chart 18
SVC2 $$BDUMPD Chart 18
Simulate Normal End-of-Job
Type of Job
UNBATCHED FOREGROUND
BG OR BJF
SVC2 $$BTERM Chart 15
SVC2 $$BEOJ Chart 15

Chart 17. Logical Transient Terminator (Part 3 of 5)

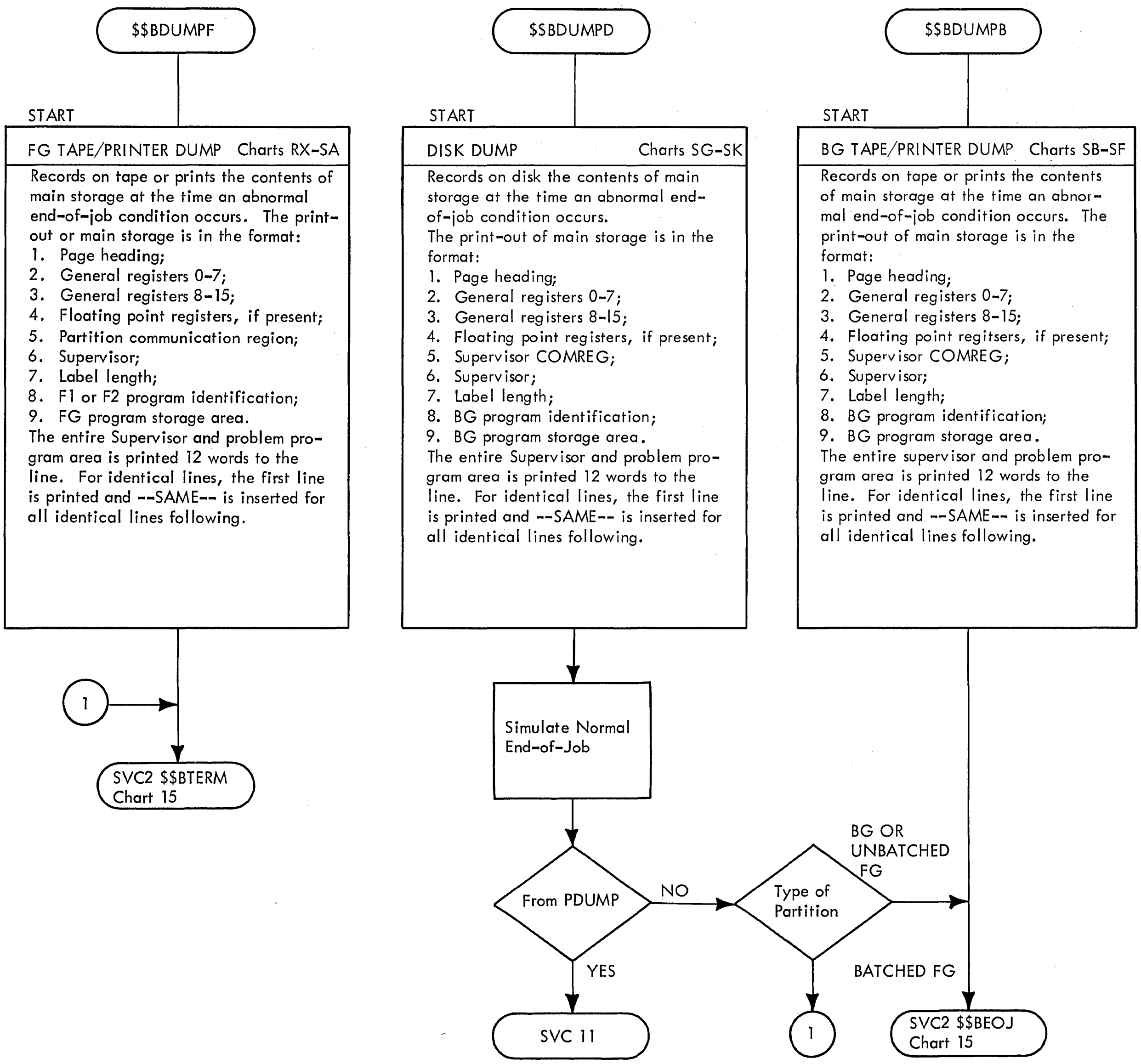


Writes a message on SYSLOG or SYSLST giving information about the program check. The msg is 2 lines in the sample format:
0S031 PROGRAM CHECK INTERRUPTION - HEX LOCATION xxxxxx - CONDITION CODE nn - FLOATING POINT DIVIDE EXCEPTION
0S001 JOB NAMEXXXX CANCELLED

Processes an invalid SVC or Phase-Not-Found condition. The message is printed on SYSLOG (1052 or Printer) if SYSLOG is assigned, but not assigned to the same unit as SYSLST. The message(s) is printed on SYSLST (printer, tape, or disk) if SYSLST is assigned. The issued message will have the sample format:
0S041 ILLEGAL SVC-HEX LOCATION xxxxxx-SVC CODE xx
0S001 JOB NAMEXXXX CANCELLED

Prints or records a message describing the reason for cancellation when a program reaches an abnormal end-of-job condition. The issued message will have the sample format:
0P75 JOB NAMEXXXX CANCELLED DUE TO I/O ERROR QUEUE OVERFLOW
The message is printed on SYSLOG (1052 printer) if SYSLOG is assigned, but not assigned to the same unit as SYSLST.
The message(s) is printed on SYSLST (printer, tape, or disk) if SYSLST is assigned.

Chart 18. Logical Transient Terminator (Part 4 of 5)



Records on tape or prints the contents of main storage at the time an abnormal end-of-job condition occurs. The print-out or main storage is in the format:
1. Page heading;
2. General registers 0-7;
3. General registers 8-15;
4. Floating point registers, if present;
5. Partition communication region;
6. Supervisor;
7. Label length;
8. F1 or F2 program identification;
9. FG program storage area.

The entire Supervisor and problem program area is printed 12 words to the line. For identical lines, the first line is printed and --SAME-- is inserted for all identical lines following.

Records on disk the contents of main storage at the time an abnormal end-of-job condition occurs.
The print-out of main storage is in the format:
1. Page heading;
2. General registers 0-7;
3. General registers 8-15;
4. Floating point registers, if present;
5. Supervisor COMREG;
6. Supervisor;
7. Label length;
8. BG program identification;
9. BG program storage area.

The entire Supervisor and problem program area is printed 12 words to the line. For identical lines, the first line is printed and --SAME-- is inserted for all identical lines following.

Records on tape or prints the contents of main storage at the time an abnormal end-of-job condition occurs. The print-out of main storage is in the format:
1. Page heading;
2. General registers 0-7;
3. General registers 8-15;
4. Floating point regitsers, if present;
5. Supervisor COMREG;
6. Supervisor;
7. Label length;
8. BG program identification;
9. BG program storage area.

The entire supervisor and problem program area is printed 12 words to the line. For identical lines, the first line is printed and --SAME-- is inserted for all identical lines following.

Chart 19. Logical Transient Terminator (Part 5 of 5)

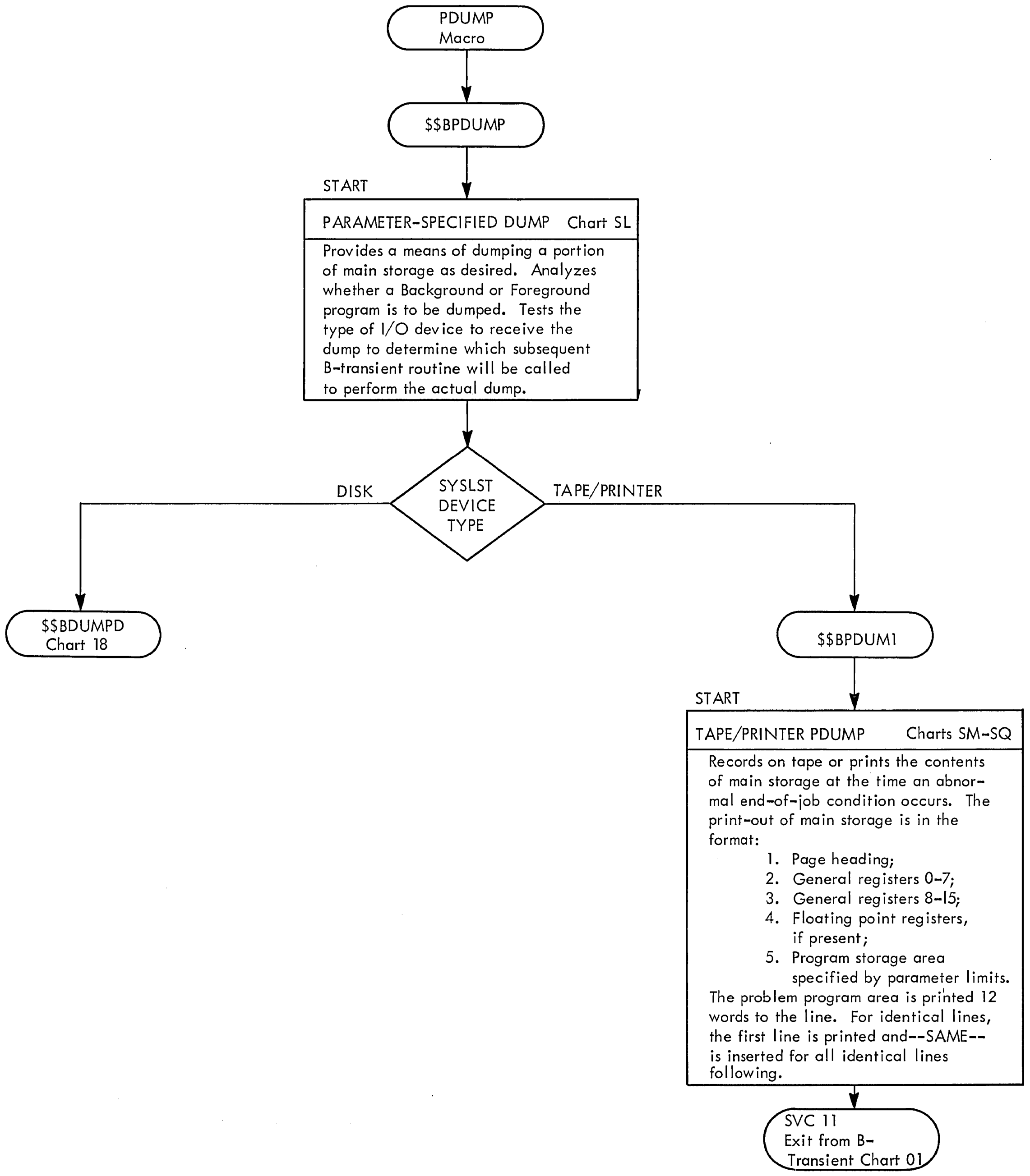

PDUMP Macro
$$BPDUMP
START
PARAMETER-SPECIFIED DUMP Chart SL
Provides a means of dumping a portion of main storage as desired. Analyzes whether a Background or Foreground program is to be dumped. Tests the type of I/O device to receive the dump to determine which subsequent B-transient routine will be called to perform the actual dump.
DISK
SYSLST DEVICE TYPE
TAPE/PRINTER
$$BDUMPD Chart 18
$$BPDUM1
START
TAPE/PRINTER PDUMP Charts SM-SQ
Records on tape or prints the contents of main storage at the time an abnormal end-of-job condition occurs. The print-out of main storage is in the format:
1. Page heading;
2. General registers 0-7;
3. General registers 8-15;
4. Floating point registers, if present;
5. Program storage area specified by parameter limits.
The problem program area is printed 12 words to the line. For identical lines, the first line is printed and--SAME-- is inserted for all identical lines following.
SVC 11 Exit from B-Transient Chart 01

Chart AA. $$A$SUP1 - SUPVR Macro, General Entry
Refer to Chart 01.

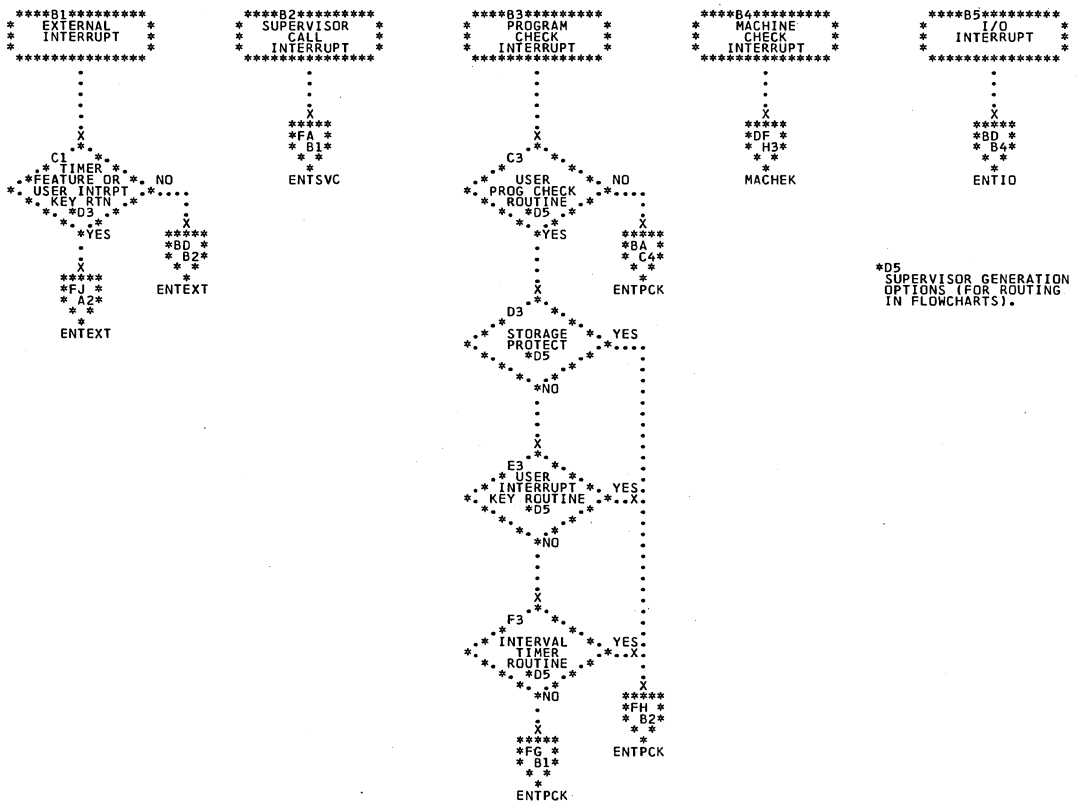


Chart BA. $$A$SUP1 - FOPT Macro, General Cancels and Program Check without PC Routine
Refer to Charts 02 and 03.

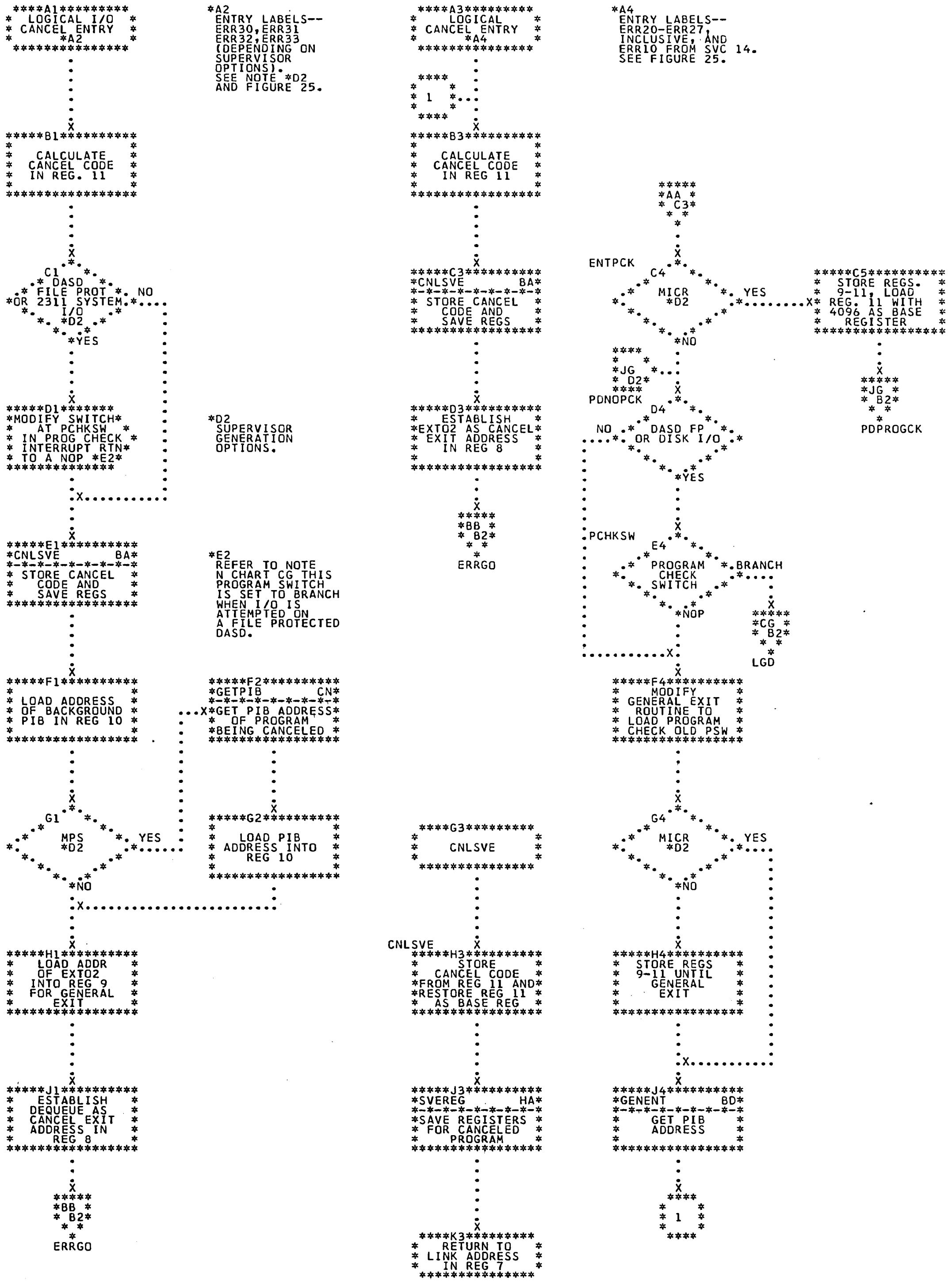
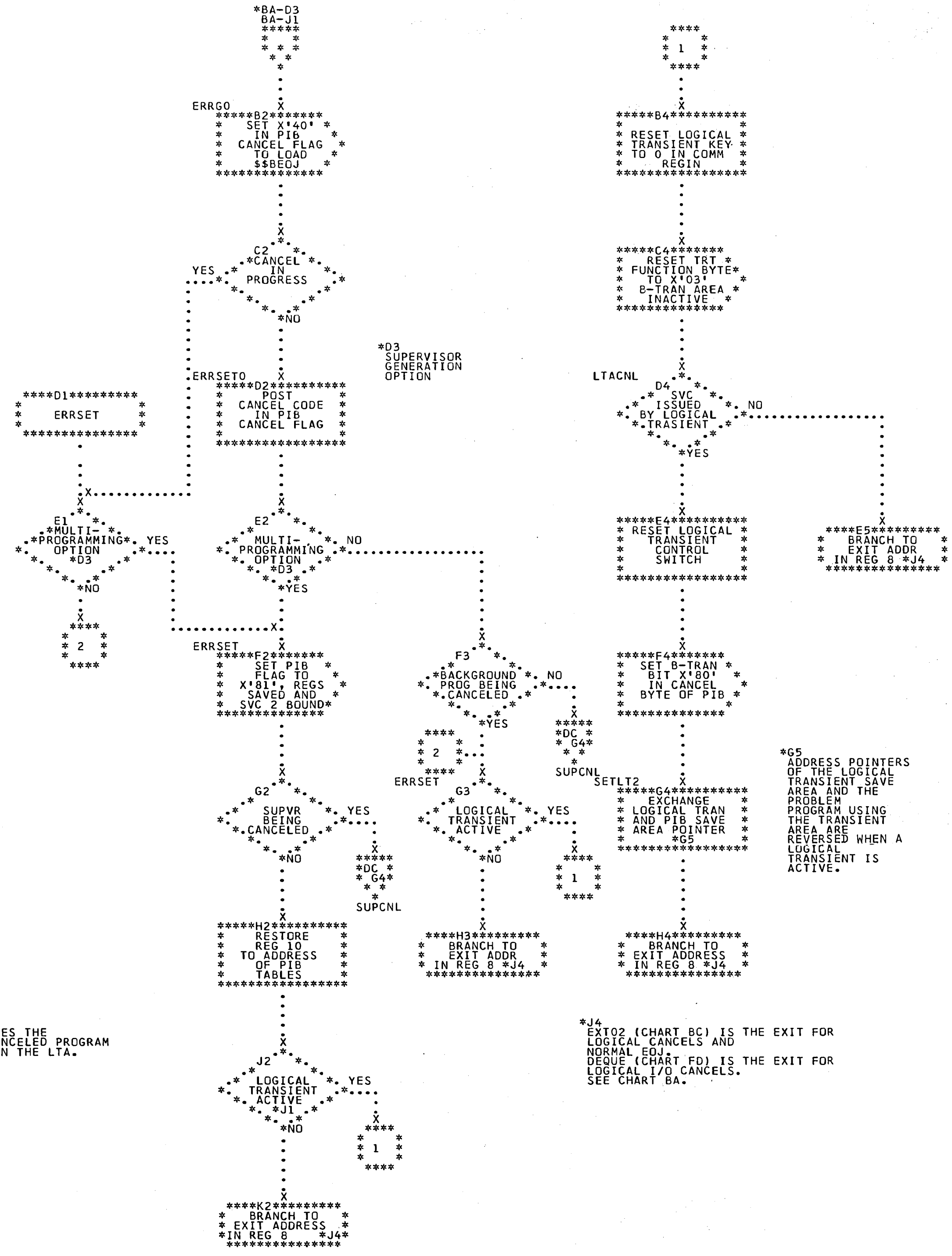

A1
LOGICAL I/O CANCEL ENTRY *A2
*A2
ENTRY LABELS-- ERR30,ERR31 ERR32,ERR33 (DEPENDING ON SUPERVISOR OPTIONS). SEE NOTE *D2 AND FIGURE 25.
A3
LOGICAL CANCEL ENTRY *A4
*A4
ENTRY LABELS-- ERR20-ERR27, INCLUSIVE, AND ERR10 FROM SVC 14. SEE FIGURE 25.
1
B1
CALCULATE CANCEL CODE IN REG. 11
B3
CALCULATE CANCEL CODE IN REG 11
AA C3
ENTPCK
C1
DASD FILE PROT OR 2311 SYSTEM I/O *D2
NO
YES
C3
CNLSVE BA
STORE CANCEL CODE AND SAVE REGS
C4
MICR *D2
YES
NO
C5
STORE REGS. 9-11, LOAD REG. 11 WITH 4096 AS BASE REGISTER
JG D2
PDNOPCK
JG B2
PDPROGCK
D1
MODIFY SWITCH AT PCHKSW IN PROG CHECK INTERRUPT RTN TO A NOP *E2
*D2
SUPERVISOR GENERATION OPTIONS.
D3
ESTABLISH EXTO2 AS CANCEL EXIT ADDRESS IN REG 8
D4
DASD FP OR DISK I/O
NO
YES
BB B2
ERRGO
PCHKSW
E1
CNLSVE BA
STORE CANCEL CODE AND SAVE REGS
*E2
REFER TO NOTE N CHART CG THIS PROGRAM SWITCH IS SET TO BRANCH WHEN I/O IS ATTEMPTED ON A FILE PROTECTED DASD.
E4
PROGRAM CHECK SWITCH
BRANCH
NOP
CG B2
LGD
F1
LOAD ADDRESS OF BACKGROUND PIB IN REG 10
F2
GETPIB CN
GET PIB ADDRESS OF PROGRAM BEING CANCELED
F4
MODIFY GENERAL EXIT ROUTINE TO LOAD PROGRAM CHECK OLD PSW
G1
MPS *D2
YES
NO
G2
LOAD PIB ADDRESS INTO REG 10
G3
CNLSVE
G4
MICR *D2
YES
NO
H1
LOAD ADDR OF EXTO2 INTO REG 9 FOR GENERAL EXIT
CNLSVE
H3
STORE CANCEL CODE FROM REG 11 AND RESTORE REG 11 AS BASE REG
H4
STORE REGS 9-11 UNTIL GENERAL EXIT
J1
ESTABLISH DEQUEUE AS CANCEL EXIT ADDRESS IN REG 8
J3
SVEREG HA
SAVE REGISTERS FOR CANCELED PROGRAM
J4
GENENT BD
GET PIB ADDRESS
BB B2
ERRGO
1
K3
RETURN TO LINK ADDRESS IN REG 7

Chart BB. $$A$SUP1 - FOPT Macro, General Cancel Subroutine
Refer to Chart 03.


*BA-D3
BA-J1
ERRGO
B2
SET X'40' IN PIB CANCEL FLAG TO LOAD $$BEOJ
C2
CANCEL IN PROGRESS
YES
NO
ERRSETO
D2
POST CANCEL CODE IN PIB CANCEL FLAG
D1
ERRSET
E1
MULTI-PROGRAMMING OPTION *D3
YES
NO
2
E2
MULTI-PROGRAMMING OPTION *D3
NO
YES
ERRSET
F2
SET PIB FLAG TO X'81', REGS SAVED AND SVC 2 BOUND
G2
SUPVR BEING CANCELED
YES
DC G4
SUPCNL
NO
H2
RESTORE REG 10 TO ADDRESS OF PIB TABLES
J2
LOGICAL TRANSIENT ACTIVE *J1
YES
1
NO
K2
BRANCH TO EXIT ADDRESS IN REG 8 *J4
F3
BACKGROUND PROG BEING CANCELED
NO
DC G4
SUPCNL
YES
2
ERRSET
G3
LOGICAL TRANSIENT ACTIVE
YES
1
NO
H3
BRANCH TO EXIT ADDR IN REG 8 *J4
1
B4
RESET LOGICAL TRANSIENT KEY TO 0 IN COMM REGIN
C4
RESET TRT FUNCTION BYTE TO X'03' B-TRAN AREA INACTIVE
LTACNL
D4
SVC ISSUED BY LOGICAL TRASIENT
NO
YES
E5
BRANCH TO EXIT ADDR IN REG 8 *J4
E4
RESET LOGICAL TRANSIENT CONTROL SWITCH
F4
SET B-TRAN BIT X'80' IN CANCEL BYTE OF PIB
SETLT2
G4
EXCHANGE LOGICAL TRAN AND PIB SAVE AREA POINTER *G5
H4
BRANCH TO EXIT ADDRESS IN REG 8 *J4

*D3
SUPERVISOR
GENERATION
OPTION

*G5
ADDRESS POINTERS OF THE LOGICAL TRANSIENT SAVE AREA AND THE PROBLEM PROGRAM USING THE TRANSIENT AREA ARE REVERSED WHEN A LOGICAL TRANSIENT IS ACTIVE.

*J1
DOES THE CANCELED PROGRAM OWN THE LTA.

*J4
EXT02 (CHART BC) IS THE EXIT FOR LOGICAL CANCELS AND NORMAL EOJ.
DEQUE (CHART FD) IS THE EXIT FOR LOGICAL I/O CANCELS.
SEE CHART BA.

Chart BC. $$A$SUP1 - FOPT Macro, General Exits
Refer to Chart 01.

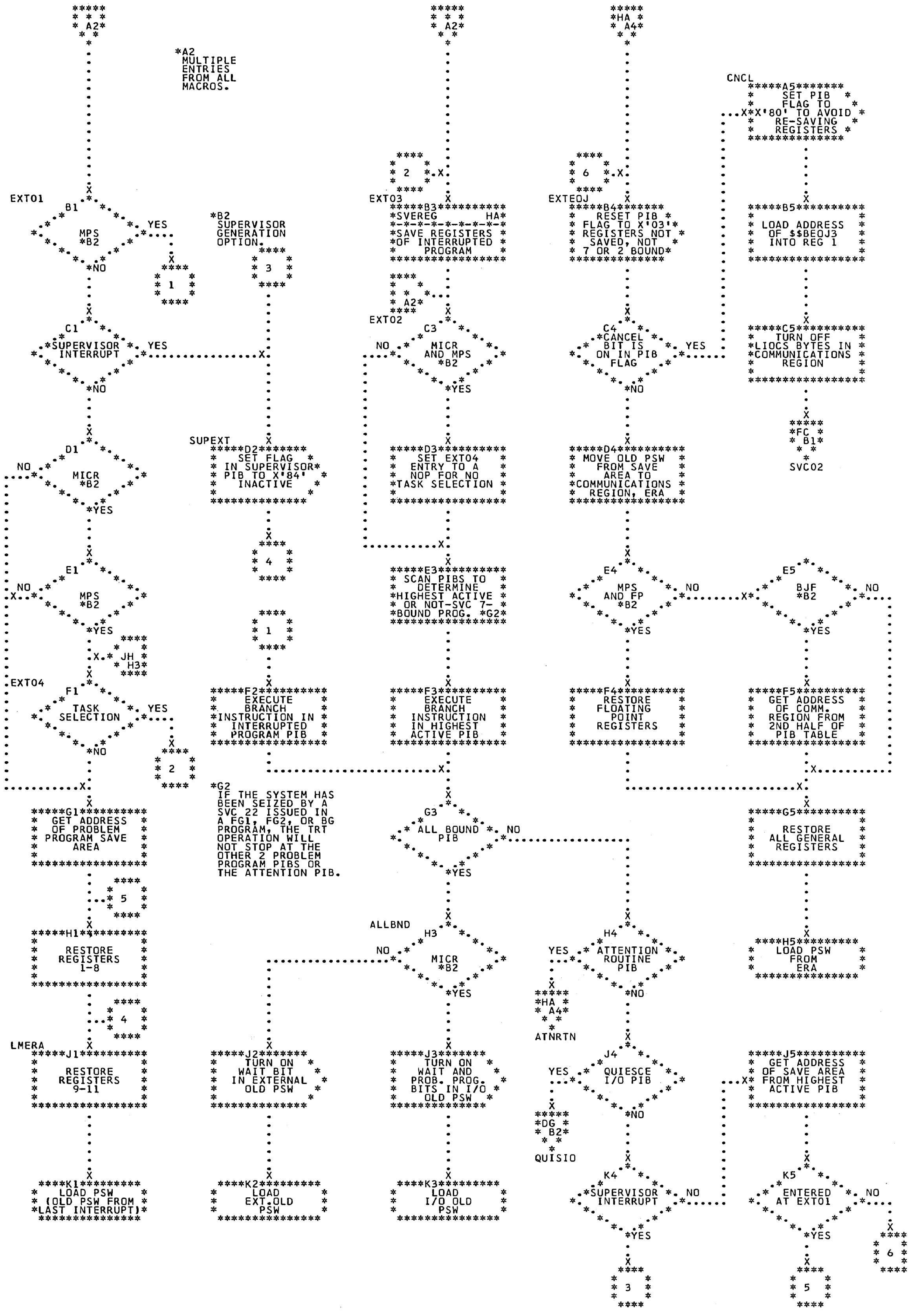

A2
*A2 MULTIPLE ENTRIES FROM ALL MACROS.
EXT01
B1
MPS *B2
YES
NO
1
*B2 SUPERVISOR GENERATION OPTION.
3
C1
SUPERVISOR INTERRUPT
YES
NO
D1
MICR *B2
NO
YES
SUPEXT
D2
SET FLAG IN SUPERVISOR PIB TO X'84' INACTIVE
4
E1
MPS *B2
NO
YES
JH H3
1
EXT04
F1
TASK SELECTION
YES
NO
2
F2
EXECUTE BRANCH INSTRUCTION IN INTERRUPTED PROGRAM PIB
G1
GET ADDRESS OF PROBLEM PROGRAM SAVE AREA
*G2 IF THE SYSTEM HAS BEEN SEIZED BY A SVC 22 ISSUED IN A FG1, FG2, OR BG PROGRAM, THE TRT OPERATION WILL NOT STOP AT THE OTHER 2 PROBLEM PROGRAM PIBS OR THE ATTENTION PIB.
5
H1
RESTORE REGISTERS 1-8
4
LMERA
J1
RESTORE REGISTERS 9-11
J2
TURN ON WAIT BIT IN EXTERNAL OLD PSW
K1
LOAD PSW (OLD PSW FROM LAST INTERRUPT)
K2
LOAD EXT.OLD PSW
A2
2
EXT03
B3
SVEREG HA
SAVE REGISTERS OF INTERRUPTED PROGRAM
A2
EXT02
C3
MICR AND MPS *B2
NO
YES
D3
SET EXT04 ENTRY TO A NOP FOR NO TASK SELECTION
E3
SCAN PIBS TO DETERMINE HIGHEST ACTIVE OR NOT-SVC 7- BOUND PROG. *G2
F3
EXECUTE BRANCH INSTRUCTION IN HIGHEST ACTIVE PIB
G3
ALL BOUND PIB
NO
YES
ALLBND
H3
MICR *B2
NO
YES
J3
TURN ON WAIT AND PROB. PROG. BITS IN I/O OLD PSW
K3
LOAD I/O OLD PSW
HA A4
6
EXTEOJ
B4
RESET PIB FLAG TO X'03' REGISTERS NOT SAVED, NOT 7 OR 2 BOUND
C4
CANCEL BIT IS ON IN PIB FLAG
YES
NO
D4
MOVE OLD PSW FROM SAVE AREA TO COMMUNICATIONS REGION, ERA
E4
MPS AND FP *B2
NO
YES
F4
RESTORE FLOATING POINT REGISTERS
H4
ATTENTION ROUTINE PIB
YES
NO
HA A4
ATNRTN
J4
QUIESCE I/O PIB
YES
NO
DG B2
QUISIO
K4
SUPERVISOR INTERRUPT
NO
YES
3
CNCL
A5
SET PIB FLAG TO X'80' TO AVOID RE-SAVING REGISTERS
B5
LOAD ADDRESS OF $$BEOJ3 INTO REG 1
C5
TURN OFF LIOCS BYTES IN COMMUNICATIONS REGION
FC B1
SVC02
E5
BJF *B2
NO
YES
F5
GET ADDRESS OF COMM. REGION FROM 2ND HALF OF PIB TABLE
G5
RESTORE ALL GENERAL REGISTERS
H5
LOAD PSW FROM ERA
J5
GET ADDRESS OF SAVE AREA FROM HIGHEST ACTIVE PIB
K5
ENTERED AT EXT01
NO
YES
6
5

Chart BD. $$A$SUP1 - FOPT Macro, General Entry
Refer to Chart 01.

EXTERNAL INTERRUPT WITHOUT TIMER FEATURE
AND WITHOUT USER INTERRUPT KEY ROUTINE.

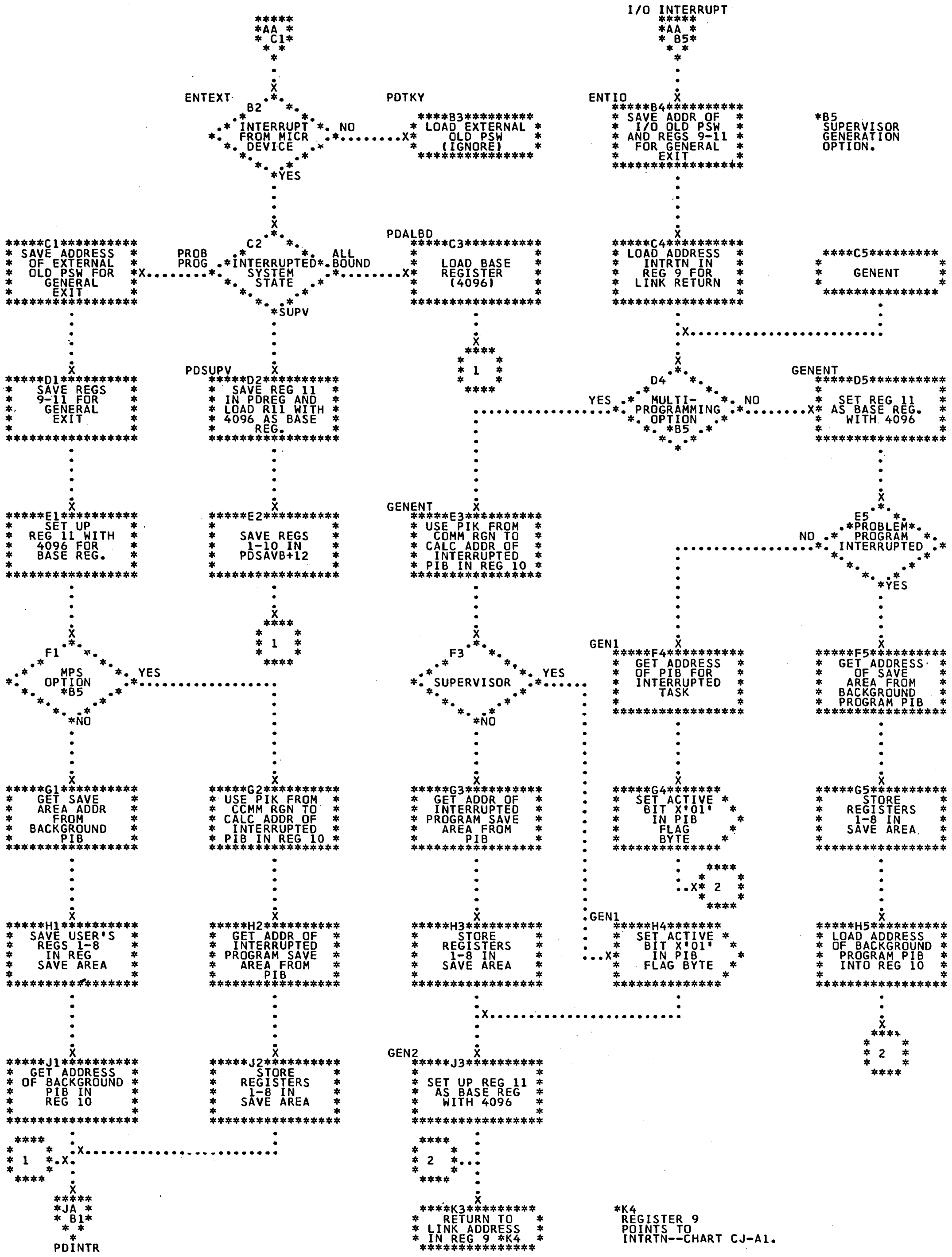
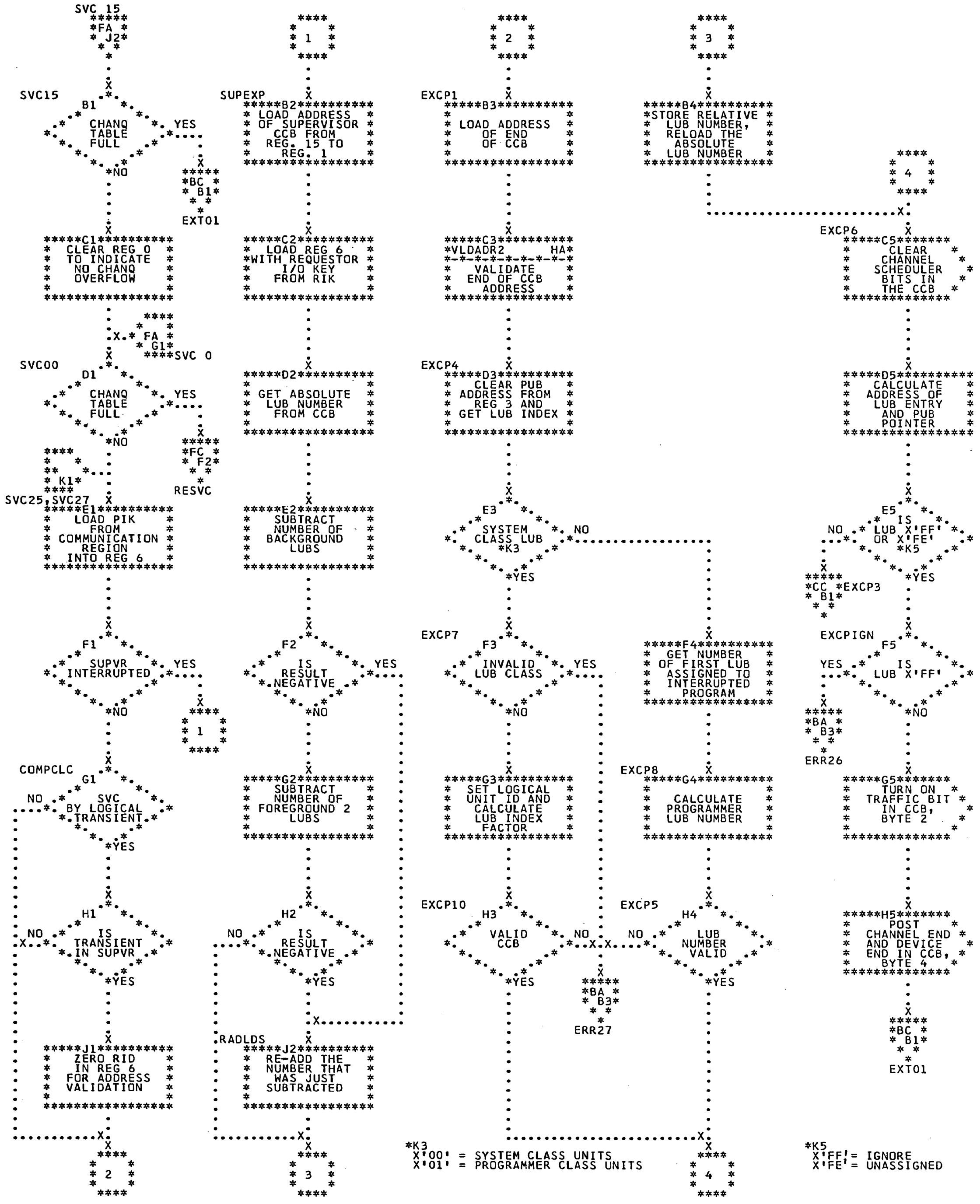

AA C1
ENTEXT B2 INTERRUPT FROM MICR DEVICE
NO
YES
PDTKY B3 LOAD EXTERNAL OLD PSW (IGNORE)
I/O INTERRUPT
AA B5
ENTIO B4 SAVE ADDR OF I/O OLD PSW AND REGS 9-11 FOR GENERAL EXIT
*B5 SUPERVISOR GENERATION OPTION.
C1 SAVE ADDRESS OF EXTERNAL OLD PSW FOR GENERAL EXIT
C2 INTERRUPTED SYSTEM STATE
PROB PROG
ALL BOUND
SUPV
PDALBD C3 LOAD BASE REGISTER (4096)
C4 LOAD ADDRESS INTRTN IN REG 9 FOR LINK RETURN
C5 GENENT
D1 SAVE REGS 9-11 FOR GENERAL EXIT
PDSUPV D2 SAVE REG 11 IN PDREG AND LOAD R11 WITH 4096 AS BASE REG.
1
D4 MULTI-PROGRAMMING OPTION *B5
YES
NO
GENENT D5 SET REG 11 AS BASE REG. WITH 4096
E1 SET UP REG 11 WITH 4096 FOR BASE REG.
E2 SAVE REGS 1-10 IN PDSAVB+12
GENENT E3 USE PIK FROM COMM RGN TO CALC ADDR OF INTERRUPTED PIB IN REG 10
E5 PROBLEM PROGRAM INTERRUPTED
NO
YES
1
F1 MPS OPTION *B5
YES
NO
F3 SUPERVISOR
YES
NO
GEN1 F4 GET ADDRESS OF PIB FOR INTERRUPTED TASK
F5 GET ADDRESS OF SAVE AREA FROM BACKGROUND PROGRAM PIB
G1 GET SAVE AREA ADDR FROM BACKGROUND PIB
G2 USE PIK FROM CCMM RGN TO CALC ADDR OF INTERRUPTED PIB IN REG 10
G3 GET ADDR OF INTERRUPTED PROGRAM SAVE AREA FROM PIB
G4 SET ACTIVE BIT X'01' IN PIB FLAG BYTE
G5 STORE REGISTERS 1-8 IN SAVE AREA.
2
H1 SAVE USER'S REGS 1-8 IN REG SAVE AREA
H2 GET ADDR OF INTERRUPTED PROGRAM SAVE AREA FROM PIB
H3 STORE REGISTERS 1-8 IN SAVE AREA
GEN1 H4 SET ACTIVE BIT X'01' IN PIB FLAG BYTE
H5 LOAD ADDRESS OF BACKGROUND PROGRAM PIB INTO REG 10
2
J1 GET ADDRESS OF BACKGROUND PIB IN REG 10
J2 STORE REGISTERS 1-8 IN SAVE AREA
GEN2 J3 SET UP REG 11 AS BASE REG WITH 4096
1
2
JA B1
PDINTR
K3 RETURN TO LINK ADDRESS IN REG 9 *K4
*K4 REGISTER 9 POINTS TO INTRTN--CHART CJ-A1.

Chart CA. $$A$SUP1 - SGTCHS Macro, Channel Scheduler with MPS
Refer to Chart 04.

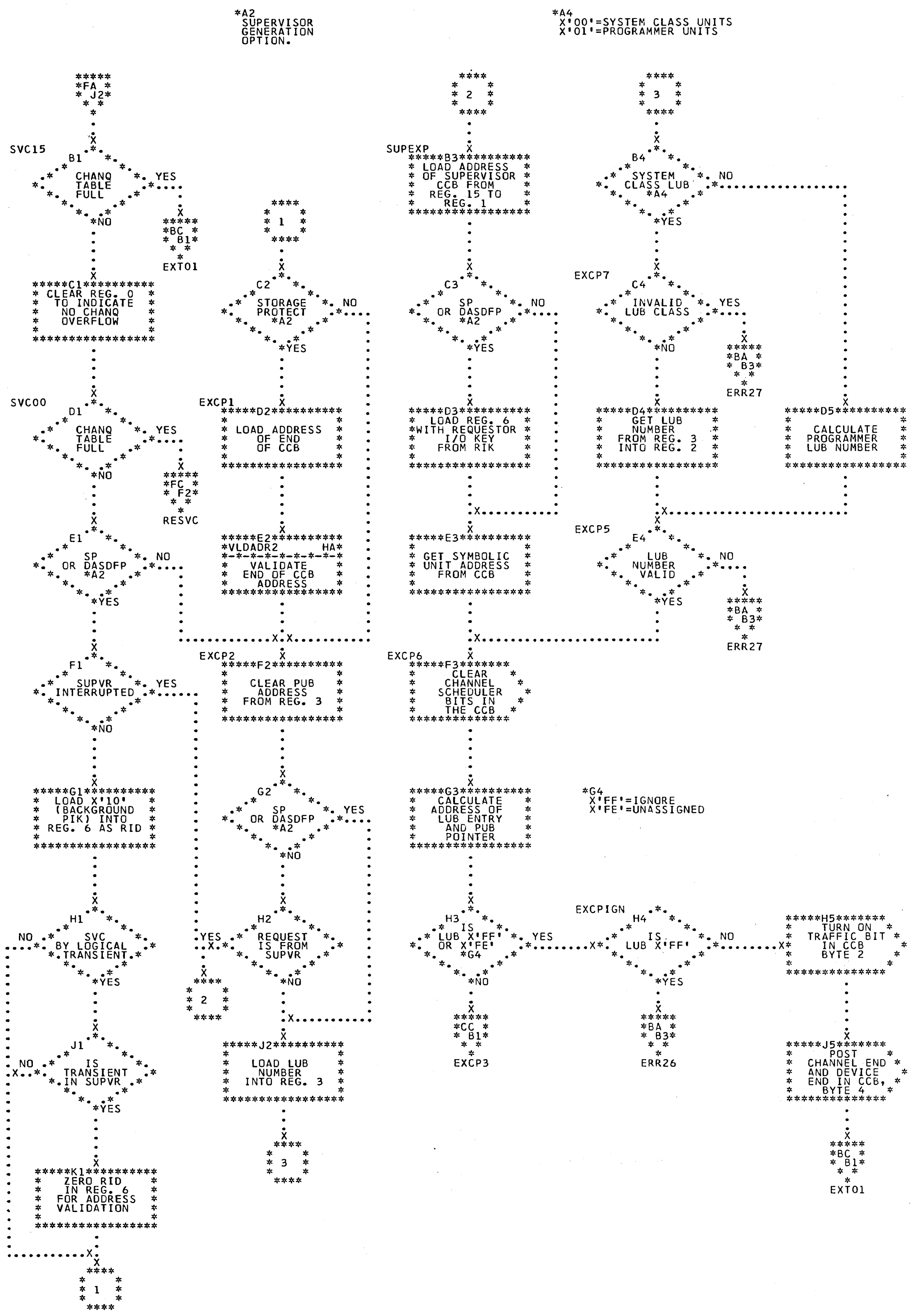

SVC 15
FA J2
SVC15 B1 CHANQ TABLE FULL
YES
NO
BC B1
EXT01
C1 CLEAR REG 0 TO INDICATE NO CHANQ OVERFLOW
FA G1
SVC 0
SVC00 D1 CHANQ TABLE FULL
YES
NO
FC F2
RESVC
K1
SVC25,SVC27
E1 LOAD PIK FROM COMMUNICATION REGION INTO REG 6
F1 SUPVR INTERRUPTED
YES
NO
1
COMPCLC G1 SVC BY LOGICAL TRANSIENT
NO
YES
H1 IS TRANSIENT IN SUPVR
NO
YES
J1 ZERO RID IN REG 6 FOR ADDRESS VALIDATION
2
1
SUPEXP B2 LOAD ADDRESS OF SUPERVISOR CCB FROM REG. 15 TO REG. 1
C2 LOAD REG 6 WITH REQUESTOR I/O KEY FROM RIK
D2 GET ABSOLUTE LUB NUMBER FROM CCB
E2 SUBTRACT NUMBER OF BACKGROUND LUBS
F2 IS RESULT NEGATIVE
YES
NO
G2 SUBTRACT NUMBER OF FOREGROUND 2 LUBS
H2 IS RESULT NEGATIVE
NO
YES
RADLDS J2 RE-ADD THE NUMBER THAT WAS JUST SUBTRACTED
3
2
EXCP1 B3 LOAD ADDRESS OF END OF CCB
C3 VLDADR2 HA VALIDATE END OF CCB ADDRESS
EXCP4 D3 CLEAR PUB ADDRESS FROM REG 3 AND GET LUB INDEX
E3 SYSTEM CLASS LUB K3
NO
YES
EXCP7 F3 INVALID LUB CLASS
YES
NO
G3 SET LOGICAL UNIT ID AND CALCULATE LUB INDEX FACTOR
EXCP10 H3 VALID CCB
NO
YES
BA B3
ERR27
3
B4 STORE RELATIVE LUB NUMBER, RELOAD THE ABSOLUTE LUB NUMBER
F4 GET NUMBER OF FIRST LUB ASSIGNED TO INTERRUPTED PROGRAM
EXCP8 G4 CALCULATE PROGRAMMER LUB NUMBER
EXCP5 H4 LUB NUMBER VALID
NO
YES
4
4
EXCP6 C5 CLEAR CHANNEL SCHEDULER BITS IN THE CCB
D5 CALCULATE ADDRESS OF LUB ENTRY AND PUB POINTER
E5 IS LUB X'FF' OR X'FE' K5
NO
YES
CC B1
EXCP3
EXCPIGN F5 IS LUB X'FF'
YES
NO
BA B3
ERR26
G5 TURN ON TRAFFIC BIT IN CCB, BYTE 2
H5 POST CHANNEL END AND DEVICE END IN CCB, BYTE 4
BC B1
EXT01

*K3
X'00' = SYSTEM CLASS UNITS
X'01' = PROGRAMMER CLASS UNITS

*K5
X'FF'= IGNORE
X'FE'= UNASSIGNED

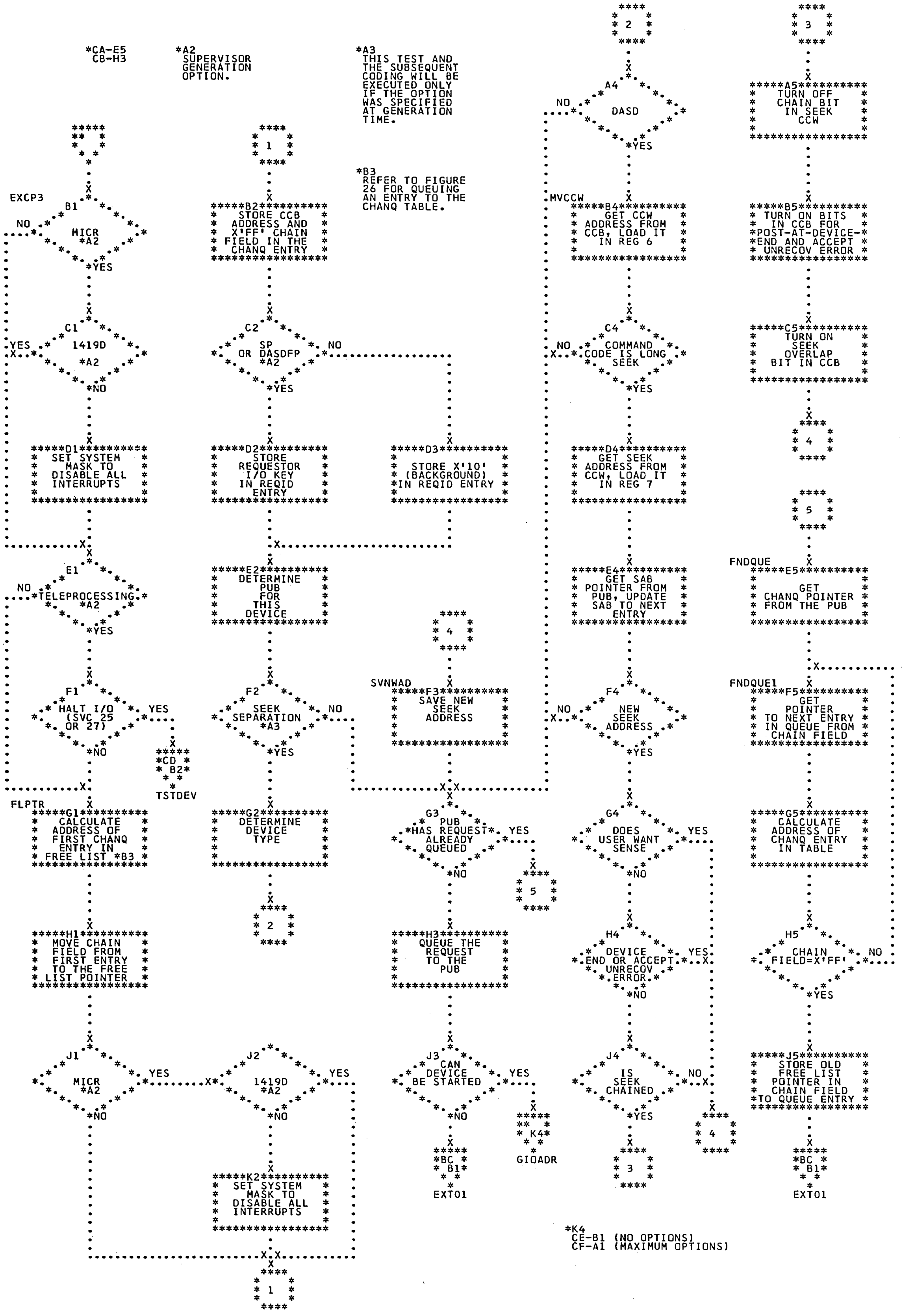
*K1
LABELS APPEAR WITH
TP OPTIONS ONLY.
ENTRY IS FROM FA-J2.

Chart CB. $$A$SUP1 - SGTCHS Macro, Channel Scheduler without MPS
Refer to Chart 04.



*A2 SUPERVISOR GENERATION OPTION.
*A4 X'00'=SYSTEM CLASS UNITS
X'01'=PROGRAMMER UNITS
FA J2
SVC15
B1 CHANQ TABLE FULL
YES
NO
BC B1
EXT01
C1 CLEAR REG. 0 TO INDICATE NO CHANQ OVERFLOW
SVC00
D1 CHANQ TABLE FULL
YES
NO
FC F2
RESVC
E1 SP OR DASDFP *A2
NO
YES
F1 SUPVR INTERRUPTED
YES
NO
G1 LOAD X'10' (BACKGROUND PIK) INTO REG. 6 AS RID
H1 SVC BY LOGICAL TRANSIENT
NO
YES
J1 IS TRANSIENT IN SUPVR
NO
YES
K1 ZERO RID IN REG. 6 FOR ADDRESS VALIDATION
1
1
C2 STORAGE PROTECT *A2
NO
YES
EXCP1
D2 LOAD ADDRESS OF END OF CCB
E2 VLDADR2 HA VALIDATE END OF CCB ADDRESS
EXCP2
F2 CLEAR PUB ADDRESS FROM REG. 3
G2 SP OR DASDFP *A2
YES
NO
H2 REQUEST IS FROM SUPVR
YES
NO
2
J2 LOAD LUB NUMBER INTO REG. 3
3
2
SUPEXP
B3 LOAD ADDRESS OF SUPERVISOR CCB FROM REG. 15 TO REG. 1
C3 SP OR DASDFP *A2
NO
YES
D3 LOAD REG. 6 WITH REQUESTOR I/O KEY FROM RIK
E3 GET SYMBOLIC UNIT ADDRESS FROM CCB
EXCP6
F3 CLEAR CHANNEL SCHEDULER BITS IN THE CCB
G3 CALCULATE ADDRESS OF LUB ENTRY AND PUB POINTER
H3 IS LUB X'FF' OR X'FE' *G4
YES
NO
CC B1
EXCP3
3
B4 SYSTEM CLASS LUB *A4
NO
YES
EXCP7
C4 INVALID LUB CLASS
YES
NO
BA B3
ERR27
D4 GET LUB NUMBER FROM REG. 3 INTO REG. 2
D5 CALCULATE PROGRAMMER LUB NUMBER
EXCP5
E4 LUB NUMBER VALID
NO
YES
BA B3
ERR27
*G4 X'FF'=IGNORE
X'FE'=UNASSIGNED
EXCPIGN
H4 IS LUB X'FF'
NO
YES
BA B3
ERR26
H5 TURN ON TRAFFIC BIT IN CCB BYTE 2
J5 POST CHANNEL END AND DEVICE END IN CCB, BYTE 4
BC B1
EXT01

Chart CC. $$A$SUP1 - SGTCHS Macro, Channel Scheduler (Part 1 of 2)
Refer to Chart 04.


*CA-E5 CB-H3
EXCP3
B1 MICR *A2
NO
YES
C1 1419D *A2
YES
NO
D1 SET SYSTEM MASK TO DISABLE ALL INTERRUPTS
E1 TELEPROCESSING *A2
NO
YES
F1 HALT I/O (SVC 25 OR 27)
YES
NO
*CD * B2*
TSTDEV
FLPTR
G1 CALCULATE ADDRESS OF FIRST CHANQ ENTRY IN FREE LIST *B3
H1 MOVE CHAIN FIELD FROM FIRST ENTRY TO THE FREE LIST POINTER
J1 MICR *A2
YES
NO
1
B2 STORE CCB ADDRESS AND X'FF' CHAIN FIELD IN THE CHANQ ENTRY
C2 SP OR DASDFP *A2
NO
YES
D2 STORE REQUESTOR I/O KEY IN REQID ENTRY
E2 DETERMINE PUB FOR THIS DEVICE
F2 SEEK SEPARATION *A3
NO
YES
G2 DETERMINE DEVICE TYPE
2
J2 1419D *A2
YES
NO
K2 SET SYSTEM MASK TO DISABLE ALL INTERRUPTS
1
D3 STORE X'10' (BACKGROUND) IN REQID ENTRY
4
SVNWAD
F3 SAVE NEW SEEK ADDRESS
G3 PUB HAS REQUEST ALREADY QUEUED
YES
NO
5
H3 QUEUE THE REQUEST TO THE PUB
J3 CAN DEVICE BE STARTED
YES
NO
*BC * B1*
EXT01
** K4*
GIOADR
2
A4 DASD
NO
YES
MVCCW
B4 GET CCW ADDRESS FROM CCB, LOAD IT IN REG 6
C4 COMMAND CODE IS LONG SEEK
NO
YES
D4 GET SEEK ADDRESS FROM CCW, LOAD IT IN REG 7
E4 GET SAB POINTER FROM PUB, UPDATE SAB TO NEXT ENTRY
F4 NEW SEEK ADDRESS
NO
YES
G4 DOES USER WANT SENSE
YES
NO
H4 DEVICE END OR ACCEPT UNRECOV ERROR
YES
NO
J4 IS SEEK CHAINED
NO
YES
3
4
3
A5 TURN OFF CHAIN BIT IN SEEK CCW
B5 TURN ON BITS IN CCB FOR POST-AT-DEVICE-END AND ACCEPT UNRECOV ERROR
C5 TURN ON SEEK OVERLAP BIT IN CCB
4
5
FNDQUE
E5 GET CHANQ POINTER FROM THE PUB
FNDQUE1
F5 GET POINTER TO NEXT ENTRY IN QUEUE FROM CHAIN FIELD
G5 CALCULATE ADDRESS OF CHANQ ENTRY IN TABLE
H5 CHAIN FIELD=X'FF'
NO
YES
J5 STORE OLD FREE LIST POINTER IN CHAIN FIELD TO QUEUE ENTRY
*BC * B1*
EXT01

*A2
SUPERVISOR GENERATION OPTION.

*A3
THIS TEST AND THE SUBSEQUENT CODING WILL BE EXECUTED ONLY IF THE OPTION WAS SPECIFIED AT GENERATION TIME.

*B3
REFER TO FIGURE 26 FOR QUEUING AN ENTRY TO THE CHANQ TABLE.

*K4
CE-B1 (NO OPTIONS)
CF-A1 (MAXIMUM OPTIONS)

Chart CD. $$A$SUP1 - SGTCHS Macro, Channel Scheduler (Part 2 of 2)
Refer to Chart 04.

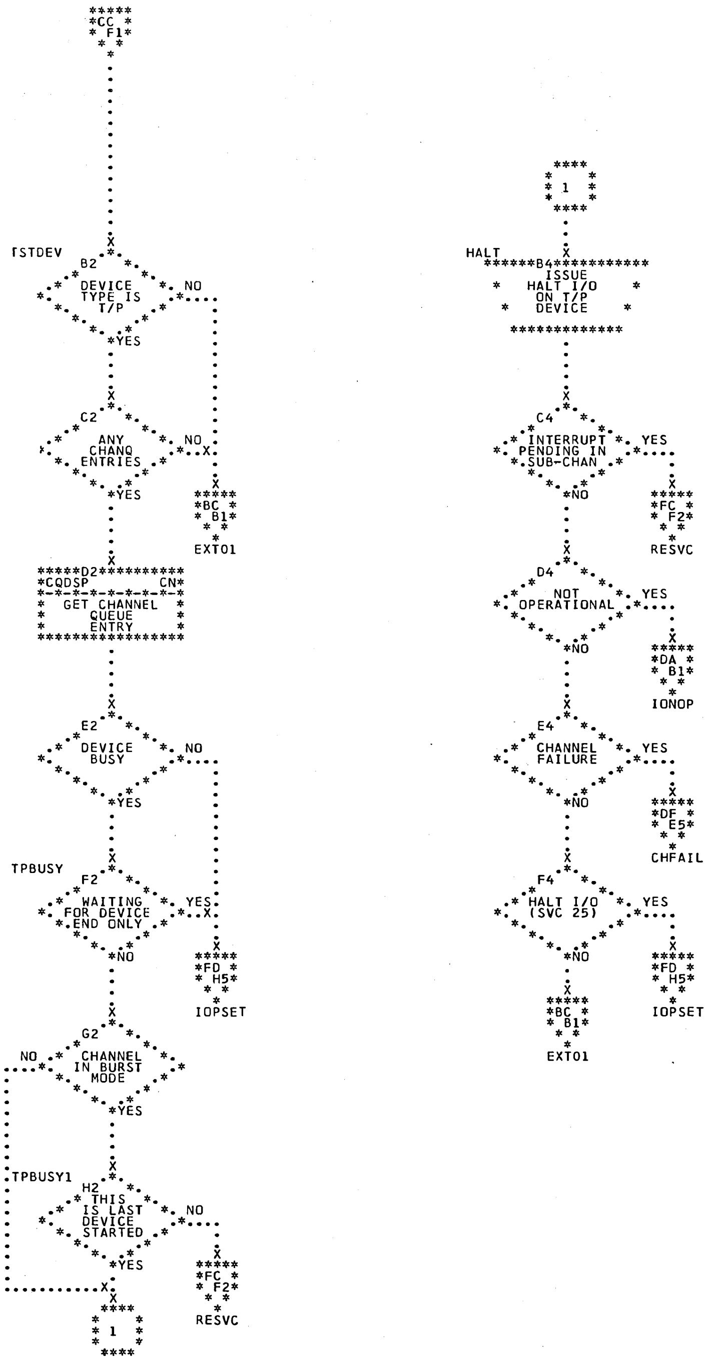
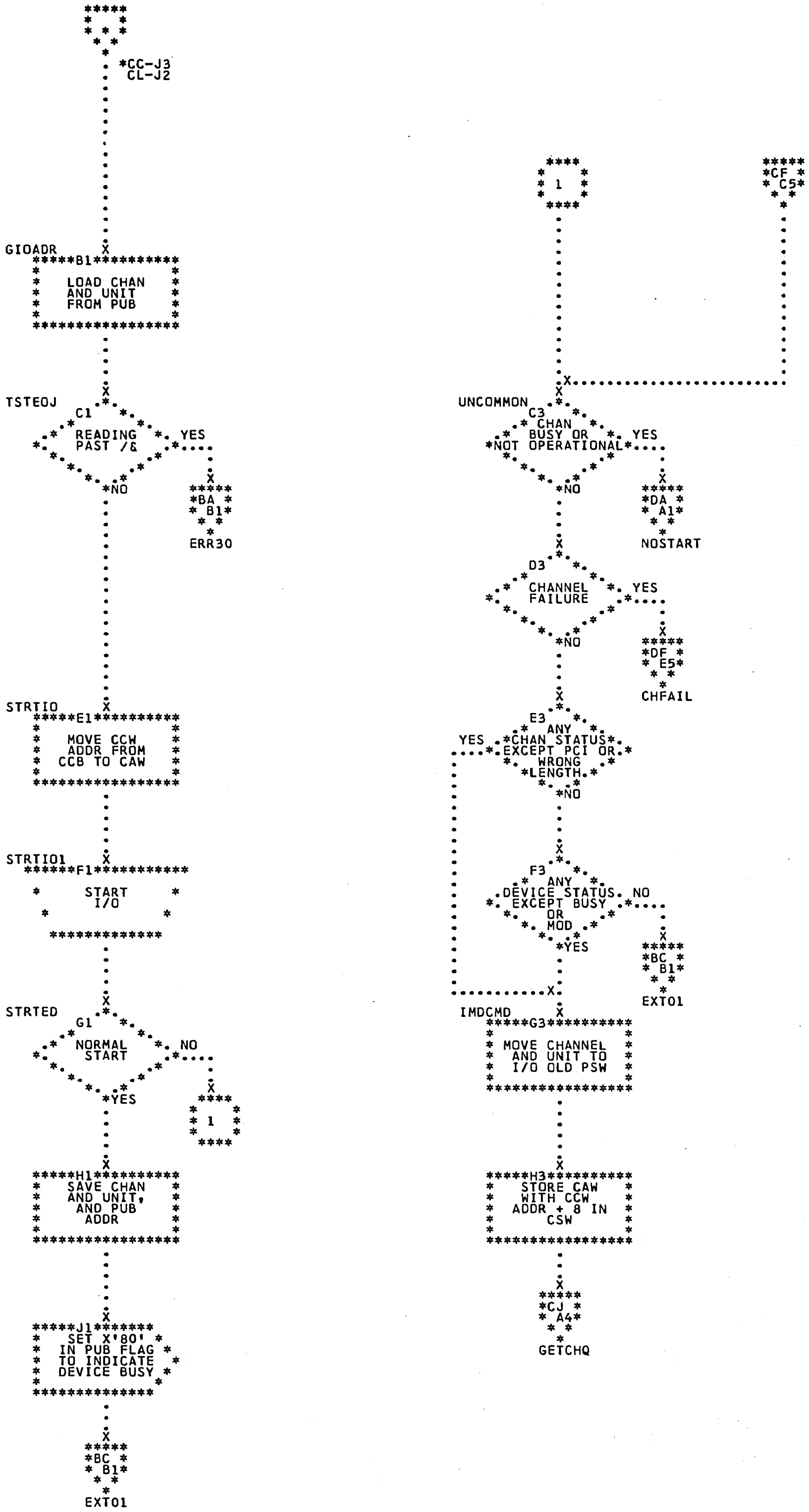
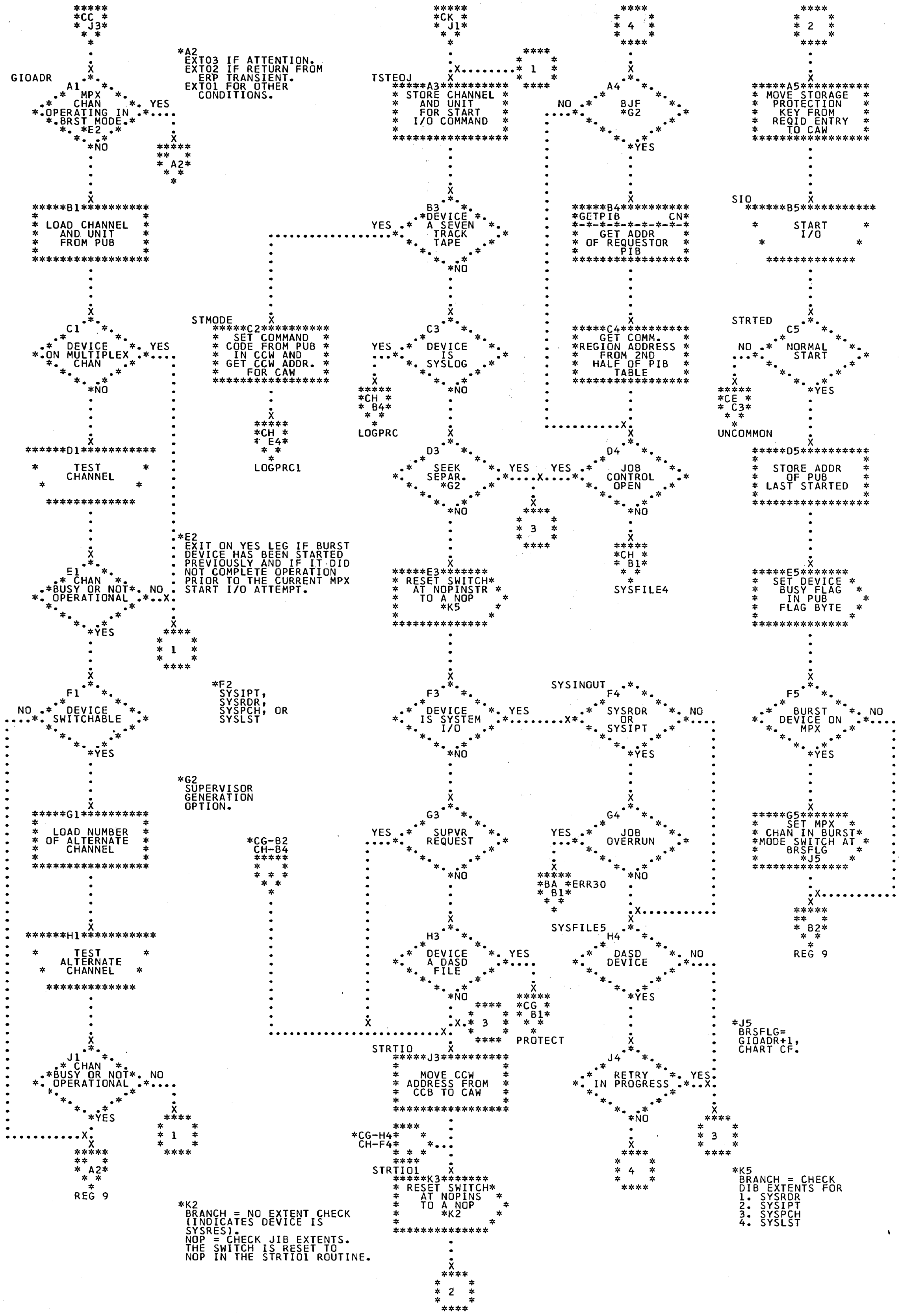


Chart CE. $$A$SUP1 - Supervisor, SGTCHS Macro, Start I/O -- No Options
Refer to Chart 04.



CC-J3
CL-J2
GIOADR
B1
LOAD CHAN AND UNIT FROM PUB
TSTEOJ
C1
READING PAST /&
YES
NO
BA B1
ERR30
STRTIO
E1
MOVE CCW ADDR FROM CCB TO CAW
STRTIO1
F1
START I/O
STRTED
G1
NORMAL START
NO
YES
1
H1
SAVE CHAN AND UNIT, AND PUB ADDR
J1
SET X'80' IN PUB FLAG TO INDICATE DEVICE BUSY
BC B1
EXT01
1
CF C5
UNCOMMON
C3
CHAN BUSY OR NOT OPERATIONAL
YES
NO
DA A1
NOSTART
D3
CHANNEL FAILURE
YES
NO
DF E5
CHFAIL
E3
ANY CHAN STATUS EXCEPT PCI OR WRONG LENGTH
YES
NO
F3
ANY DEVICE STATUS EXCEPT BUSY OR MOD
NO
YES
BC B1
EXT01
IMDCMD
G3
MOVE CHANNEL AND UNIT TO I/O OLD PSW
H3
STORE CAW WITH CCW ADDR + 8 IN CSW
CJ A4
GETCHQ

Chart CF. $$A$SUP1 - SGTCHS Macro, Start I/O -- Maximum Options (Part 1 of 3)
Refer to Chart 04.



*A2
EXT03 IF ATTENTION.
EXT02 IF RETURN FROM ERP TRANSIENT.
EXT01 FOR OTHER CONDITIONS.

*E2
EXIT ON YES LEG IF BURST DEVICE HAS BEEN STARTED PREVIOUSLY AND IF IT DID NOT COMPLETE OPERATION PRIOR TO THE CURRENT MPX START I/O ATTEMPT.

*F2
SYSIPT,
SYSRDR,
SYSPCH, OR
SYSLST

*G2
SUPERVISOR GENERATION OPTION.

*J5
BRSFLG=
GIOADR+1,
CHART CF.

*K2
BRANCH = NO EXTENT CHECK (INDICATES DEVICE IS SYSRES).
NOP = CHECK JIB EXTENTS. THE SWITCH IS RESET TO NOP IN THE STRTIO1 ROUTINE.

*K5
BRANCH = CHECK DIB EXTENTS FOR
1. SYSRDR
2. SYSIPT
3. SYSPCH
4. SYSLST

Chart CG. $$A$SUP1 - SGTCHS Macro, Start I/O -- Maximum Options (Part 2 of 3)
Refer to Chart 04.

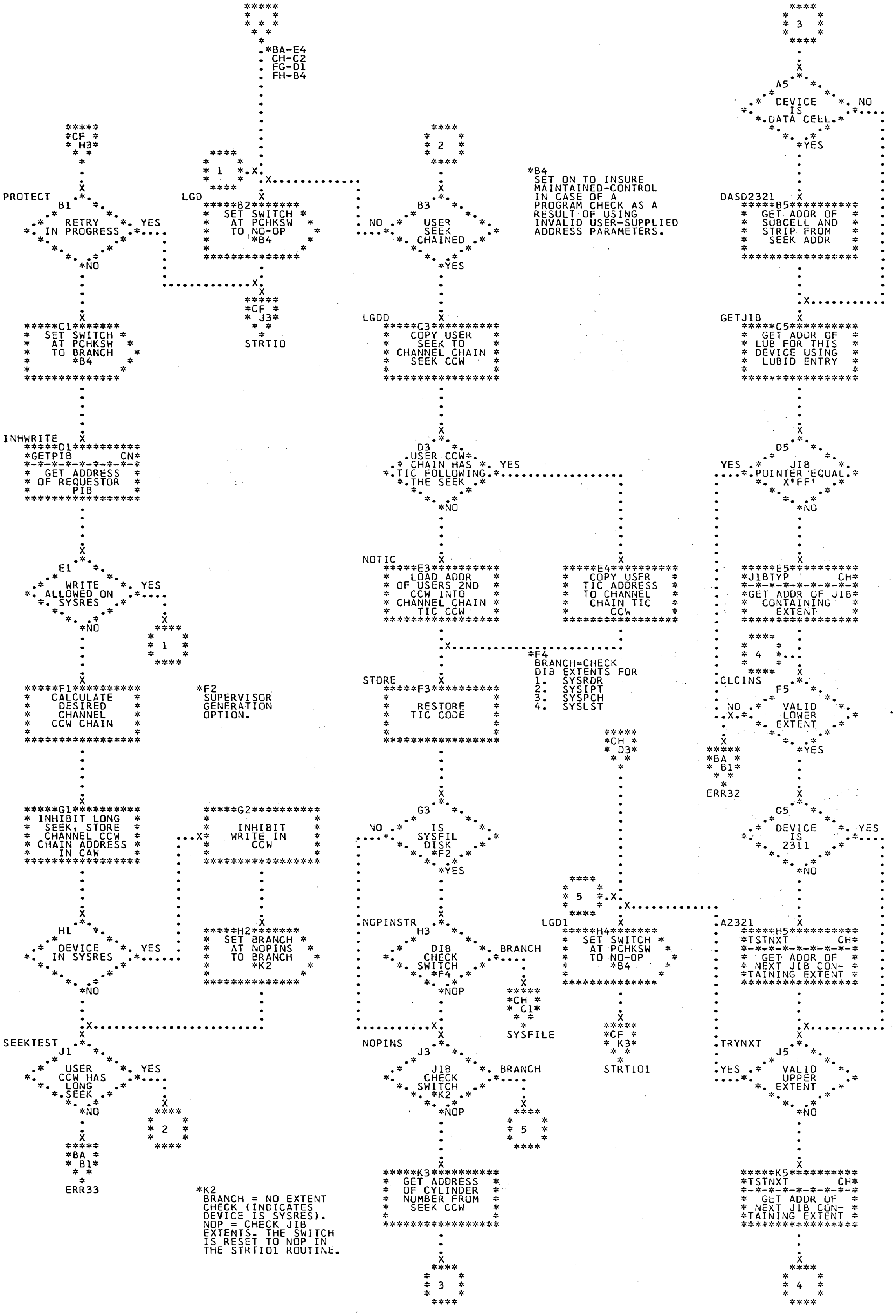
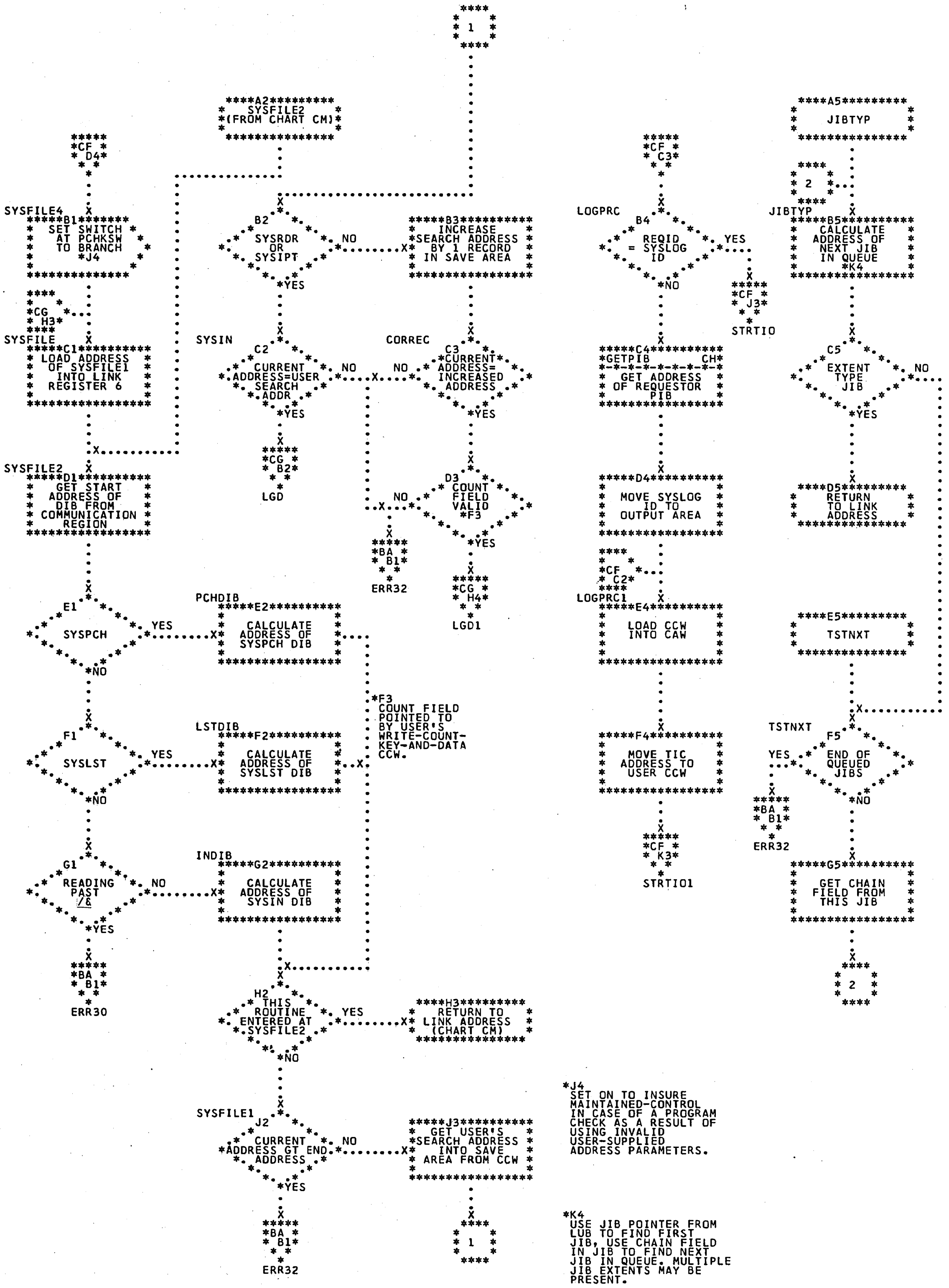

*BA-E4
CH-C2
FG-D1
FH-B4
3
A5
DEVICE IS DATA CELL
NO
YES
CF H3
PROTECT
B1
RETRY IN PROGRESS
YES
NO
1
LGD
B2
SET SWITCH AT PCHKSW TO NO-OP *B4
2
B3
USER SEEK CHAINED
NO
YES
*B4
SET ON TO INSURE MAINTAINED-CONTROL IN CASE OF A PROGRAM CHECK AS A RESULT OF USING INVALID USER-SUPPLIED ADDRESS PARAMETERS.
DASD2321
B5
GET ADDR OF SUBCELL AND STRIP FROM SEEK ADDR
CF J3
STRTIO
C1
SET SWITCH AT PCHKSW TO BRANCH *B4
LGDD
C3
COPY USER SEEK TO CHANNEL CHAIN SEEK CCW
GETJIB
C5
GET ADDR OF LUB FOR THIS DEVICE USING LUBID ENTRY
INHWRITE
D1
GETPIB CN
GET ADDRESS OF REQUESTOR PIB
D3
USER CCW CHAIN HAS TIC FOLLOWING THE SEEK
YES
NO
D5
JIB POINTER EQUAL X'FF'
YES
NO
E1
WRITE ALLOWED ON SYSRES
YES
NO
1
NOTIC
E3
LOAD ADDR OF USERS 2ND CCW INTO CHANNEL CHAIN TIC CCW
E4
COPY USER TIC ADDRESS TO CHANNEL CHAIN TIC CCW
E5
JIBTYP CH
GET ADDR OF JIB CONTAINING EXTENT
4
F1
CALCULATE DESIRED CHANNEL CCW CHAIN
*F2
SUPERVISOR GENERATION OPTION.
STORE
F3
RESTORE TIC CODE
*F4
BRANCH=CHECK DIB EXTENTS FOR
1. SYSRDR
2. SYSIPT
3. SYSPCH
4. SYSLST
CLCINS
F5
VALID LOWER EXTENT
NO
YES
CH D3
BA B1
ERR32
G1
INHIBIT LONG SEEK, STORE CHANNEL CCW CHAIN ADDRESS IN CAW
G2
INHIBIT WRITE IN CCW
G3
IS SYSFIL DISK *F2
NO
YES
G5
DEVICE IS 2311
YES
NO
5
H1
DEVICE IN SYSRES
YES
NO
H2
SET BRANCH AT NOPINS TO BRANCH *K2
NOPINSTR
H3
DIB CHECK SWITCH *F4
BRANCH
NOP
CH C1
SYSFILE
LGD1
H4
SET SWITCH AT PCHKSW TO NO-OP *B4
CF K3
STRTIO1
A2321
H5
TSTNXT CH
GET ADDR OF NEXT JIB CONTAINING EXTENT
SEEKTEST
J1
USER CCW HAS LONG SEEK
YES
NO
2
BA B1
ERR33
NOPINS
J3
JIB CHECK SWITCH *K2
BRANCH
NOP
5
TRYNXT
J5
VALID UPPER EXTENT
YES
NO
*K2
BRANCH = NO EXTENT CHECK (INDICATES DEVICE IS SYSRES). NOP = CHECK JIB EXTENTS. THE SWITCH IS RESET TO NOP IN THE STRTIO1 ROUTINE.
K3
GET ADDRESS OF CYLINDER NUMBER FROM SEEK CCW
3
K5
TSTNXT CH
GET ADDR OF NEXT JIB CONTAINING EXTENT
4

Chart CH. $$A$SUP1 - SGTCHS Macro, Start I/O -- Maximum Options (Part 3 of 3)
Refer to Chart 04.

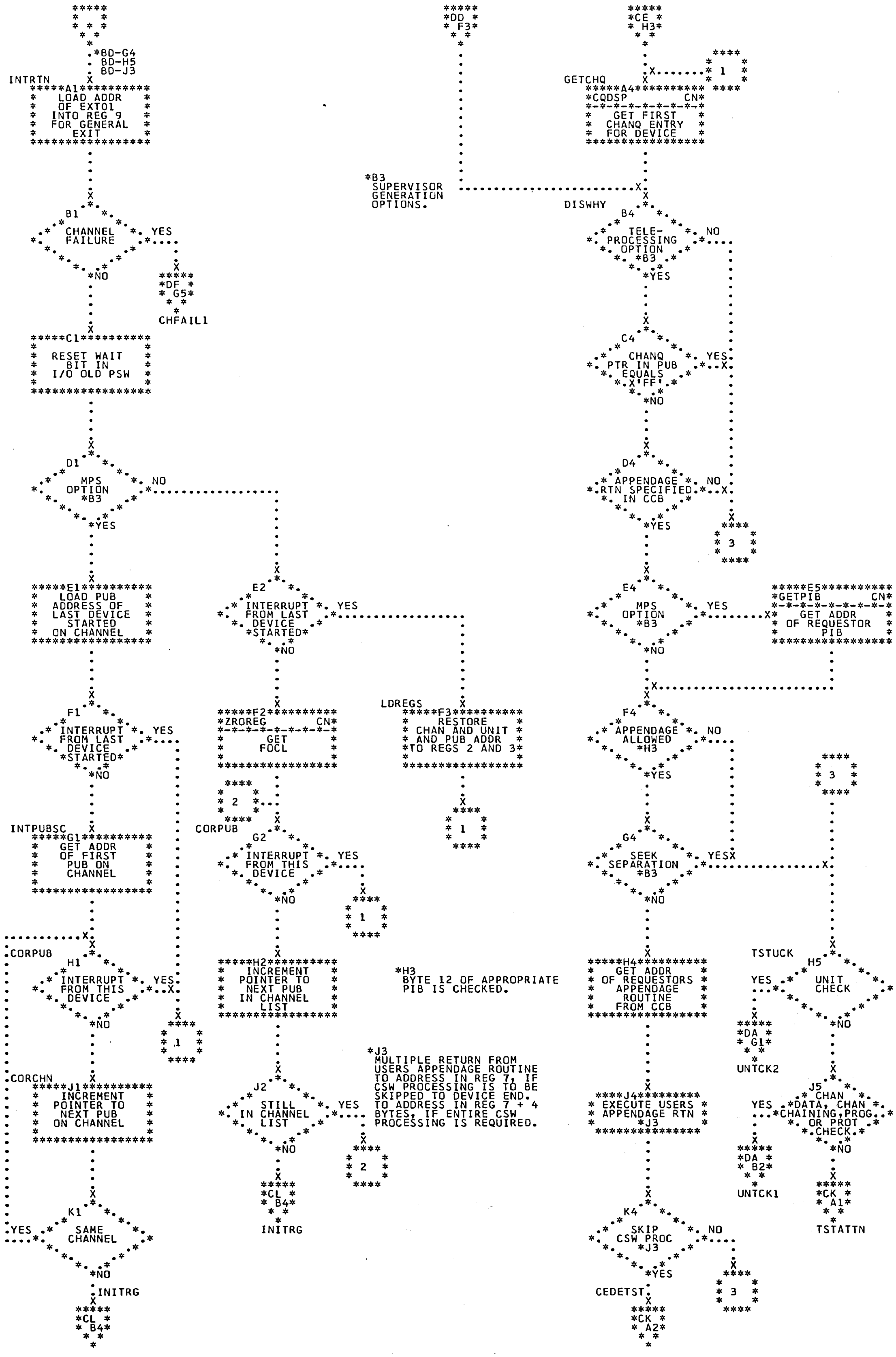

1
A2 SYSFILE2 (FROM CHART CM)
A5 JIBTYP
CF D4
SYSFILE4 B1 SET SWITCH AT PCHKSW TO BRANCH *J4
B2 SYSRDR OR SYSIPT
NO
YES
B3 INCREASE SEARCH ADDRESS BY 1 RECORD IN SAVE AREA
CF C3
LOGPRC B4 REQID = SYSLOG ID
YES
NO
CF J3 STRTIO
2
JIBTYP B5 CALCULATE ADDRESS OF NEXT JIB IN QUEUE *K4
CG H3
SYSFILE C1 LOAD ADDRESS OF SYSFILE1 INTO LINK REGISTER 6
SYSIN C2 CURRENT ADDRESS=USER SEARCH ADDR
NO
YES
CG B2 LGD
CORREC C3 CURRENT ADDRESS= INCREASED ADDRESS
NO
YES
C4 GETPIB CH GET ADDRESS OF REQUESTOR PIB
C5 EXTENT TYPE JIB
NO
YES
SYSFILE2 D1 GET START ADDRESS OF DIB FROM COMMUNICATION REGION
D3 COUNT FIELD VALID *F3
NO
YES
BA B1 ERR32
CG H4 LGD1
D4 MOVE SYSLOG ID TO OUTPUT AREA
D5 RETURN TO LINK ADDRESS
CF C2
E1 SYSPCH
YES
NO
PCHDIB E2 CALCULATE ADDRESS OF SYSPCH DIB
LOGPRC1 E4 LOAD CCW INTO CAW
E5 TSTNXT
F1 SYSLST
YES
NO
LSTDIB F2 CALCULATE ADDRESS OF SYSLST DIB
F4 MOVE TIC ADDRESS TO USER CCW
TSTNXT F5 END OF QUEUED JIBS
YES
NO
BA B1 ERR32
CF K3 STRTIO1
G1 READING PAST /&
NO
YES
BA B1 ERR30
INDIB G2 CALCULATE ADDRESS OF SYSIN DIB
G5 GET CHAIN FIELD FROM THIS JIB
2
H2 THIS ROUTINE ENTERED AT SYSFILE2
YES
NO
H3 RETURN TO LINK ADDRESS (CHART CM)
SYSFILE1 J2 CURRENT ADDRESS GT END ADDRESS
NO
YES
J3 GET USER'S SEARCH ADDRESS INTO SAVE AREA FROM CCW
BA B1 ERR32
1

*F3
COUNT FIELD POINTED TO BY USER'S WRITE-COUNT-KEY-AND-DATA CCW.

*J4
SET ON TO INSURE MAINTAINED-CONTROL IN CASE OF A PROGRAM CHECK AS A RESULT OF USING INVALID USER-SUPPLIED ADDRESS PARAMETERS.

*K4
USE JIB POINTER FROM LUB TO FIND FIRST JIB, USE CHAIN FIELD IN JIB TO FIND NEXT JIB IN QUEUE. MULTIPLE JIB EXTENTS MAY BE PRESENT.

Chart CJ. $$A$SUP1 - SGTCHS Macro, I/O Interrupt (Part 1 of 4)
Refer to Chart 04.


BD-G4
BD-H5
BD-J3
INTRTN
A1
LOAD ADDR OF EXT01 INTO REG 9 FOR GENERAL EXIT
B1
CHANNEL FAILURE
YES
NO
DF G5
CHFAIL1
C1
RESET WAIT BIT IN I/O OLD PSW
D1
MPS OPTION *B3
NO
YES
E1
LOAD PUB ADDRESS OF LAST DEVICE STARTED ON CHANNEL
F1
INTERRUPT FROM LAST DEVICE *STARTED*
YES
NO
INTPUBSC
G1
GET ADDR OF FIRST PUB ON CHANNEL
CORPUB
H1
INTERRUPT FROM THIS DEVICE
YES
NO
1
CORCHN
J1
INCREMENT POINTER TO NEXT PUB ON CHANNEL
K1
SAME CHANNEL
YES
NO
INITRG
CL B4
DD F3
E2
INTERRUPT FROM LAST DEVICE *STARTED*
YES
NO
F2
ZROREG CN
GET FOCL
2
CORPUB
G2
INTERRUPT FROM THIS DEVICE
YES
NO
1
H2
INCREMENT POINTER TO NEXT PUB IN CHANNEL LIST
J2
STILL IN CHANNEL LIST
YES
NO
2
CL B4
INITRG
LDREGS
F3
RESTORE CHAN AND UNIT AND PUB ADDR TO REGS 2 AND 3
1
CE H3
1
GETCHQ
A4
CQDSP CN
GET FIRST CHANQ ENTRY FOR DEVICE
DISWHY
B4
TELE-PROCESSING OPTION *B3
NO
YES
C4
CHANQ PTR IN PUB EQUALS X'FF'
YES
NO
D4
APPENDAGE RTN SPECIFIED IN CCB
NO
YES
3
E4
MPS OPTION *B3
YES
NO
E5
GETPIB CN
GET ADDR OF REQUESTOR PIB
F4
APPENDAGE ALLOWED *H3
NO
YES
3
G4
SEEK SEPARATION *B3
YES
NO
H4
GET ADDR OF REQUESTORS APPENDAGE ROUTINE FROM CCB
J4
EXECUTE USERS APPENDAGE RTN *J3
K4
SKIP CSW PROC *J3
NO
YES
3
CEDETST
CK A2
TSTUCK
H5
UNIT CHECK
YES
NO
DA G1
UNTCK2
J5
CHAN DATA, CHAN CHAINING, PROG. OR PROT CHECK
YES
NO
DA B2
UNTCK1
CK A1
TSTATTN

*B3
SUPERVISOR GENERATION OPTIONS.

*H3
BYTE 12 OF APPROPRIATE PIB IS CHECKED.

*J3
MULTIPLE RETURN FROM USERS APPENDAGE ROUTINE TO ADDRESS IN REG 7, IF CSW PROCESSING IS TO BE SKIPPED TO DEVICE END. TO ADDRESS IN REG 7 + 4 BYTES, IF ENTIRE CSW PROCESSING IS REQUIRED.

Chart CK. $$A$SUP1 - SGTCHS Macro, I/O Interrupt (Part 2 of 4)
Refer to Chart 04.

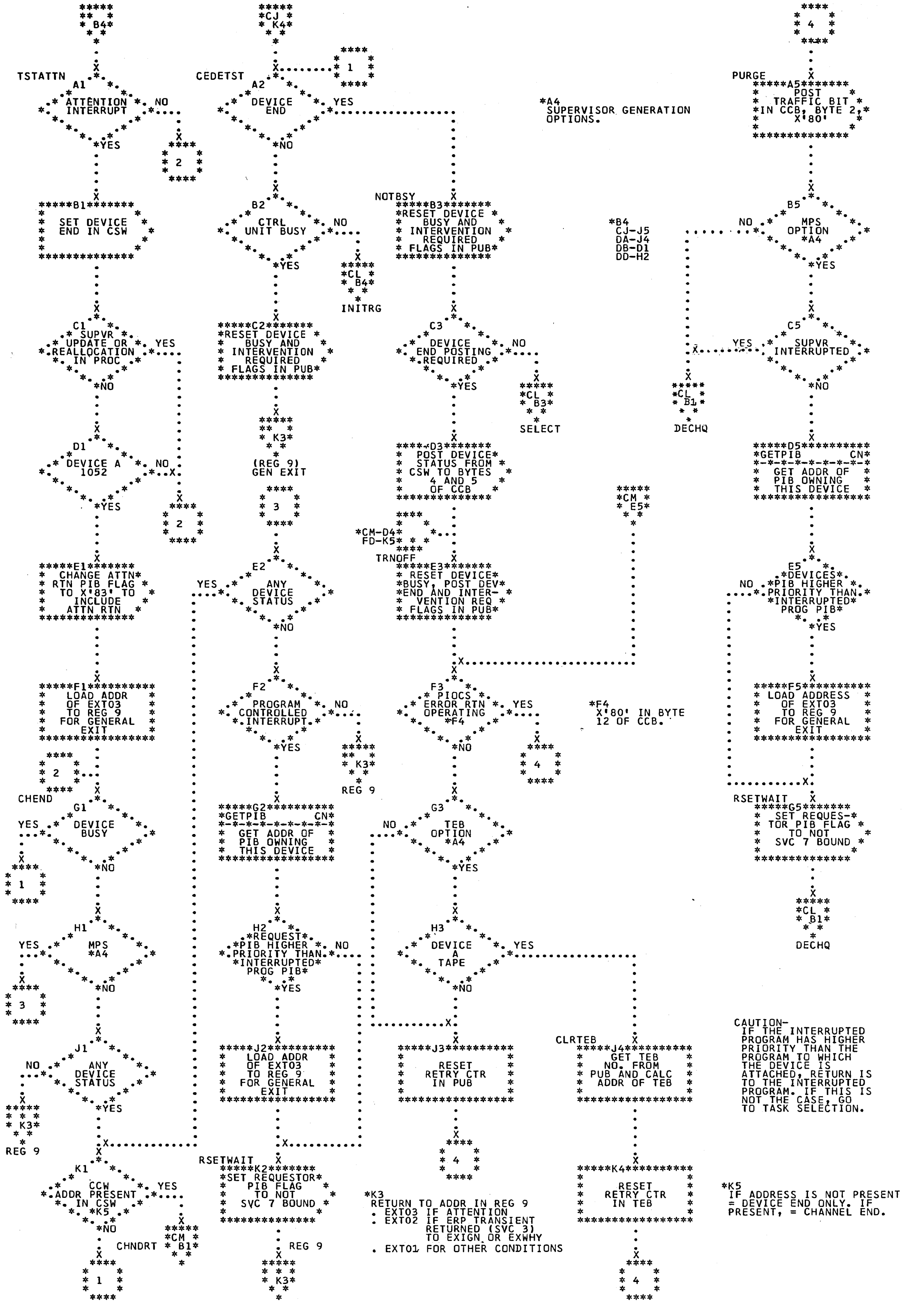
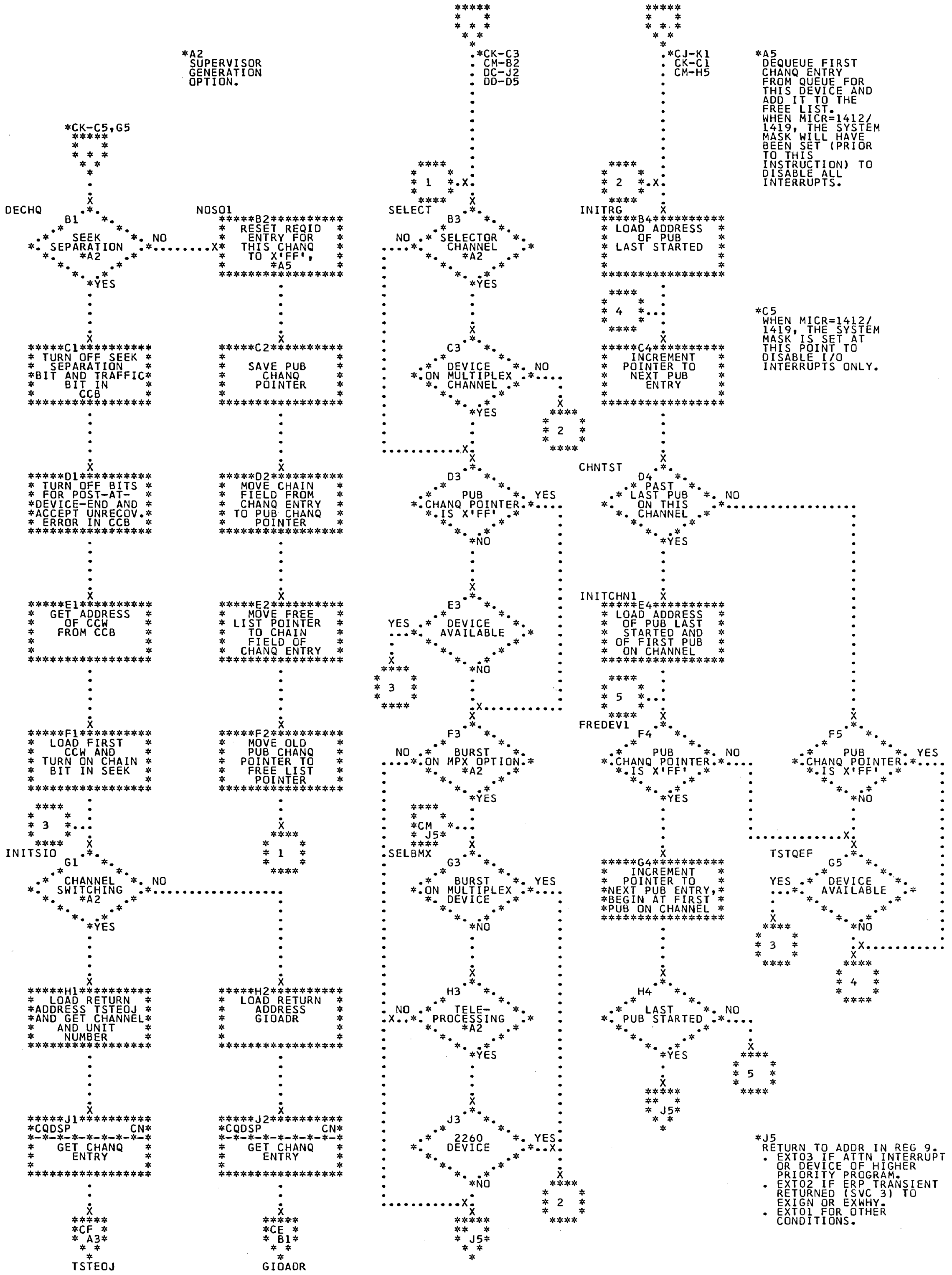

B4
TSTATTN
A1
ATTENTION INTERRUPT
NO
YES
2
B1
SET DEVICE END IN CSW
C1
SUPVR UPDATE OR REALLOCATION IN PROC
YES
NO
D1
DEVICE A 1052
NO
YES
2
E1
CHANGE ATTN RTN PIB FLAG TO X'83' TO INCLUDE ATTN RTN
F1
LOAD ADDR OF EXT03 TO REG 9 FOR GENERAL EXIT
2
CHEND
G1
DEVICE BUSY
YES
NO
1
H1
MPS *A4
YES
NO
3
J1
ANY DEVICE STATUS
NO
YES
K3
REG 9
K1
CCW ADDR PRESENT IN CSW *K5
YES
NO
CHNDRT
CM B1
1
CJ K4
CEDETST
1
A2
DEVICE END
YES
NO
B2
CTRL UNIT BUSY
NO
YES
CL B4
INITRG
C2
RESET DEVICE BUSY AND INTERVENTION REQUIRED FLAGS IN PUB
K3
(REG 9) GEN EXIT
3
E2
ANY DEVICE STATUS
YES
NO
F2
PROGRAM CONTROLLED INTERRUPT
NO
YES
K3
REG 9
G2
GETPIB CN
GET ADDR OF PIB OWNING THIS DEVICE
H2
REQUEST PIB HIGHER PRIORITY THAN INTERRUPTED PROG PIB
NO
YES
J2
LOAD ADDR OF EXT03 TO REG 9 FOR GENERAL EXIT
RSETWAIT
K2
SET REQUESTOR PIB FLAG TO NOT SVC 7 BOUND
REG 9
K3
NOTBSY
B3
RESET DEVICE BUSY AND INTERVENTION REQUIRED FLAGS IN PUB
C3
DEVICE END POSTING REQUIRED
NO
YES
CL B3
SELECT
D3
POST DEVICE STATUS FROM CSW TO BYTES 4 AND 5 OF CCB
CM-D4
FD-K5
TRNOFF
E3
RESET DEVICE BUSY, POST DEV END AND INTERVENTION REQ FLAGS IN PUB
CM E5
F3
PIOCS ERROR RTN OPERATING *F4
YES
NO
4
G3
TEB OPTION *A4
NO
YES
H3
DEVICE A TAPE
YES
NO
J3
RESET RETRY CTR IN PUB
4
CLRTEB
J4
GET TEB NO. FROM PUB AND CALC ADDR OF TEB
K4
RESET RETRY CTR IN TEB
4
*A4
SUPERVISOR GENERATION OPTIONS.
*B4
CJ-J5
DA-J4
DB-D1
DD-H2
CL B1
DECHQ
*F4
X'80' IN BYTE 12 OF CCB.
4
PURGE
A5
POST TRAFFIC BIT IN CCB, BYTE 2, X'80'
B5
MPS OPTION *A4
NO
YES
C5
SUPVR INTERRUPTED
YES
NO
D5
GETPIB CN
GET ADDR OF PIB OWNING THIS DEVICE
E5
DEVICES PIB HIGHER PRIORITY THAN INTERRUPTED PROG PIB
NO
YES
F5
LOAD ADDRESS OF EXT03 TO REG 9 FOR GENERAL EXIT
RSETWAIT
G5
SET REQUESTOR PIB FLAG TO NOT SVC 7 BOUND
CL B1
DECHQ
CAUTION-
IF THE INTERRUPTED PROGRAM HAS HIGHER PRIORITY THAN THE PROGRAM TO WHICH THE DEVICE IS ATTACHED, RETURN IS TO THE INTERRUPTED PROGRAM. IF THIS IS NOT THE CASE, GO TO TASK SELECTION.
*K3
RETURN TO ADDR IN REG 9
. EXT03 IF ATTENTION
. EXT02 IF ERP TRANSIENT RETURNED (SVC 3) TO EXIGN OR EXWHY
. EXT01 FOR OTHER CONDITIONS
*K5
IF ADDRESS IS NOT PRESENT = DEVICE END ONLY. IF PRESENT, = CHANNEL END.

Chart CL. $$A$SUP1 - SGTCHS Macro, I/O Interrupt (Part 3 of 4)
Refer to Chart 04.


CK-C5,G5
DECHQ
B1 SEEK SEPARATION *A2
NO
YES
NOSO1
B2 RESET REQID ENTRY FOR THIS CHANQ TO X'FF'. *A5
C1 TURN OFF SEEK SEPARATION BIT AND TRAFFIC BIT IN CCB
C2 SAVE PUB CHANQ POINTER
D1 TURN OFF BITS FOR POST-AT-DEVICE-END AND ACCEPT UNRECOV. ERROR IN CCB
D2 MOVE CHAIN FIELD FROM CHANQ ENTRY TO PUB CHANQ POINTER
E1 GET ADDRESS OF CCW FROM CCB
E2 MOVE FREE LIST POINTER TO CHAIN FIELD OF CHANQ ENTRY
F1 LOAD FIRST CCW AND TURN ON CHAIN BIT IN SEEK
F2 MOVE OLD PUB CHANQ POINTER TO FREE LIST POINTER
3
1
INITSIO
G1 CHANNEL SWITCHING *A2
NO
YES
H1 LOAD RETURN ADDRESS TSTEOJ AND GET CHANNEL AND UNIT NUMBER
H2 LOAD RETURN ADDRESS GIOADR
J1 CQDSP CN GET CHANQ ENTRY
J2 CQDSP CN GET CHANQ ENTRY
CF A3
TSTEOJ
CE B1
GIOADR
CK-C3 CM-B2 DC-J2 DD-D5
1
SELECT
B3 SELECTOR CHANNEL *A2
NO
YES
C3 ON MULTIPLEX CHANNEL
NO
YES
2
D3 PUB CHANQ POINTER IS X'FF'
YES
NO
E3 DEVICE AVAILABLE
YES
NO
3
F3 BURST ON MPX OPTION *A2
NO
YES
CM J5
SELBMX
G3 BURST ON MULTIPLEX DEVICE
YES
NO
H3 TELE-PROCESSING *A2
NO
YES
J3 2260 DEVICE
YES
NO
2
J5
CJ-K1 CK-C1 CM-H5
2
INITRG
B4 LOAD ADDRESS OF PUB LAST STARTED
4
C4 INCREMENT POINTER TO NEXT PUB ENTRY
CHNTST
D4 PAST LAST PUB ON THIS CHANNEL
NO
YES
INITCHN1
E4 LOAD ADDRESS OF PUB LAST STARTED AND OF FIRST PUB ON CHANNEL
5
FREDEV1
F4 PUB CHANQ POINTER IS X'FF'
NO
YES
G4 INCREMENT POINTER TO NEXT PUB ENTRY, BEGIN AT FIRST PUB ON CHANNEL
H4 LAST PUB STARTED
NO
YES
5
J5
F5 PUB CHANQ POINTER IS X'FF'
YES
NO
TSTQEF
G5 DEVICE AVAILABLE
YES
NO
3
4

*A2
SUPERVISOR GENERATION OPTION.

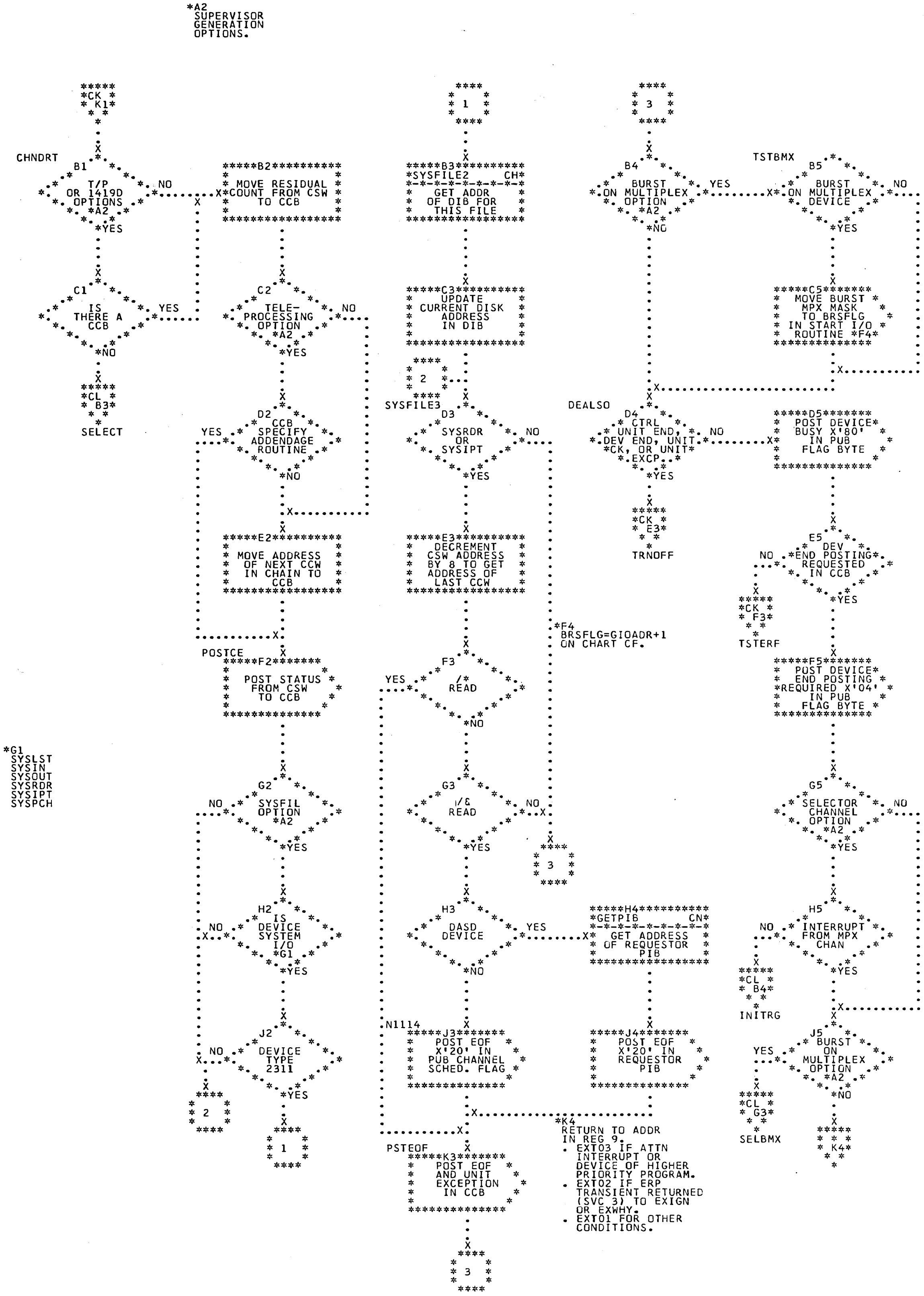
*A5
DEQUEUE FIRST CHANQ ENTRY FROM QUEUE FOR THIS DEVICE AND ADD IT TO THE FREE LIST. WHEN MICR=1412/1419, THE SYSTEM MASK WILL HAVE BEEN SET (PRIOR TO THIS INSTRUCTION) TO DISABLE ALL INTERRUPTS.

*C5
WHEN MICR=1412/1419, THE SYSTEM MASK IS SET AT THIS POINT TO DISABLE I/O INTERRUPTS ONLY.

*J5
RETURN TO ADDR IN REG 9.

- EXT03 IF ATTN INTERRUPT OR DEVICE OF HIGHER PRIORITY PROGRAM.
- EXT02 IF ERP TRANSIENT RETURNED (SVC 3) TO EXIGN OR EXWHY.
- EXT01 FOR OTHER CONDITIONS.

Chart CM. $$A$SUP1 - SGTCHS Macro, I/O Interrupt (Part 4 of 4)
Refer to Chart 04.



*A2
SUPERVISOR
GENERATION
OPTIONS.
CK K1
CHNDRT
B1
T/P OR 1419D OPTIONS *A2
NO
YES
C1
IS THERE A CCB
YES
NO
CL B3
SELECT
*G1
SYSLST
SYSIN
SYSOUT
SYSRDR
SYSIPT
SYSPCH
B2
MOVE RESIDUAL COUNT FROM CSW TO CCB
C2
TELE-PROCESSING OPTION *A2
NO
YES
D2
CCB SPECIFY ADDENDAGE ROUTINE
YES
NO
E2
MOVE ADDRESS OF NEXT CCW IN CHAIN TO CCB
POSTCE
F2
POST STATUS FROM CSW TO CCB
G2
SYSFIL OPTION *A2
NO
YES
H2
IS DEVICE SYSTEM I/O *G1
NO
YES
J2
DEVICE TYPE 2311
NO
YES
2
1
1
B3
SYSFILE2 CH
GET ADDR OF DIB FOR THIS FILE
C3
UPDATE CURRENT DISK ADDRESS IN DIB
2
SYSFILE3
D3
SYSRDR OR SYSIPT
NO
YES
E3
DECREMENT CSW ADDRESS BY 8 TO GET ADDRESS OF LAST CCW
*F4
BRSFLG=GIOADR+1
ON CHART CF.
F3
/* READ
YES
NO
G3
/& READ
NO
YES
3
H3
DASD DEVICE
YES
NO
N1114
J3
POST EOF X'20' IN PUB CHANNEL SCHED. FLAG
PSTEOF
K3
POST EOF AND UNIT EXCEPTION IN CCB
3
H4
GETPIB CN
GET ADDRESS OF REQUESTOR PIB
J4
POST EOF X'20' IN REQUESTOR PIB
*K4
RETURN TO ADDR IN REG 9.
. EXT03 IF ATTN INTERRUPT OR DEVICE OF HIGHER PRIORITY PROGRAM.
. EXT02 IF ERP TRANSIENT RETURNED (SVC 3) TO EXIGN OR EXWHY.
. EXT01 FOR OTHER CONDITIONS.
3
B4
BURST ON MULTIPLEX OPTION *A2
YES
NO
DEALSO
D4
CTRL UNIT END, DEV END, UNIT CK, OR UNIT EXCP
NO
YES
CK E3
TRNOFF
TSTBMX
B5
BURST ON MULTIPLEX DEVICE
NO
YES
C5
MOVE BURST MPX MASK TO BRSFLG IN START I/O ROUTINE *F4
D5
POST DEVICE BUSY X'80' IN PUB FLAG BYTE
E5
DEV END POSTING REQUESTED IN CCB
NO
YES
CK F3
TSTERF
F5
POST DEVICE END POSTING REQUIRED X'04' IN PUB FLAG BYTE
G5
SELECTOR CHANNEL OPTION *A2
NO
YES
H5
INTERRUPT FROM MPX CHAN
NO
YES
CL B4
INITRG
J5
BURST ON MULTIPLEX OPTION *A2
YES
NO
CL G3
SELBMX
K4

Chart CN. $$A$SUP1 - SGTCHS Macro, I/O Interrupt Subroutines
Refer to Chart 04.

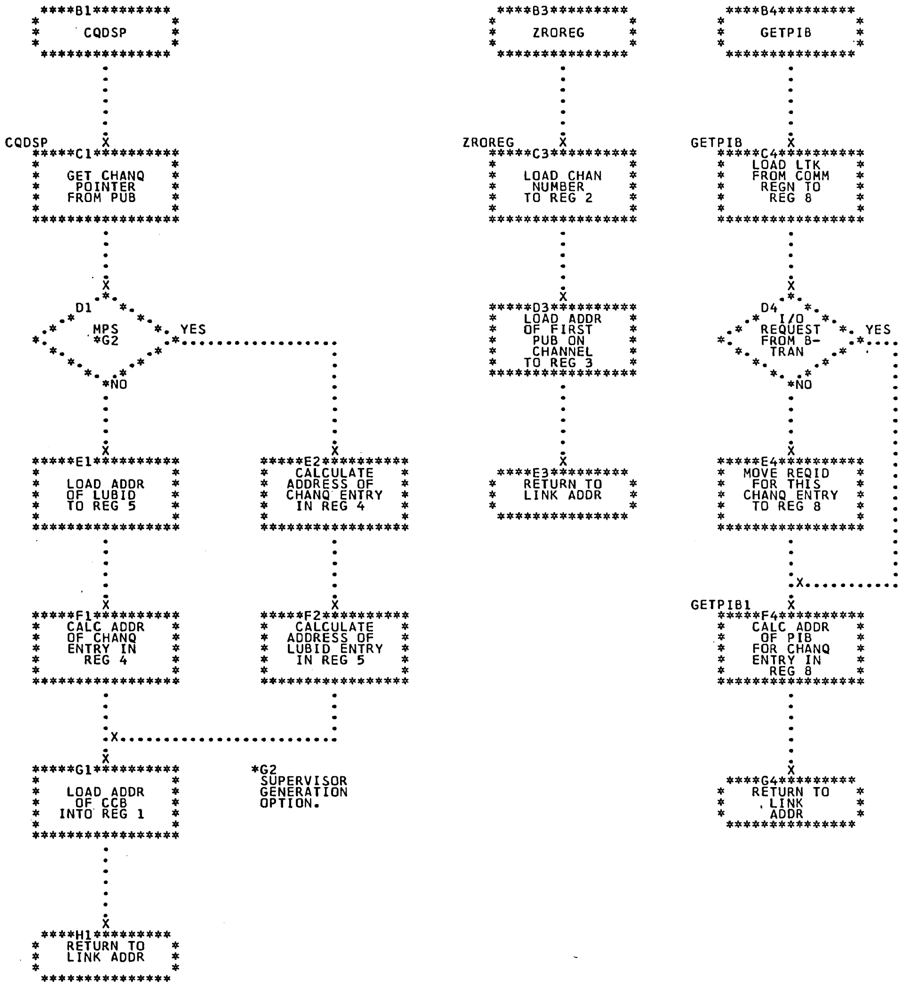

B1
CQDSP
CQDSP
C1
GET CHANQ POINTER FROM PUB
D1
MPS *G2
YES
NO
E1
LOAD ADDR OF LUBID TO REG 5
E2
CALCULATE ADDRESS OF CHANQ ENTRY IN REG 4
F1
CALC ADDR OF CHANQ ENTRY IN REG 4
F2
CALCULATE ADDRESS OF LUBID ENTRY IN REG 5
G1
LOAD ADDR OF CCB INTO REG 1
*G2 SUPERVISOR GENERATION OPTION.
H1
RETURN TO LINK ADDR
B3
ZROREG
ZROREG
C3
LOAD CHAN NUMBER TO REG 2
D3
LOAD ADDR OF FIRST PUB ON CHANNEL TO REG 3
E3
RETURN TO LINK ADDR
B4
GETPIB
GETPIB
C4
LOAD LTK FROM COMM REGN TO REG 8
D4
I/O REQUEST FROM B-TRAN
YES
NO
E4
MOVE REQID FOR THIS CHANQ ENTRY TO REG 8
GETPIB1
F4
CALC ADDR OF PIB FOR CHANQ ENTRY IN REG 8
G4
RETURN TO LINK ADDR

Chart DA. $$A$SUP1 - SGUNCK Macro, Unit Check Entries
Refer to Chart 06.

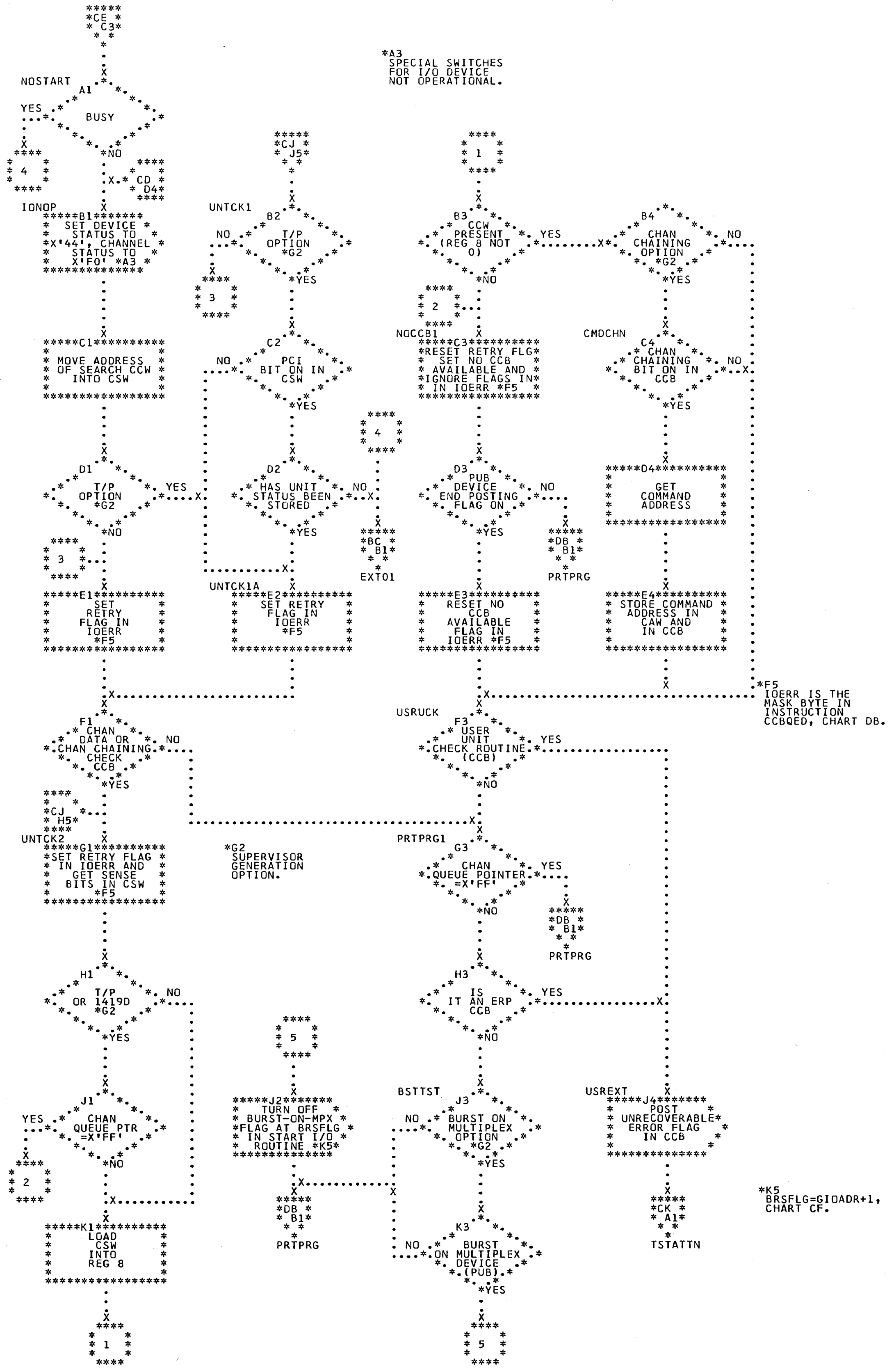

CE C3
NOSTART
A1 BUSY
YES
NO
4
CD D4
IONOP
B1 SET DEVICE STATUS TO X'44', CHANNEL STATUS TO X'F0' *A3
C1 MOVE ADDRESS OF SEARCH CCW INTO CSW
D1 T/P OPTION *G2
YES
NO
3
E1 SET RETRY FLAG IN IOERR *F5
F1 CHAN DATA OR CHAN CHAINING CHECK CCB
NO
YES
CJ H5
UNTCK2
G1 SET RETRY FLAG IN IOERR AND GET SENSE BITS IN CSW *F5
H1 T/P OR 1419D *G2
NO
YES
J1 CHAN QUEUE PTR =X'FF'
YES
NO
2
K1 LOAD CSW INTO REG 8
1
*A3 SPECIAL SWITCHES FOR I/O DEVICE NOT OPERATIONAL.
CJ J5
UNTCK1
B2 T/P OPTION *G2
NO
YES
3
C2 PCI BIT ON IN CSW
NO
YES
D2 HAS UNIT STATUS BEEN STORED
NO
YES
4
BC B1
EXT01
UNTCK1A
E2 SET RETRY FLAG IN IOERR *F5
*G2 SUPERVISOR GENERATION OPTION.
5
J2 TURN OFF BURST-ON-MPX FLAG AT BRSFLG IN START I/O ROUTINE *K5
DB B1
PRTPRG
1
B3 CCW PRESENT (REG 8 NOT 0)
YES
NO
2
NOCCB1
C3 RESET RETRY FLG SET NO CCB AVAILABLE AND IGNORE FLAGS IN IN IOERR *F5
D3 PUB DEVICE END POSTING FLAG ON
NO
YES
DB B1
PRTPRG
E3 RESET NO CCB AVAILABLE FLAG IN IOERR *F5
USRUCK
F3 USER UNIT CHECK ROUTINE (CCB)
YES
NO
PRTPRG1
G3 CHAN QUEUE POINTER =X'FF'
YES
NO
DB B1
PRTPRG
H3 IS IT AN ERP CCB
YES
NO
BSTTST
J3 BURST ON MULTIPLEX OPTION *G2
NO
YES
K3 BURST ON MULTIPLEX DEVICE (PUB)
NO
YES
5
B4 CHAN CHAINING OPTION *G2
NO
YES
CMDCHN
C4 CHAN CHAINING BIT ON IN CCB
NO
YES
D4 GET COMMAND ADDRESS
E4 STORE COMMAND ADDRESS IN CAW AND IN CCB
*F5 IOERR IS THE MASK BYTE IN INSTRUCTION CCBQED, CHART DB.
USREXT
J4 POST UNRECOVERABLE ERROR FLAG IN CCB
CK A1
TSTATTN
*K5 BRSFLG=GIOADR+1, CHART CF.

Chart DB. $$A$SUP1 - SGUNCK Macro, Build Error Queue Entry
Refer to Chart 06.

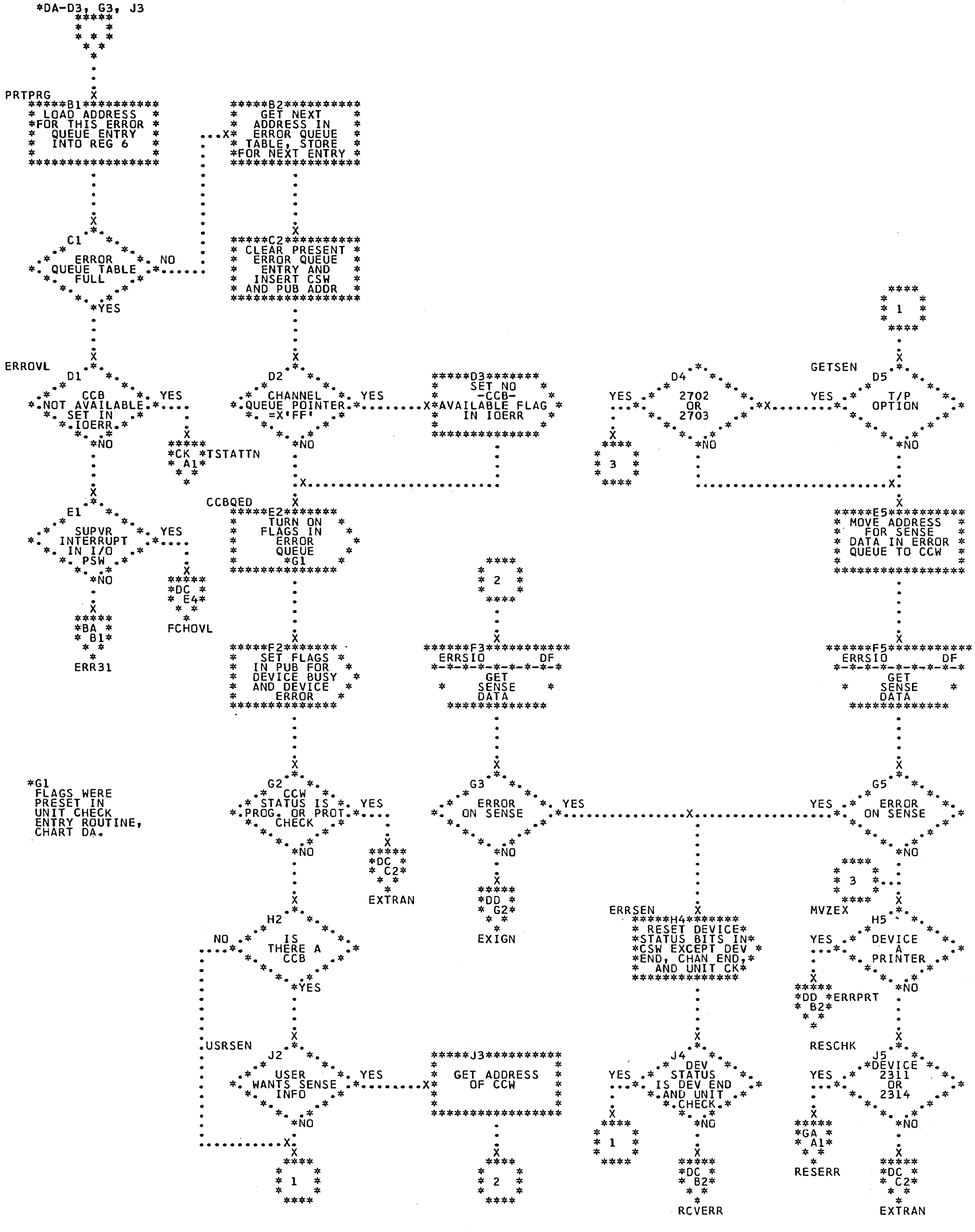

*DA-D3, G3, J3
PRTPRG
B1 LOAD ADDRESS FOR THIS ERROR QUEUE ENTRY INTO REG 6
B2 GET NEXT ADDRESS IN ERROR QUEUE TABLE, STORE FOR NEXT ENTRY
C1 ERROR QUEUE TABLE FULL
NO
YES
C2 CLEAR PRESENT ERROR QUEUE ENTRY AND INSERT CSW AND PUB ADDR
ERROVL
D1 CCB NOT AVAILABLE SET IN IOERR
YES
NO
CK A1 TSTATTN
D2 CHANNEL QUEUE POINTER =X'FF'
YES
NO
D3 SET NO -CCB- AVAILABLE FLAG IN IOERR
D4 2702 OR 2703
YES
NO
3
1
GETSEN
D5 T/P OPTION
YES
NO
E1 SUPVR INTERRUPT IN I/O PSW
YES
NO
DC E4 FCHOVL
BA B1 ERR31
CCBQED
E2 TURN ON FLAGS IN ERROR QUEUE *G1
E5 MOVE ADDRESS FOR SENSE DATA IN ERROR QUEUE TO CCW
2
F2 SET FLAGS IN PUB FOR DEVICE BUSY AND DEVICE ERROR
F3 ERRSIO DF GET SENSE DATA
F5 ERRSIO DF GET SENSE DATA
*G1 FLAGS WERE PRESET IN UNIT CHECK ENTRY ROUTINE, CHART DA.
G2 CCW STATUS IS PROG. OR PROT. CHECK
YES
NO
DC C2 EXTRAN
G3 ERROR ON SENSE
YES
NO
DD G2 EXIGN
G5 ERROR ON SENSE
YES
NO
3
H2 IS THERE A CCB
NO
YES
ERRSEN
H4 RESET DEVICE STATUS BITS IN CSW EXCEPT DEV END, CHAN END, AND UNIT CK
MVZEX
H5 DEVICE A PRINTER
YES
NO
DD B2 ERRPRT
USRSEN
J2 USER WANTS SENSE INFO
YES
NO
J3 GET ADDRESS OF CCW
J4 DEV STATUS IS DEV END AND UNIT CHECK
YES
NO
1
DC B2 RCVERR
RESCHK
J5 DEVICE 2311 OR 2314
YES
NO
GA A1 RESERR
DC C2 EXTRAN
1
2

Chart DC. $$A$SUP1 - SGUNCK Macro, Error Recovery Exits (Part 1 of 2)
Refer to Chart 06.

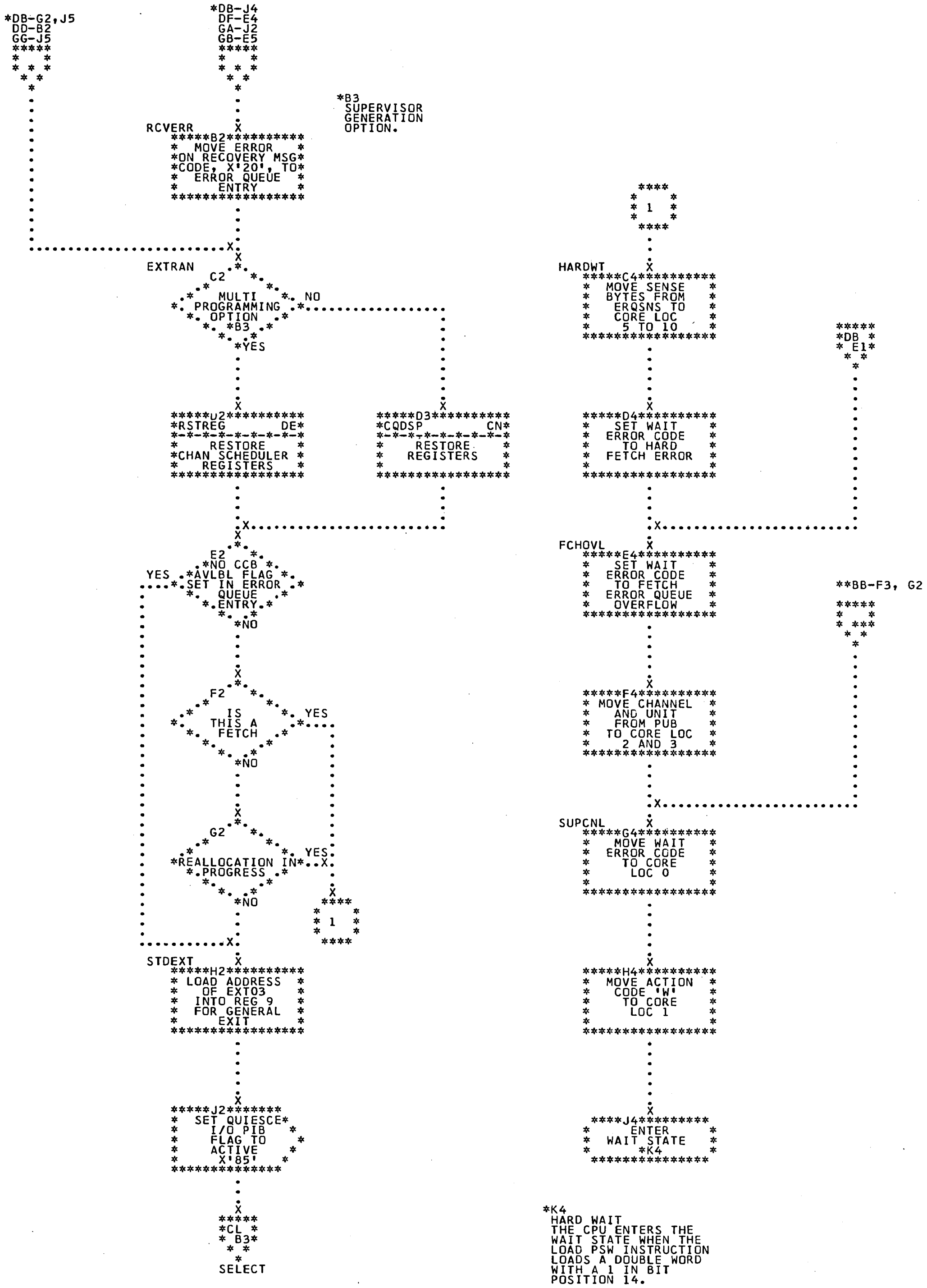


Chart DD. $$A$SUP1 - SGUNCK Macro, Error Recovery Exits (Part 2 of 2)
Refer to Chart 06.

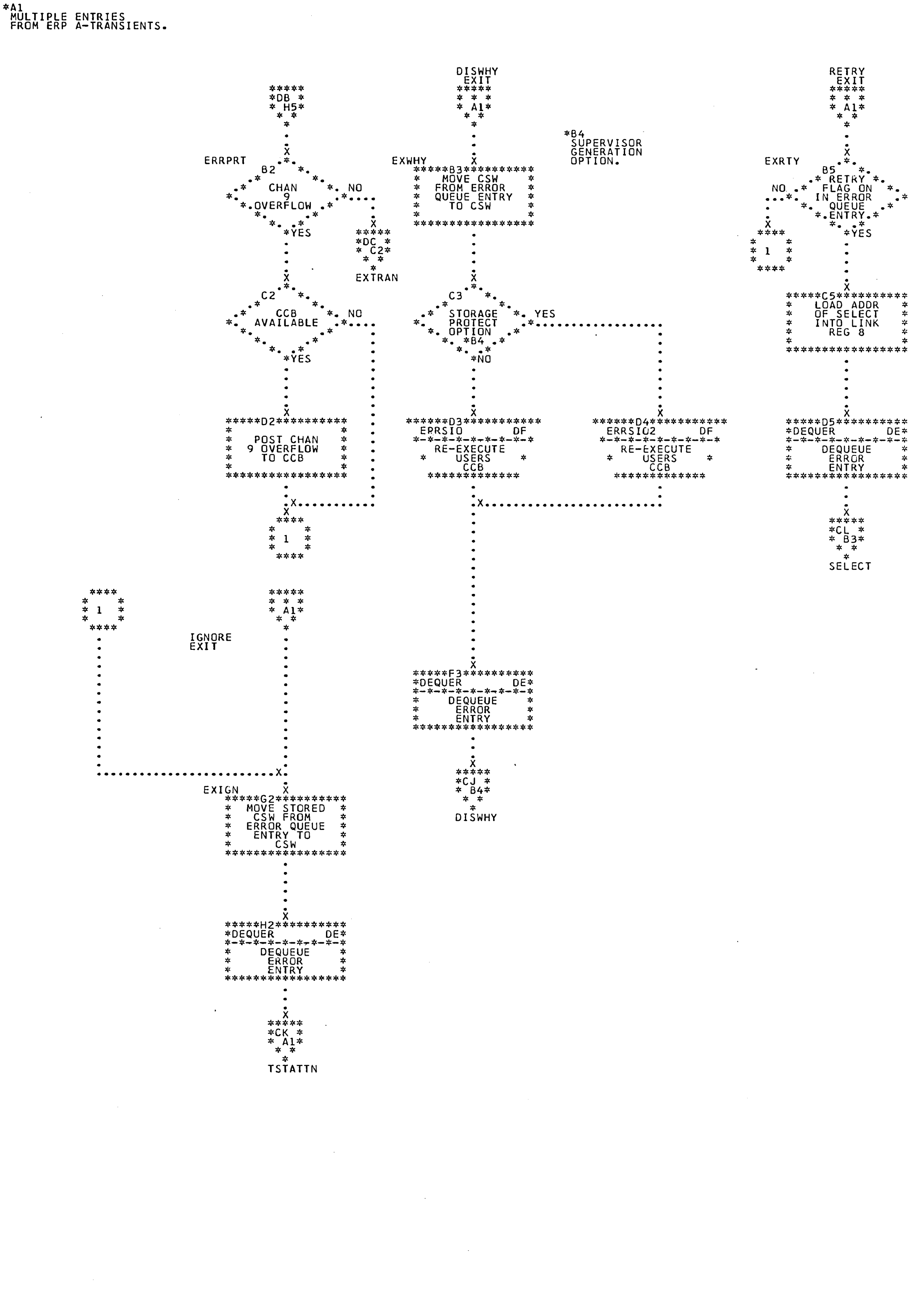


Chart DE. $$A$SUP1 - SGUNCK Macro, DEQUER and RSTREG Subroutines
Refer to Chart 06.

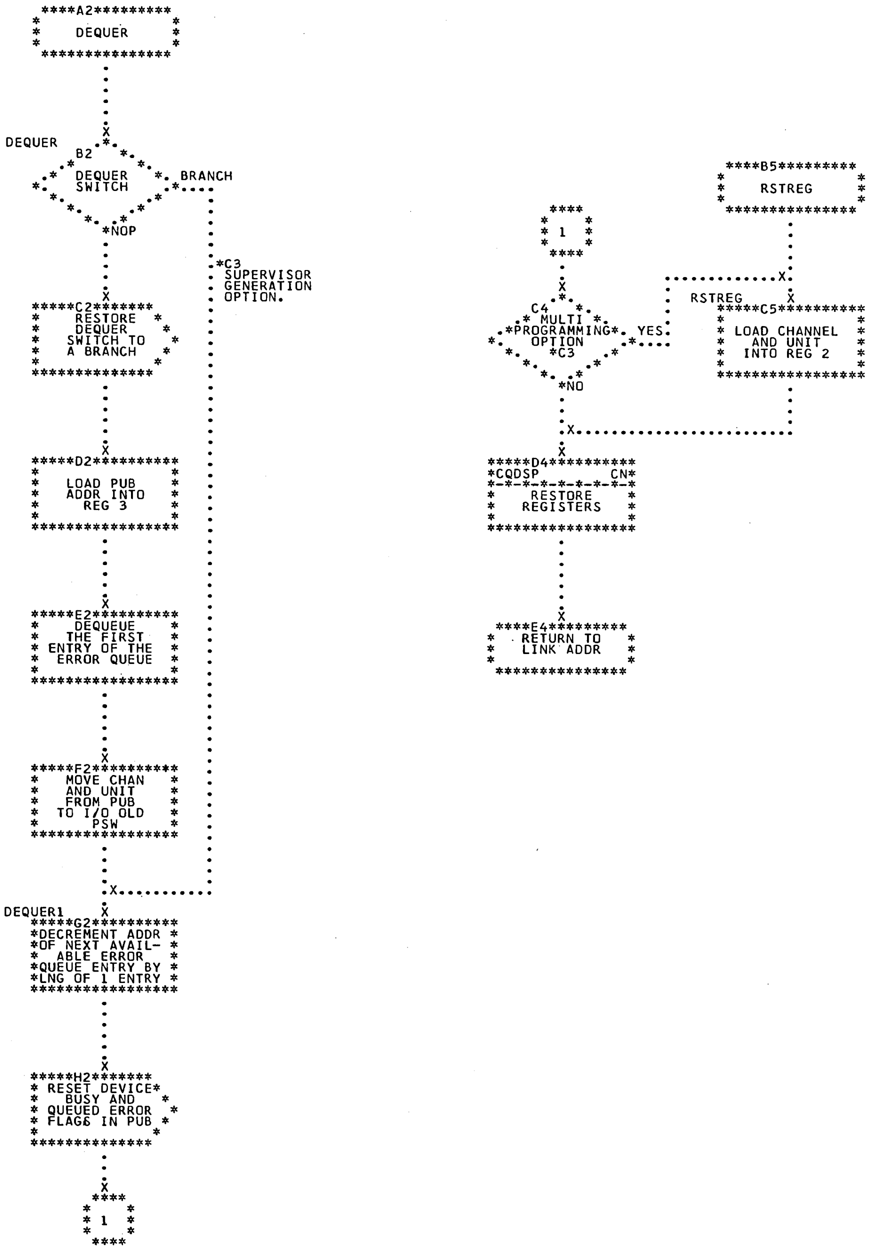
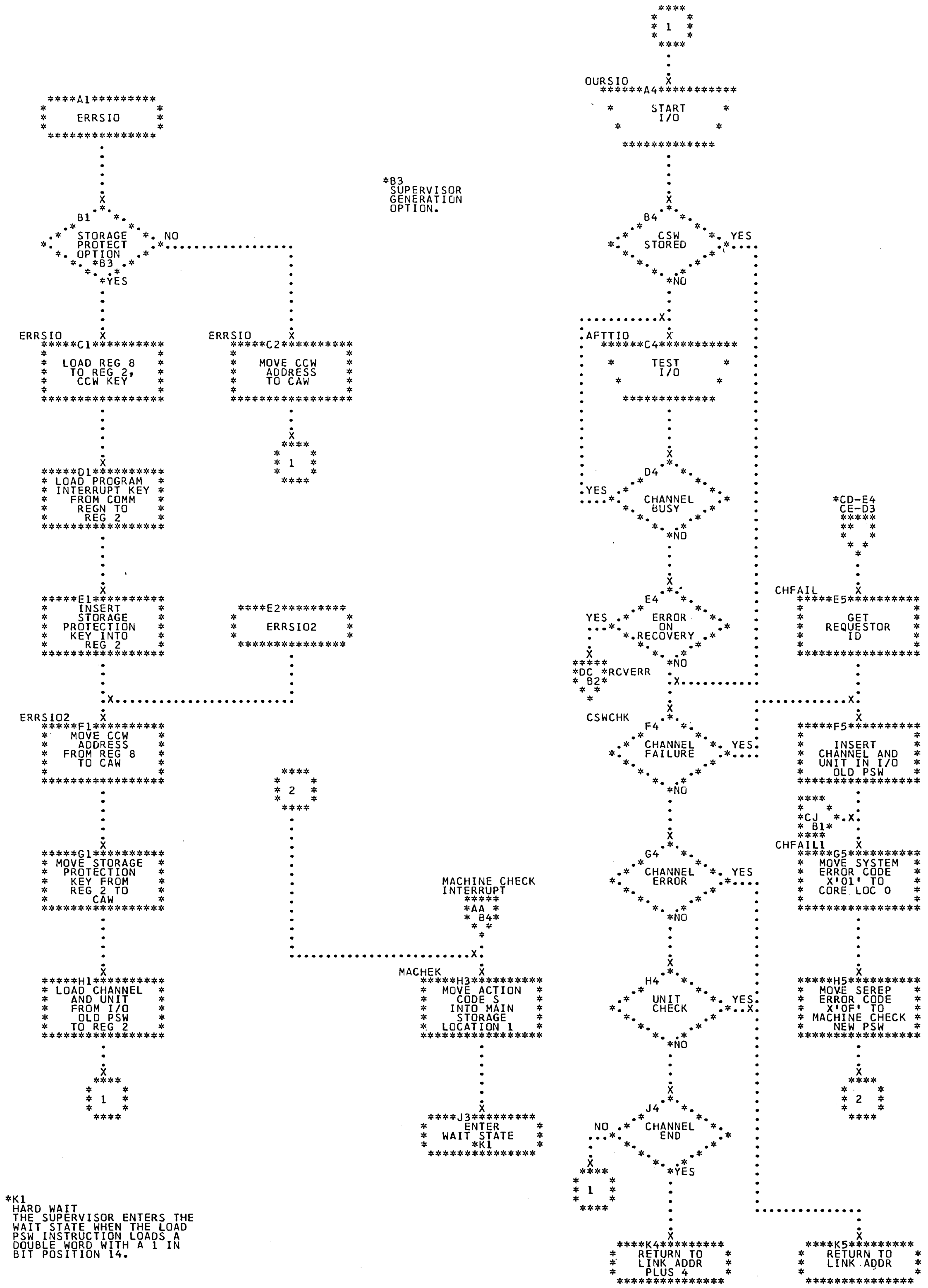

A2
DEQUER
DEQUER
B2
DEQUER SWITCH
BRANCH
NOP
C3
SUPERVISOR GENERATION OPTION.
C2
RESTORE DEQUER SWITCH TO A BRANCH
D2
LOAD PUB ADDR INTO REG 3
E2
DEQUEUE THE FIRST ENTRY OF THE ERROR QUEUE
F2
MOVE CHAN AND UNIT FROM PUB TO I/O OLD PSW
DEQUER1
G2
DECREMENT ADDR OF NEXT AVAIL-ABLE ERROR QUEUE ENTRY BY LNG OF 1 ENTRY
H2
RESET DEVICE BUSY AND QUEUED ERROR FLAGS IN PUB
1
B5
RSTREG
1
C4
MULTI PROGRAMMING OPTION
C3
YES
NO
RSTREG
C5
LOAD CHANNEL AND UNIT INTO REG 2
D4
CQDSP
CN
RESTORE REGISTERS
E4
RETURN TO LINK ADDR

Chart DF. $$A$SUP1 - SGUNCK Macro, Error Start I/O Subroutine
Refer to Chart 06.


1
OURSIO
A4
START I/O
A1
ERRSIO
*B3 SUPERVISOR GENERATION OPTION.
B1
STORAGE PROTECT OPTION
*B3
NO
YES
B4
CSW STORED
YES
NO
AFTTIO
C4
TEST I/O
ERRSIO
C1
LOAD REG 8 TO REG 2, CCW KEY
ERRSIO
C2
MOVE CCW ADDRESS TO CAW
1
D1
LOAD PROGRAM INTERRUPT KEY FROM COMM REGN TO REG 2
D4
YES
CHANNEL BUSY
NO
*CD-E4 CE-D3
E1
INSERT STORAGE PROTECTION KEY INTO REG 2
E2
ERRSIO2
E4
YES
ERROR ON RECOVERY
NO
*DC *RCVERR * B2*
CHFAIL
E5
GET REQUESTOR ID
ERRSIO2
F1
MOVE CCW ADDRESS FROM REG 8 TO CAW
CSWCHK
F4
CHANNEL FAILURE
YES
NO
F5
INSERT CHANNEL AND UNIT IN I/O OLD PSW
2
*CJ * B1*
CHFAIL1
G1
MOVE STORAGE PROTECTION KEY FROM REG 2 TO CAW
G4
CHANNEL ERROR
YES
NO
G5
MOVE SYSTEM ERROR CODE X'01' TO CORE LOC 0
MACHINE CHECK INTERRUPT
*AA * B4*
H1
LOAD CHANNEL AND UNIT FROM I/O OLD PSW TO REG 2
MACHEK
H3
MOVE ACTION CODE 5 INTO MAIN STORAGE LOCATION 1
H4
UNIT CHECK
YES
NO
H5
MOVE SEREP ERROR CODE X'OF' TO MACHINE CHECK NEW PSW
1
J3
ENTER WAIT STATE *K1
J4
NO
CHANNEL END
YES
2
1
K4
RETURN TO LINK ADDR PLUS 4
K5
RETURN TO LINK ADDR

*K1
HARD WAIT
THE SUPERVISOR ENTERS THE WAIT STATE WHEN THE LOAD PSW INSTRUCTION LOADS A DOUBLE WORD WITH A 1 IN BIT POSITION 14.

Chart DG. $$A$SUP1 - SGUNCK Macro, Quiesce I/O Task
Refer to Chart 06.

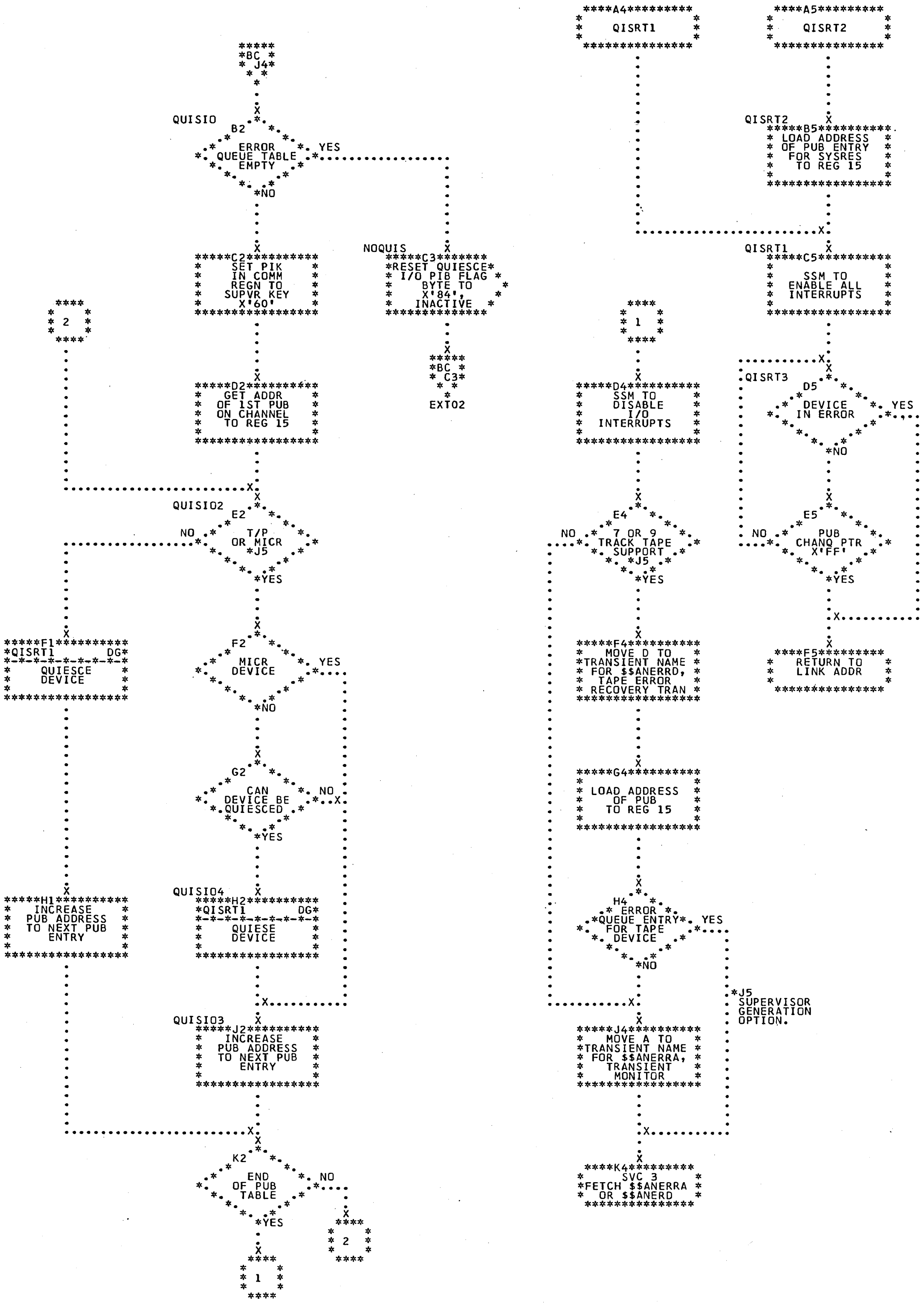

A4 QISRT1
A5 QISRT2
BC J4
QUISIO B2 ERROR QUEUE TABLE EMPTY
YES
NO
QISRT2 B5 LOAD ADDRESS OF PUB ENTRY FOR SYSRES TO REG 15
C2 SET PIK IN COMM REGN TO SUPVR KEY X'60'
NOQUIS C3 RESET QUIESCE I/O PIB FLAG BYTE TO X'84', INACTIVE
QISRT1 C5 SSM TO ENABLE ALL INTERRUPTS
2
BC C3
EXT02
1
QISRT3 D5 DEVICE IN ERROR
D2 GET ADDR OF 1ST PUB ON CHANNEL TO REG 15
D4 SSM TO DISABLE I/O INTERRUPTS
QUISIO2 E2 T/P OR MICR J5
E4 7 OR 9 TRACK TAPE SUPPORT J5
E5 PUB CHANQ PTR X'FF'
F1 QISRT1 DG QUIESCE DEVICE
F2 MICR DEVICE
F4 MOVE D TO TRANSIENT NAME FOR $$ANERRD, TAPE ERROR RECOVERY TRAN
F5 RETURN TO LINK ADDR
G2 CAN DEVICE BE QUIESCED
G4 LOAD ADDRESS OF PUB TO REG 15
H1 INCREASE PUB ADDRESS TO NEXT PUB ENTRY
QUISIO4 H2 QISRT1 DG QUIESE DEVICE
H4 ERROR QUEUE ENTRY FOR TAPE DEVICE
J5 SUPERVISOR GENERATION OPTION.
QUISIO3 J2 INCREASE PUB ADDRESS TO NEXT PUB ENTRY
J4 MOVE A TO TRANSIENT NAME FOR $$ANERRA, TRANSIENT MONITOR
K2 END OF PUB TABLE
K4 SVC 3 FETCH $$ANERRA OR $$ANERD
2
1

Chart EA. $$A$SUP1 - SGDFCH Macro, Fetch (Part 1 of 3)
Refer to Chart 03.

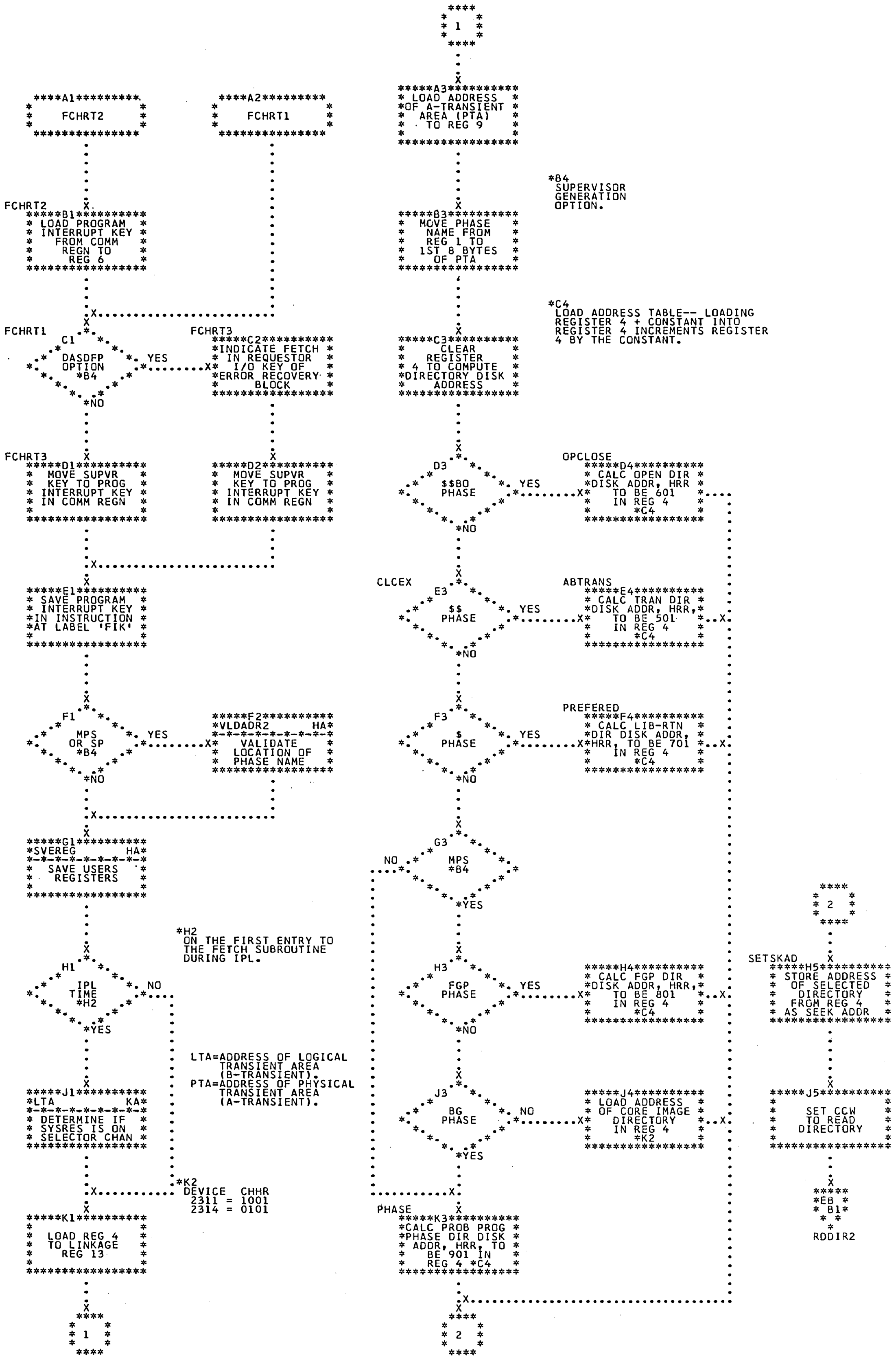
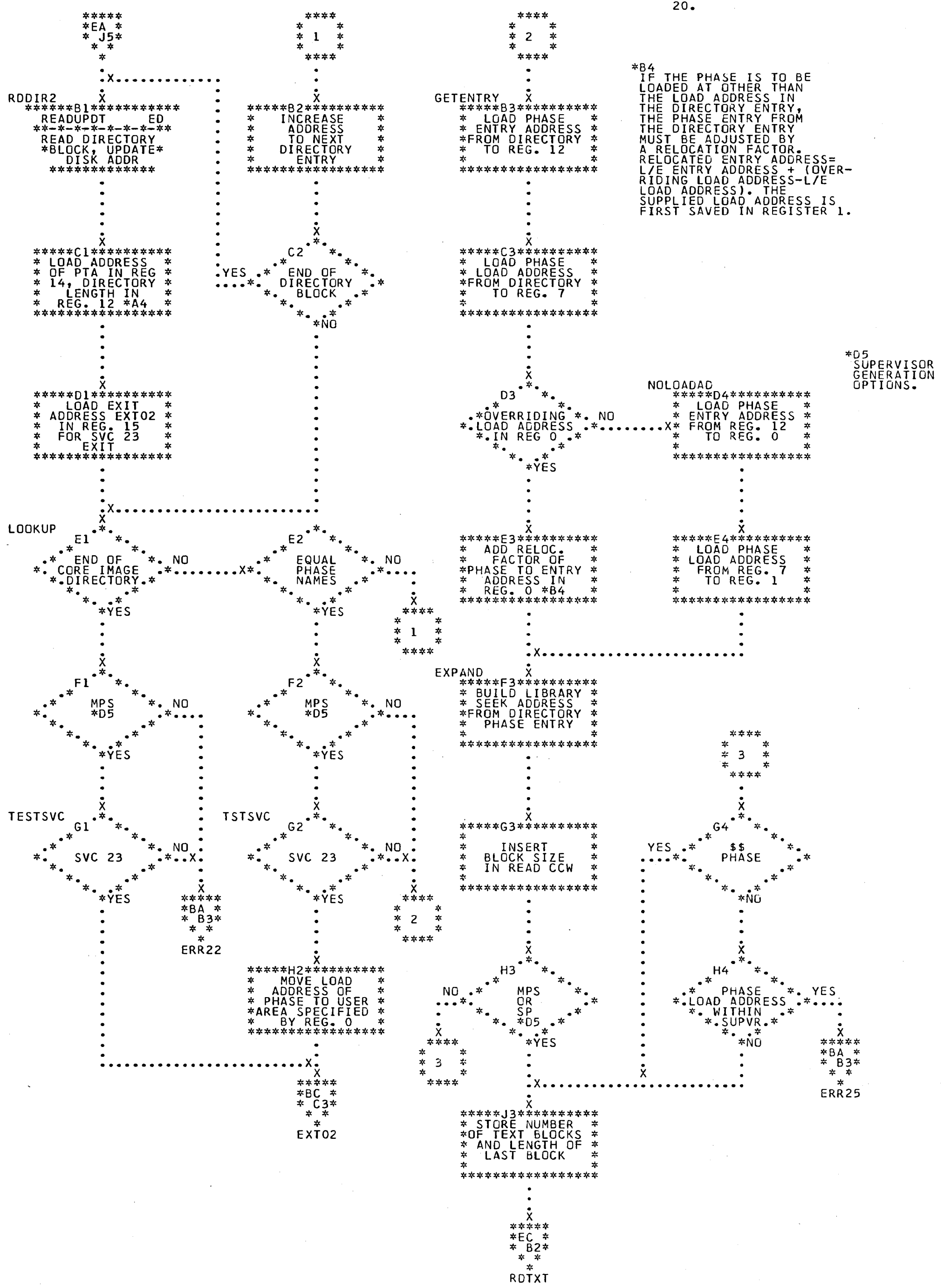

1
A3 LOAD ADDRESS OF A-TRANSIENT AREA (PTA) TO REG 9
A1 FCHRT2
A2 FCHRT1
*B4 SUPERVISOR GENERATION OPTION.
FCHRT2
B1 LOAD PROGRAM INTERRUPT KEY FROM COMM REGN TO REG 6
B3 MOVE PHASE NAME FROM REG 1 TO 1ST 8 BYTES OF PTA
*C4 LOAD ADDRESS TABLE-- LOADING REGISTER 4 + CONSTANT INTO REGISTER 4 INCREMENTS REGISTER 4 BY THE CONSTANT.
FCHRT1
C1 DASDFP OPTION *B4
YES
NO
FCHRT3
C2 INDICATE FETCH IN REQUESTOR I/O KEY OF ERROR RECOVERY BLOCK
C3 CLEAR REGISTER 4 TO COMPUTE DIRECTORY DISK ADDRESS
FCHRT3
D1 MOVE SUPVR KEY TO PROG INTERRUPT KEY IN COMM REGN
D2 MOVE SUPVR KEY TO PROG INTERRUPT KEY IN COMM REGN
D3 $$BO PHASE
YES
NO
OPCLOSE
D4 CALC OPEN DIR DISK ADDR, HRR TO BE 601 IN REG 4 *C4
E1 SAVE PROGRAM INTERRUPT KEY IN INSTRUCTION AT LABEL 'FIK'
CLCEX
E3 $$ PHASE
YES
NO
ABTRANS
E4 CALC TRAN DIR DISK ADDR, HRR, TO BE 501 IN REG 4 *C4
F1 MPS OR SP *B4
YES
NO
F2 VLDADR2 HA VALIDATE LOCATION OF PHASE NAME
F3 $ PHASE
YES
NO
PREFERED
F4 CALC LIB-RTN DIR DISK ADDR, HRR, TO BE 701 IN REG 4 *C4
G1 SVEREG HA SAVE USERS REGISTERS
G3 MPS *B4
NO
YES
2
*H2 ON THE FIRST ENTRY TO THE FETCH SUBROUTINE DURING IPL.
H1 IPL TIME *H2
NO
YES
H3 FGP PHASE
YES
NO
H4 CALC FGP DIR DISK ADDR, HRR, TO BE 801 IN REG 4 *C4
SETSKAD
H5 STORE ADDRESS OF SELECTED DIRECTORY FROM REG 4 AS SEEK ADDR
LTA=ADDRESS OF LOGICAL TRANSIENT AREA (B-TRANSIENT).
PTA=ADDRESS OF PHYSICAL TRANSIENT AREA (A-TRANSIENT).
J1 LTA KA DETERMINE IF SYSRES IS ON SELECTOR CHAN
J3 BG PHASE
NO
YES
J4 LOAD ADDRESS OF CORE IMAGE DIRECTORY IN REG 4 *K2
J5 SET CCW TO READ DIRECTORY
*K2 DEVICE CHHR
2311 = 1001
2314 = 0101
EB B1
RDDIR2
K1 LOAD REG 4 TO LINKAGE REG 13
PHASE
K3 CALC PROB PROG PHASE DIR DISK ADDR, HRR, TO BE 901 IN REG 4 *C4
1
2

Chart EB. $$A$SUP1 - SGDFCH Macro, Fetch (Part 2 of 3)
Refer to Chart 03.


EA J5
1
2
RDDIR2
B1 READUPDT ED READ DIRECTORY BLOCK, UPDATE DISK ADDR
B2 INCREASE ADDRESS TO NEXT DIRECTORY ENTRY
GETENTRY
B3 LOAD PHASE ENTRY ADDRESS FROM DIRECTORY TO REG. 12
C1 LOAD ADDRESS OF PTA IN REG 14, DIRECTORY LENGTH IN REG. 12 *A4
C2 END OF DIRECTORY BLOCK
YES
NO
C3 LOAD PHASE LOAD ADDRESS FROM DIRECTORY TO REG. 7
D1 LOAD EXIT ADDRESS EXT02 IN REG. 15 FOR SVC 23 EXIT
D3 OVERRIDING LOAD ADDRESS IN REG 0
NO
YES
NOLOADAD
D4 LOAD PHASE ENTRY ADDRESS FROM REG. 12 TO REG. 0
LOOKUP
E1 END OF CORE IMAGE DIRECTORY
NO
YES
E2 EQUAL PHASE NAMES
NO
YES
1
E3 ADD RELOC. FACTOR OF PHASE TO ENTRY ADDRESS IN REG. 0 *B4
E4 LOAD PHASE LOAD ADDRESS FROM REG. 7 TO REG. 1
F1 MPS *D5
NO
YES
F2 MPS *D5
NO
YES
EXPAND
F3 BUILD LIBRARY SEEK ADDRESS FROM DIRECTORY PHASE ENTRY
3
TESTSVC
G1 SVC 23
NO
YES
BA B3 ERR22
TSTSVC
G2 SVC 23
NO
YES
2
G3 INSERT BLOCK SIZE IN READ CCW
G4 $$ PHASE
YES
NO
H2 MOVE LOAD ADDRESS OF PHASE TO USER AREA SPECIFIED BY REG. 0
H3 MPS OR SP *D5
NO
YES
3
H4 PHASE LOAD ADDRESS WITHIN SUPVR.
YES
NO
BA B3 ERR25
BC C3 EXT02
J3 STORE NUMBER OF TEXT BLOCKS AND LENGTH OF LAST BLOCK
EC B2 RDTXT

*A4
PTA=ORIGIN OF DSECT INTO WHICH DIRECTORY BLOCK WAS READ.
DIRECTORY LENGTH=360. SEARCH IS THUS LIMITED TO 18 ENTRIES BECAUSE EACH SEARCH INCREMENT= 20.

*B4
IF THE PHASE IS TO BE LOADED AT OTHER THAN THE LOAD ADDRESS IN THE DIRECTORY ENTRY, THE PHASE ENTRY FROM THE DIRECTORY ENTRY MUST BE ADJUSTED BY A RELOCATION FACTOR. RELOCATED ENTRY ADDRESS= L/E ENTRY ADDRESS + (OVER-RIDING LOAD ADDRESS-L/E LOAD ADDRESS). THE SUPPLIED LOAD ADDRESS IS FIRST SAVED IN REGISTER 1.

*D5
SUPERVISOR GENERATION OPTIONS.

Chart EC. $$A$SUP1 - SGDFCH Macro, Fetch (Part 3 of 3)
Refer to Chart 03.

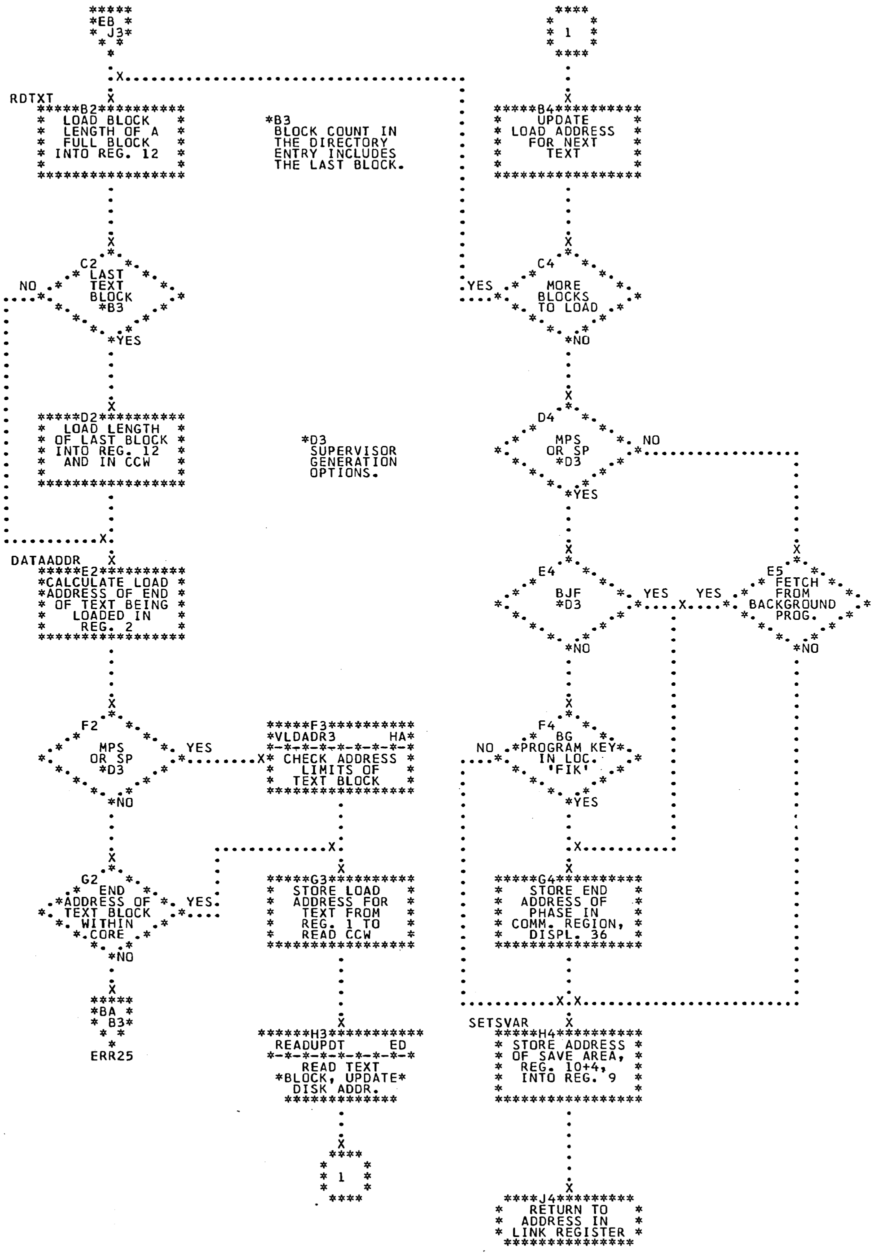

EB J3
1
RDTXT
B2 LOAD BLOCK LENGTH OF A FULL BLOCK INTO REG. 12
*B3 BLOCK COUNT IN THE DIRECTORY ENTRY INCLUDES THE LAST BLOCK.
B4 UPDATE LOAD ADDRESS FOR NEXT TEXT
C2 LAST TEXT BLOCK *B3
NO
YES
C4 MORE BLOCKS TO LOAD
YES
NO
D2 LOAD LENGTH OF LAST BLOCK INTO REG. 12 AND IN CCW
*D3 SUPERVISOR GENERATION OPTIONS.
D4 MPS OR SP *D3
NO
YES
DATAADDR
E2 CALCULATE LOAD ADDRESS OF END OF TEXT BEING LOADED IN REG. 2
E4 BJF *D3
YES
NO
E5 FETCH FROM BACKGROUND PROG.
YES
NO
F2 MPS OR SP *D3
YES
NO
F3 VLDADR3 HA
CHECK ADDRESS LIMITS OF TEXT BLOCK
F4 BG PROGRAM KEY IN LOC. 'FIK'
NO
YES
G2 END ADDRESS OF TEXT BLOCK WITHIN CORE
YES
NO
G3 STORE LOAD ADDRESS FOR TEXT FROM REG. 1 TO READ CCW
G4 STORE END ADDRESS OF PHASE IN COMM. REGION, DISPL. 36
BA B3
ERR25
H3 READUPDT ED
READ TEXT BLOCK, UPDATE DISK ADDR.
SETSVAR
H4 STORE ADDRESS OF SAVE AREA, REG. 10+4, INTO REG. 9
1
J4 RETURN TO ADDRESS IN LINK REGISTER

Chart ED. $$A$SUP1 - SGDFCH Macro, READUPDT and RSTPUB Subroutines
Refer to Chart 03.

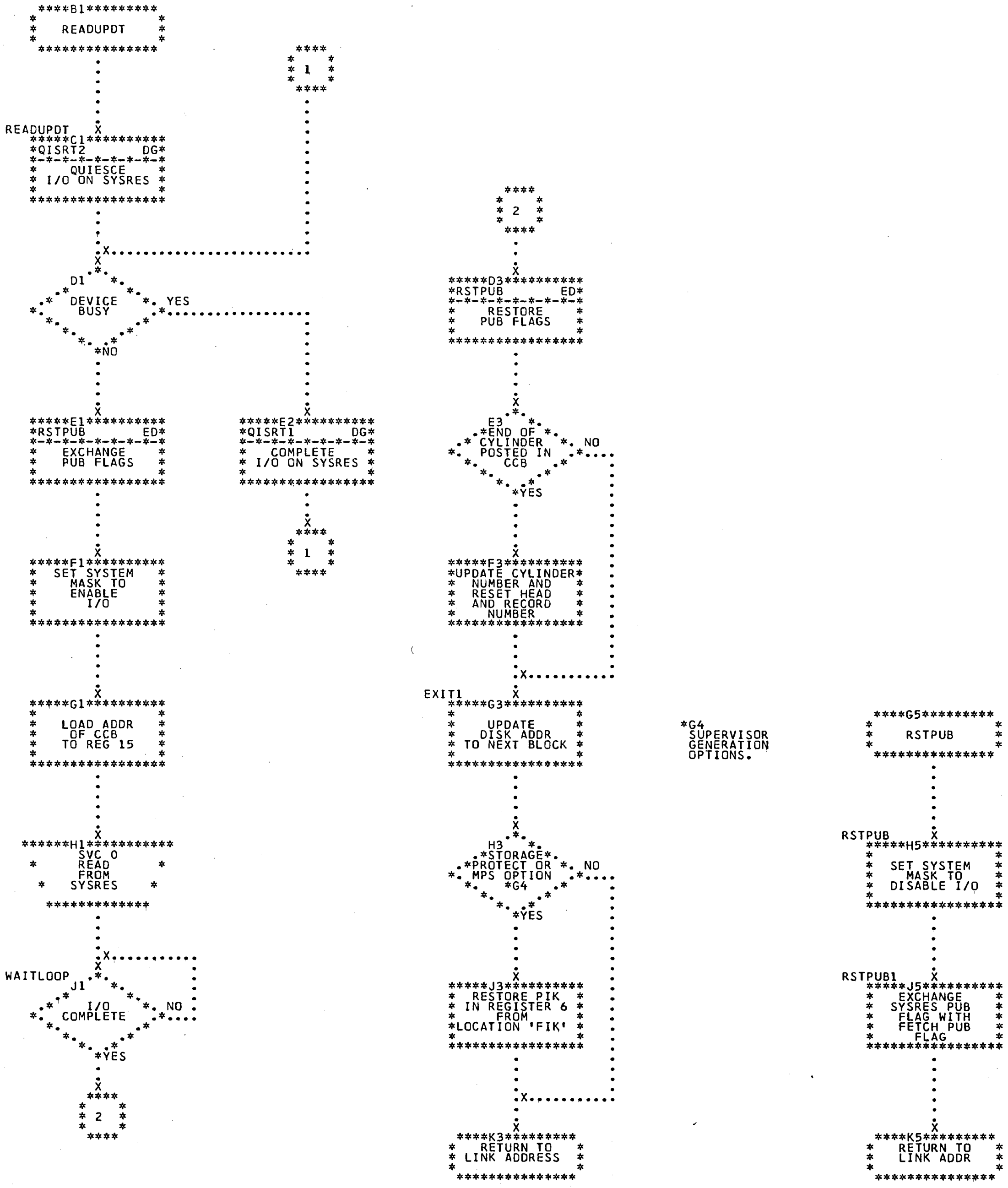
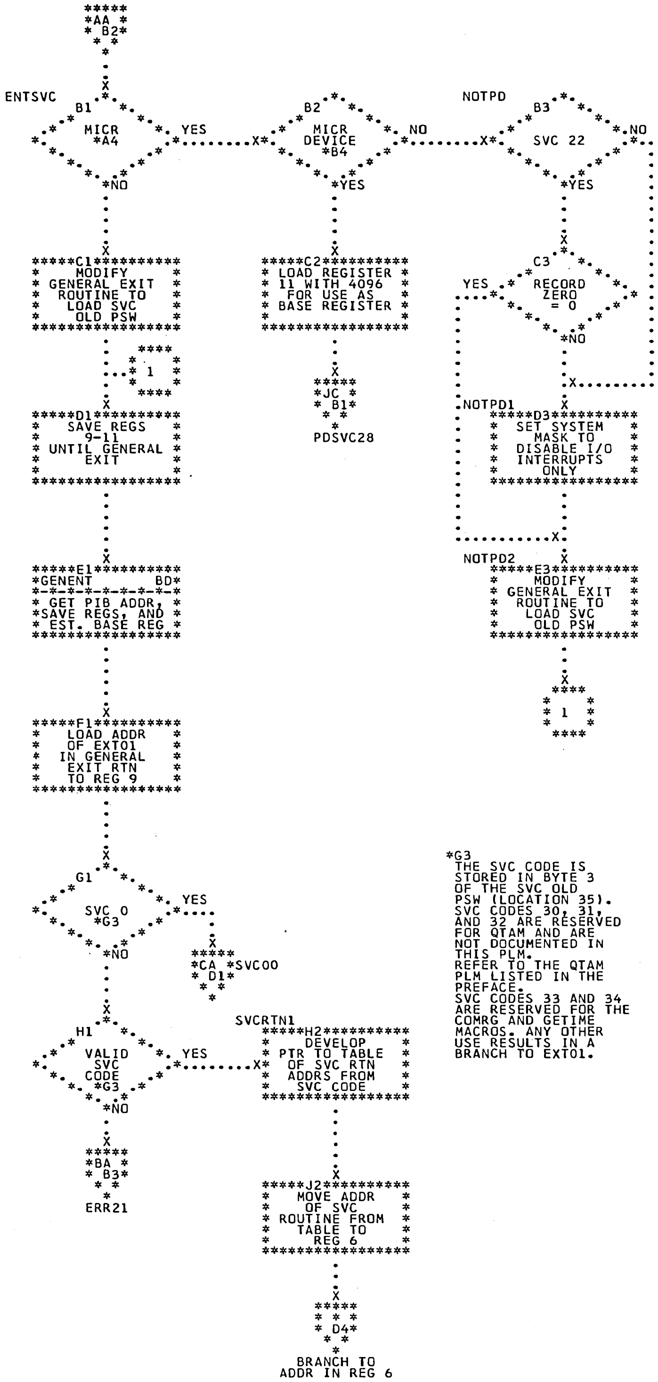

B1
READUPDT
1
READUPDT
C1
QISRT2 DG
QUIESCE I/O ON SYSRES
D1
DEVICE BUSY
YES
NO
E1
RSTPUB ED
EXCHANGE PUB FLAGS
E2
QISRT1 DG
COMPLETE I/O ON SYSRES
1
F1
SET SYSTEM MASK TO ENABLE I/O
G1
LOAD ADDR OF CCB TO REG 15
H1
SVC 0 READ FROM SYSRES
WAITLOOP
J1
I/O COMPLETE
NO
YES
2
2
D3
RSTPUB ED
RESTORE PUB FLAGS
E3
END OF CYLINDER POSTED IN CCB
NO
YES
F3
UPDATE CYLINDER NUMBER AND RESET HEAD AND RECORD NUMBER
EXIT1
G3
UPDATE DISK ADDR TO NEXT BLOCK
H3
STORAGE PROTECT OR MPS OPTION G4
NO
YES
J3
RESTORE PIK IN REGISTER 6 FROM LOCATION 'FIK'
K3
RETURN TO LINK ADDRESS
G4
SUPERVISOR GENERATION OPTIONS.
G5
RSTPUB
RSTPUB
H5
SET SYSTEM MASK TO DISABLE I/O
RSTPUB1
J5
EXCHANGE SYSRES PUB FLAG WITH FETCH PUB FLAG
K5
RETURN TO LINK ADDR

Chart FA. $$A$SUP1 - SGSVC Macro, SVC Interrupt Handler Refer to Chart 03.


AA B2
ENTSVC
B1
MICR *A4
YES
NO
B2
MICR DEVICE *B4
NO
YES
NOTPD
B3
SVC 22
NO
YES
C1 MODIFY GENERAL EXIT ROUTINE TO LOAD SVC OLD PSW
C2 LOAD REGISTER 11 WITH 4096 FOR USE AS BASE REGISTER
C3
RECORD ZERO = 0
YES
NO
1
JC B1
PDSVC28
NOTPD1
D1 SAVE REGS 9-11 UNTIL GENERAL EXIT
D3 SET SYSTEM MASK TO DISABLE I/O INTERRUPTS ONLY
NOTPD2
E1 GENENT BD
GET PIB ADDR, SAVE REGS, AND EST. BASE REG
E3 MODIFY GENERAL EXIT ROUTINE TO LOAD SVC OLD PSW
1
F1 LOAD ADDR OF EXT01 IN GENERAL EXIT RTN TO REG 9
G1
SVC 0 *G3
YES
NO
CA D1 SVC00
H1
VALID SVC CODE *G3
YES
NO
SVCRTN1
H2 DEVELOP PTR TO TABLE OF SVC RTN ADDRS FROM SVC CODE
BA B3
ERR21
J2 MOVE ADDR OF SVC ROUTINE FROM TABLE TO REG 6
D4
BRANCH TO ADDR IN REG 6

*A4
SUPERVISOR
GENERATION
OPTION.

*B4
EXTERNAL INTERRUPTS ARE NOT PROCESSED FOR MICR DEVICES ONLY. THE ENTSVC INSTRUCTION WILL BE SET TO A NO-OP BEFORE ENTERING THE USER'S STACKER-SELECT ROUTINE (SEE CHART JB-H2). THE NO-OP NOW PERMITS A RETURN TO THE MICR INTERRUPT HANDLER AT PDSVC28, CHART JB.

*G3
THE SVC CODE IS STORED IN BYTE 3 OF THE SVC OLD PSW (LOCATION 35).
SVC CODES 30, 31, AND 32 ARE RESERVED FOR QTAM AND ARE NOT DOCUMENTED IN THIS PLM.
REFER TO THE QTAM PLM LISTED IN THE PREFACE.
SVC CODES 33 AND 34 ARE RESERVED FOR THE COMRG AND GETIME MACROS. ANY OTHER USE RESULTS IN A BRANCH TO EXT01.

*D4

SVC	OPTION	LABEL	CHART
0	ALL	SVC00	CA-D1
1	ALL	SVC01	FB-B1
2	ALL	SVC02	FC-B1
3	ALL	SVC03	FD-B1
4	ALL	SVC04	FD-B3
5	ALL	SVC05	FB-B2
6	ALL	ERR23	BA-B3
7	ALL	SVC07	FD-B4
8	ALL	SVC08	FE-B1
9	ALL	SVC09	FE-B3
10	TIMER NO TIMER	SVC10 ERR21	FE-B5 BA-B3
11	ALL	SVC11	FC-B4
12	ALL	SVC12	FB-B3
13	ALL	SVC13	FB-B4
14	ALL	ERR10	BA-C4
15	MPS NO MPS	SVC15 SVC15	CA-B1 CB-B1
16	PC, NO IT OR OC PC AND IT OR OC NO PC	SXTRT1 SXTRT1 ERR21	FG-B5 FD-B2 BA-B3
17	PC, NO IT OR OC PC AND IT OR OC NO PC	EXTRT1 EXTRT1 ERR21	FG-B4 FL-B2 BA-B3
18	TIMER NO TIMER	SVC18 ERR21	FL-B2 BA-B3
19	TIMER NO TIMER	SVC19 ERR21	FL-B2 BA-B3
20	OC RTN NO OC RTN	SXTRT1 ERR21	FL-B2 BA-B3
21	OC RTN NO OC RTN	EXTRT1 ERR21	FL-B2 BA-B3
22	MPS NO MPS	SVC22 EXT01	FF-B1 BC-B1
23	MPS NO MPS	SVC23 ERR21	FF-B3 BA-B3
24	TIMER NO TIMER	SVC24 ERR21	FF-B5 BA-B3
25	BTAM NO BTAM	SVC25 ERR21	CA-E1 BA-B3
26	STOR PROT NO STOR PROT	SVC26 EXT01	FF-H4 BC-B1
27	BTAM NO BTAM	SVC27 EXT01	CA-E1 BC-B1
28	MICR NO MICR	PDSVC28 ERR21	JC-B1 BA-B3
29	MICR NO MICR	SVC29 ERR21	FB-B5 BA-B3

Chart FB. $$A$SUP1 - SGSVC Macro, SVCs 1, 5, 12, 13, and 29
Refer to Chart 03.

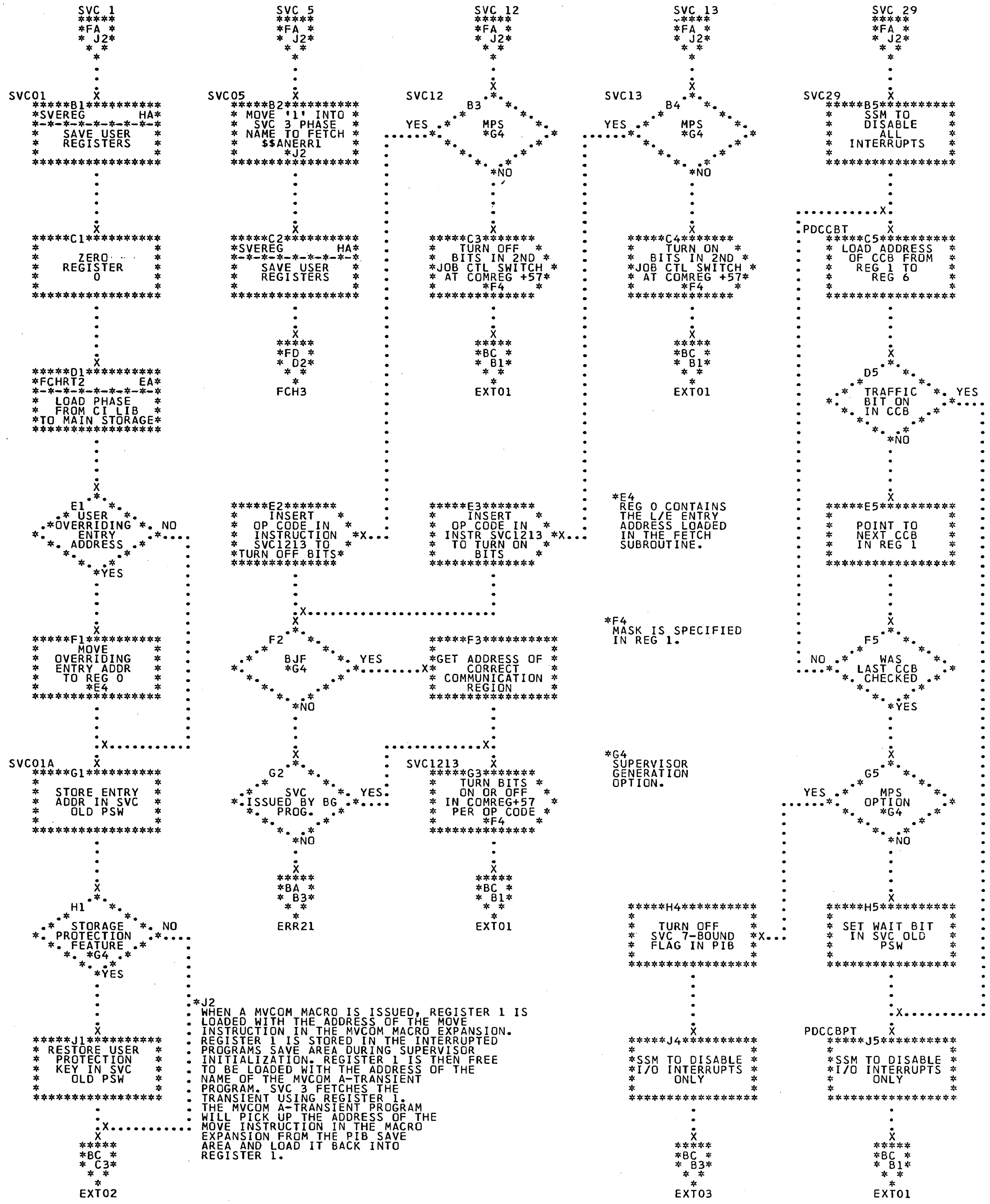
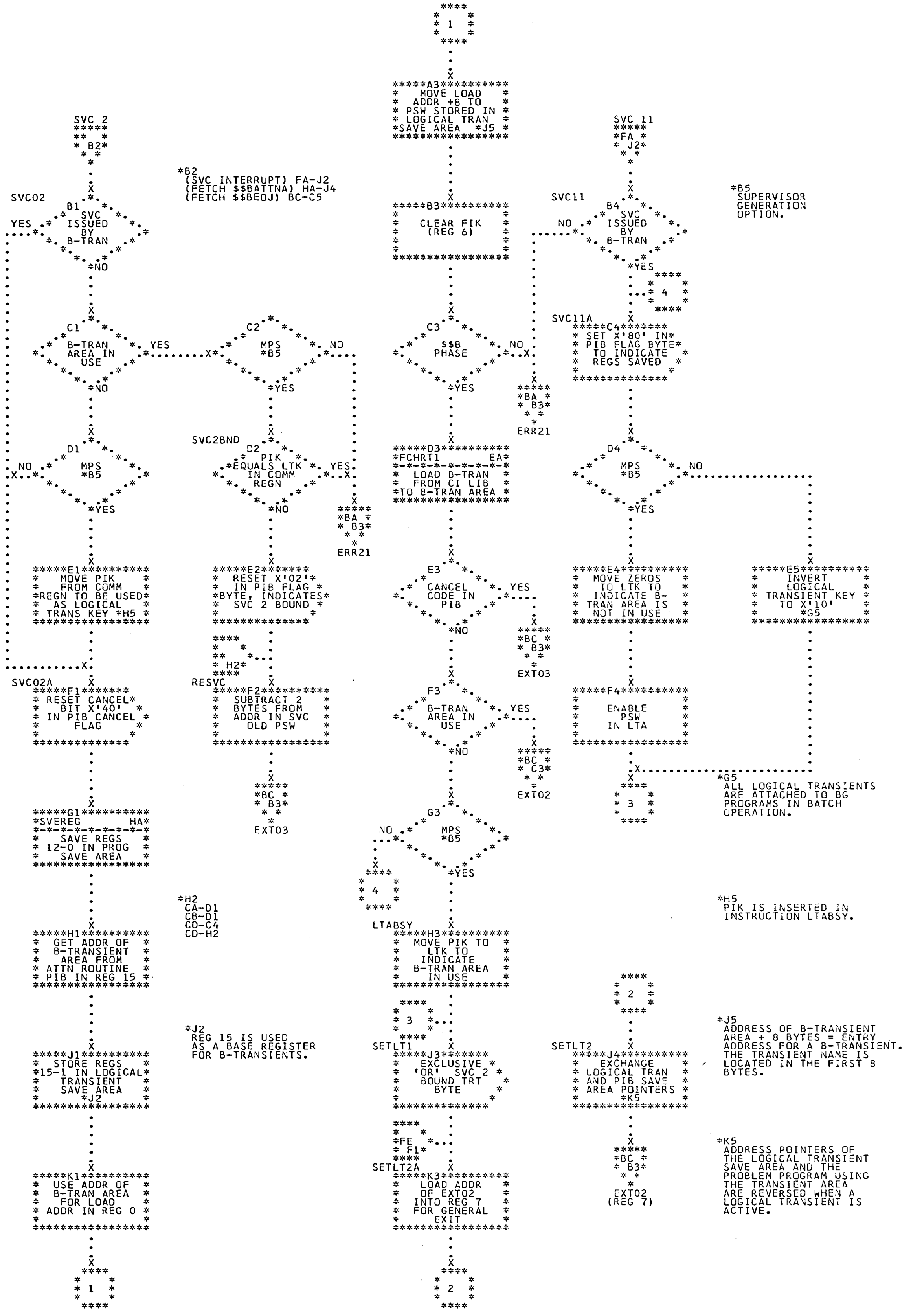

SVC 1
FA J2
SVC01
B1
SVEREG HA
SAVE USER REGISTERS
C1
ZERO REGISTER 0
D1
FCHRT2 EA
LOAD PHASE FROM CI LIB TO MAIN STORAGE
E1
USER OVERRIDING ENTRY ADDRESS
NO
YES
F1
MOVE OVERRIDING ENTRY ADDR TO REG 0
E4
SVC01A
G1
STORE ENTRY ADDR IN SVC OLD PSW
H1
STORAGE PROTECTION FEATURE
G4
NO
YES
J1
RESTORE USER PROTECTION KEY IN SVC OLD PSW
BC C3
EXT02
SVC 5
FA J2
SVC05
B2
MOVE '1' INTO SVC 3 PHASE NAME TO FETCH $$ANERR1
J2
C2
SVEREG HA
SAVE USER REGISTERS
FD D2
FCH3
E2
INSERT OP CODE IN INSTRUCTION SVC1213 TO TURN OFF BITS
F2
BJF
G4
YES
NO
G2
SVC ISSUED BY BG PROG.
YES
NO
BA B3
ERR21
SVC 12
FA J2
SVC12
B3
MPS
G4
YES
NO
C3
TURN OFF BITS IN 2ND JOB CTL SWITCH AT COMREG +57
F4
BC B1
EXT01
E3
INSERT OP CODE IN INSTR SVC1213 TO TURN ON BITS
F3
GET ADDRESS OF CORRECT COMMUNICATION REGION
SVC1213
G3
TURN BITS ON OR OFF IN COMREG+57 PER OP CODE
F4
BC B1
EXT01
SVC 13
FA J2
SVC13
B4
MPS
G4
YES
NO
C4
TURN ON BITS IN 2ND JOB CTL SWITCH AT COMREG +57
F4
BC B1
EXT01
*E4
REG 0 CONTAINS THE L/E ENTRY ADDRESS LOADED IN THE FETCH SUBROUTINE.
*F4
MASK IS SPECIFIED IN REG 1.
*G4
SUPERVISOR GENERATION OPTION.
H4
TURN OFF SVC 7-BOUND FLAG IN PIB
J4
SSM TO DISABLE I/O INTERRUPTS ONLY
BC B3
EXT03
SVC 29
FA J2
SVC29
B5
SSM TO DISABLE ALL INTERRUPTS
PDCCBT
C5
LOAD ADDRESS OF CCB FROM REG 1 TO REG 6
D5
TRAFFIC BIT ON IN CCB
YES
NO
E5
POINT TO NEXT CCB IN REG 1
F5
WAS LAST CCB CHECKED
NO
YES
G5
MPS OPTION
G4
YES
NO
H5
SET WAIT BIT IN SVC OLD PSW
PDCCBPT
J5
SSM TO DISABLE I/O INTERRUPTS ONLY
BC B1
EXT01
*J2
WHEN A MVCOM MACRO IS ISSUED, REGISTER 1 IS LOADED WITH THE ADDRESS OF THE MOVE INSTRUCTION IN THE MVCOM MACRO EXPANSION. REGISTER 1 IS STORED IN THE INTERRUPTED PROGRAMS SAVE AREA DURING SUPERVISOR INITIALIZATION. REGISTER 1 IS THEN FREE TO BE LOADED WITH THE ADDRESS OF THE NAME OF THE MVCOM A-TRANSIENT PROGRAM. SVC 3 FETCHES THE TRANSIENT USING REGISTER 1. THE MVCOM A-TRANSIENT PROGRAM WILL PICK UP THE ADDRESS OF THE MOVE INSTRUCTION IN THE MACRO EXPANSION FROM THE PIB SAVE AREA AND LOAD IT BACK INTO REGISTER 1.

Chart FC. $$A$SUP1 - SGSVC Macro, SVCs 2 and 11
Refer to Chart 03.



1
A3 MOVE LOAD ADDR +8 TO PSW STORED IN LOGICAL TRAN SAVE AREA *J5
SVC 2
B2
SVC02
B1 SVC ISSUED BY B-TRAN
YES
NO
B3 CLEAR FIK (REG 6)
SVC 11
FA J2
SVC11
B4 SVC ISSUED BY B-TRAN
NO
YES
4
C1 B-TRAN AREA IN USE
YES
NO
C2 MPS *B5
NO
YES
C3 $$B PHASE
NO
YES
BA B3 ERR21
SVC11A
C4 SET X'80' IN PIB FLAG BYTE TO INDICATE REGS SAVED
D1 MPS *B5
NO
YES
SVC2BND
D2 PIK EQUALS LTK IN COMM REGN
YES
NO
BA B3 ERR21
D3 FCHRT1 EA LOAD B-TRAN FROM CI LIB TO B-TRAN AREA
D4 MPS *B5
NO
YES
E1 MOVE PIK FROM COMM REGN TO BE USED AS LOGICAL TRANS KEY *H5
E2 RESET X'02' IN PIB FLAG BYTE, INDICATES SVC 2 BOUND
E3 CANCEL CODE IN PIB
YES
NO
BC B3 EXT03
E4 MOVE ZEROS TO LTK TO INDICATE B-TRAN AREA IS NOT IN USE
E5 INVERT LOGICAL TRANSIENT KEY TO X'10' *G5
H2
SVC02A
F1 RESET CANCEL BIT X'40' IN PIB CANCEL FLAG
RESVC
F2 SUBTRACT 2 BYTES FROM ADDR IN SVC OLD PSW
F3 B-TRAN AREA IN USE
YES
NO
BC C3 EXT02
F4 ENABLE PSW IN LTA
3
BC B3 EXT03
G1 SVEREG HA SAVE REGS 12-0 IN PROG SAVE AREA
G3 MPS *B5
NO
YES
4
LTABSY
H3 MOVE PIK TO LTK TO INDICATE B-TRAN AREA IN USE
H1 GET ADDR OF B-TRANSIENT AREA FROM ATTN ROUTINE PIB IN REG 15
2
3
SETLT1
J3 EXCLUSIVE 'OR' SVC 2 BOUND TRT BYTE
SETLT2
J4 EXCHANGE LOGICAL TRAN AND PIB SAVE AREA POINTERS *K5
J1 STORE REGS 15-1 IN LOGICAL TRANSIENT SAVE AREA *J2
FE F1
SETLT2A
K3 LOAD ADDR OF EXT02 INTO REG 7 FOR GENERAL EXIT
BC B3 EXT02 (REG 7)
K1 USE ADDR OF B-TRAN AREA FOR LOAD ADDR IN REG 0
1
2

*B2
(SVC INTERRUPT) FA-J2
(FETCH $$BATTNA) HA-J4
(FETCH $$BEOJ) BC-C5

*B5
SUPERVISOR GENERATION OPTION.

*G5
ALL LOGICAL TRANSIENTS ARE ATTACHED TO BG PROGRAMS IN BATCH OPERATION.

*H2
CA-D1
CB-D1
CD-C4
CD-H2

*H5
PIK IS INSERTED IN INSTRUCTION LTABSY.

*J2
REG 15 IS USED AS A BASE REGISTER FOR B-TRANSIENTS.

*J5
ADDRESS OF B-TRANSIENT AREA + 8 BYTES = ENTRY ADDRESS FOR A B-TRANSIENT. THE TRANSIENT NAME IS LOCATED IN THE FIRST 8 BYTES.

*K5
ADDRESS POINTERS OF THE LOGICAL TRANSIENT SAVE AREA AND THE PROBLEM PROGRAM USING THE TRANSIENT AREA ARE REVERSED WHEN A LOGICAL TRANSIENT IS ACTIVE.

Chart FD. $$A$SUP1 - SGSVC Macro, SVCs 3, 4, and 7
Refer to Chart 03.

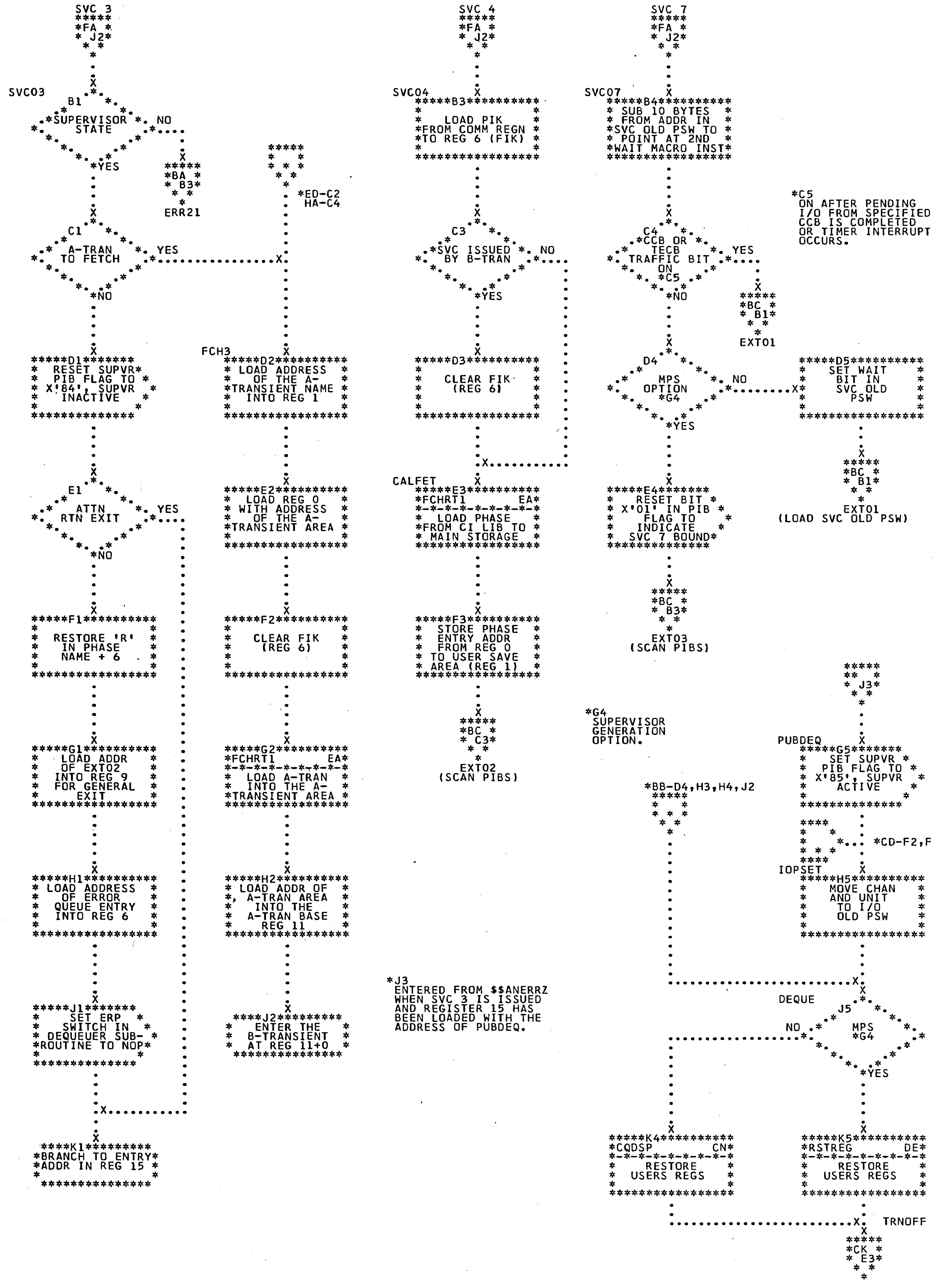
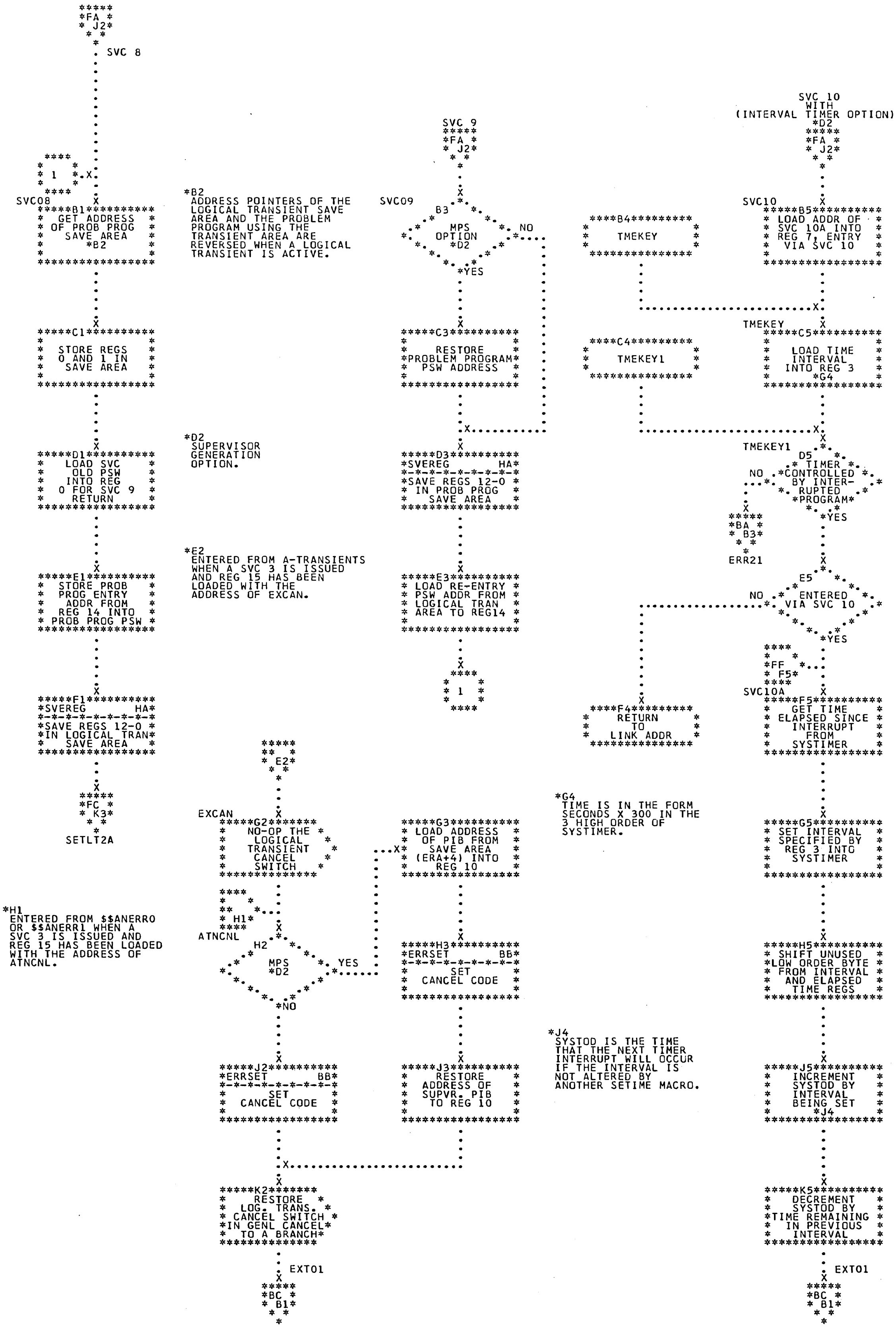

SVC 3
FA J2
SVC03
B1
SUPERVISOR STATE
NO
YES
BA B3
ERR21
C1
A-TRAN TO FETCH
YES
NO
*ED-C2 HA-C4
D1
RESET SUPVR PIB FLAG TO X'84', SUPVR INACTIVE
FCH3
D2
LOAD ADDRESS OF THE A-TRANSIENT NAME INTO REG 1
E1
ATTN RTN EXIT
YES
NO
E2
LOAD REG 0 WITH ADDRESS OF THE A-TRANSIENT AREA
F1
RESTORE 'R' IN PHASE NAME + 6
F2
CLEAR FIK (REG 6)
G1
LOAD ADDR OF EXT02 INTO REG 9 FOR GENERAL EXIT
G2
FCHRT1 EA
LOAD A-TRAN INTO THE A-TRANSIENT AREA
H1
LOAD ADDRESS OF ERROR QUEUE ENTRY INTO REG 6
H2
LOAD ADDR OF *, A-TRAN AREA INTO THE A-TRAN BASE REG 11
J1
SET ERP SWITCH IN DEQUEUER SUB-ROUTINE TO NOP
J2
ENTER THE B-TRANSIENT AT REG 11+0
K1
BRANCH TO ENTRY ADDR IN REG 15
SVC 4
FA J2
SVC04
B3
LOAD PIK FROM COMM REGN TO REG 6 (FIK)
C3
SVC ISSUED BY B-TRAN
NO
YES
D3
CLEAR FIK (REG 6)
CALFET
E3
FCHRT1 EA
LOAD PHASE FROM CI LIB TO MAIN STORAGE
F3
STORE PHASE ENTRY ADDR FROM REG 0 TO USER SAVE AREA (REG 1)
BC C3
EXT02
(SCAN PIBS)
*J3
ENTERED FROM $$ANERRZ WHEN SVC 3 IS ISSUED AND REGISTER 15 HAS BEEN LOADED WITH THE ADDRESS OF PUBDEQ.
SVC 7
FA J2
SVC07
B4
SUB 10 BYTES FROM ADDR IN SVC OLD PSW TO POINT AT 2ND WAIT MACRO INST
*C5
ON AFTER PENDING I/O FROM SPECIFIED CCB IS COMPLETED OR TIMER INTERRUPT OCCURS.
C4
CCB OR TECB TRAFFIC BIT ON *C5
YES
NO
BC B1
EXT01
D4
MPS OPTION *G4
NO
YES
D5
SET WAIT BIT IN SVC OLD PSW
BC B1
EXT01
(LOAD SVC OLD PSW)
E4
RESET BIT X'01' IN PIB FLAG TO INDICATE SVC 7 BOUND
BC B3
EXT03
(SCAN PIBS)
*G4
SUPERVISOR GENERATION OPTION.
J3
PUBDEQ
G5
SET SUPVR PIB FLAG TO X'85', SUPVR ACTIVE
*BB-D4,H3,H4,J2
*CD-F2,F4
IOPSET
H5
MOVE CHAN AND UNIT TO I/O OLD PSW
DEQUE
J5
MPS *G4
NO
YES
K4
CQDSP CN
RESTORE USERS REGS
K5
RSTREG DE
RESTORE USERS REGS
TRNOFF
CK E3

Chart FE. $$A$SUP1 - SGSVC Macro, SVCs 8, 9, and 10
Refer to Chart 03.


FA J2
SVC 8
1
SVC08
B1 GET ADDRESS OF PROB PROG SAVE AREA *B2
C1 STORE REGS 0 AND 1 IN SAVE AREA
D1 LOAD SVC OLD PSW INTO REG 0 FOR SVC 9 RETURN
E1 STORE PROB PROG ENTRY ADDR FROM REG 14 INTO PROB PROG PSW
F1 SVEREG HA SAVE REGS 12-0 IN LOGICAL TRAN SAVE AREA
FC K3
SETLT2A
SVC 9
FA J2
SVC09
B3 MPS OPTION *D2
NO
YES
C3 RESTORE PROBLEM PROGRAM PSW ADDRESS
D3 SVEREG HA SAVE REGS 12-0 IN PROB PROG SAVE AREA
E3 LOAD RE-ENTRY PSW ADDR FROM LOGICAL TRAN AREA TO REG14
1
E2
EXCAN
G2 NO-OP THE LOGICAL TRANSIENT CANCEL SWITCH
H1
ATNCNL
H2 MPS *D2
YES
NO
G3 LOAD ADDRESS OF PIB FROM SAVE AREA (ERA+4) INTO REG 10
H3 ERRSET BB SET CANCEL CODE
J2 ERRSET BB SET CANCEL CODE
J3 RESTORE ADDRESS OF SUPVR. PIB TO REG 10
K2 RESTORE LOG. TRANS. CANCEL SWITCH IN GENL CANCEL TO A BRANCH
EXT01
BC B1
B4 TMEKEY
C4 TMEKEY1
SVC 10 WITH (INTERVAL TIMER OPTION) *D2
FA J2
SVC10
B5 LOAD ADDR OF SVC 10A INTO REG 7, ENTRY VIA SVC 10
TMEKEY
C5 LOAD TIME INTERVAL INTO REG 3 *G4
TMEKEY1
D5 TIMER CONTROLLED BY INTER- RUPTED PROGRAM
NO
YES
BA B3
ERR21
E5 ENTERED VIA SVC 10
NO
YES
FF F5
SVC10A
F4 RETURN TO LINK ADDR
F5 GET TIME ELAPSED SINCE INTERRUPT FROM SYSTIMER
G5 SET INTERVAL SPECIFIED BY REG 3 INTO SYSTIMER
H5 SHIFT UNUSED LOW ORDER BYTE FROM INTERVAL AND ELAPSED TIME REGS
J5 INCREMENT SYSTOD BY INTERVAL BEING SET *J4
K5 DECREMENT SYSTOD BY TIME REMAINING IN PREVIOUS INTERVAL
EXT01
BC B1

*B2
ADDRESS POINTERS OF THE LOGICAL TRANSIENT SAVE AREA AND THE PROBLEM PROGRAM USING THE TRANSIENT AREA ARE REVERSED WHEN A LOGICAL TRANSIENT IS ACTIVE.

*D2
SUPERVISOR GENERATION OPTION.

*E2
ENTERED FROM A-TRANSIENTS WHEN A SVC 3 IS ISSUED AND REG 15 HAS BEEN LOADED WITH THE ADDRESS OF EXCAN.

*H1
ENTERED FROM $$ANERRO OR $$ANERR1 WHEN A SVC 3 IS ISSUED AND REG 15 HAS BEEN LOADED WITH THE ADDRESS OF ATNCNL.

*G4
TIME IS IN THE FORM SECONDS X 300 IN THE 3 HIGH ORDER OF SYSTIMER.

*J4
SYSTOD IS THE TIME THAT THE NEXT TIMER INTERRUPT WILL OCCUR IF THE INTERVAL IS NOT ALTERED BY ANOTHER SETIME MACRO.

Chart FF. $$A$SUP1 - SGSVC Macro, SVCs 22, 23, 24, and 26
Refer to Chart 03.

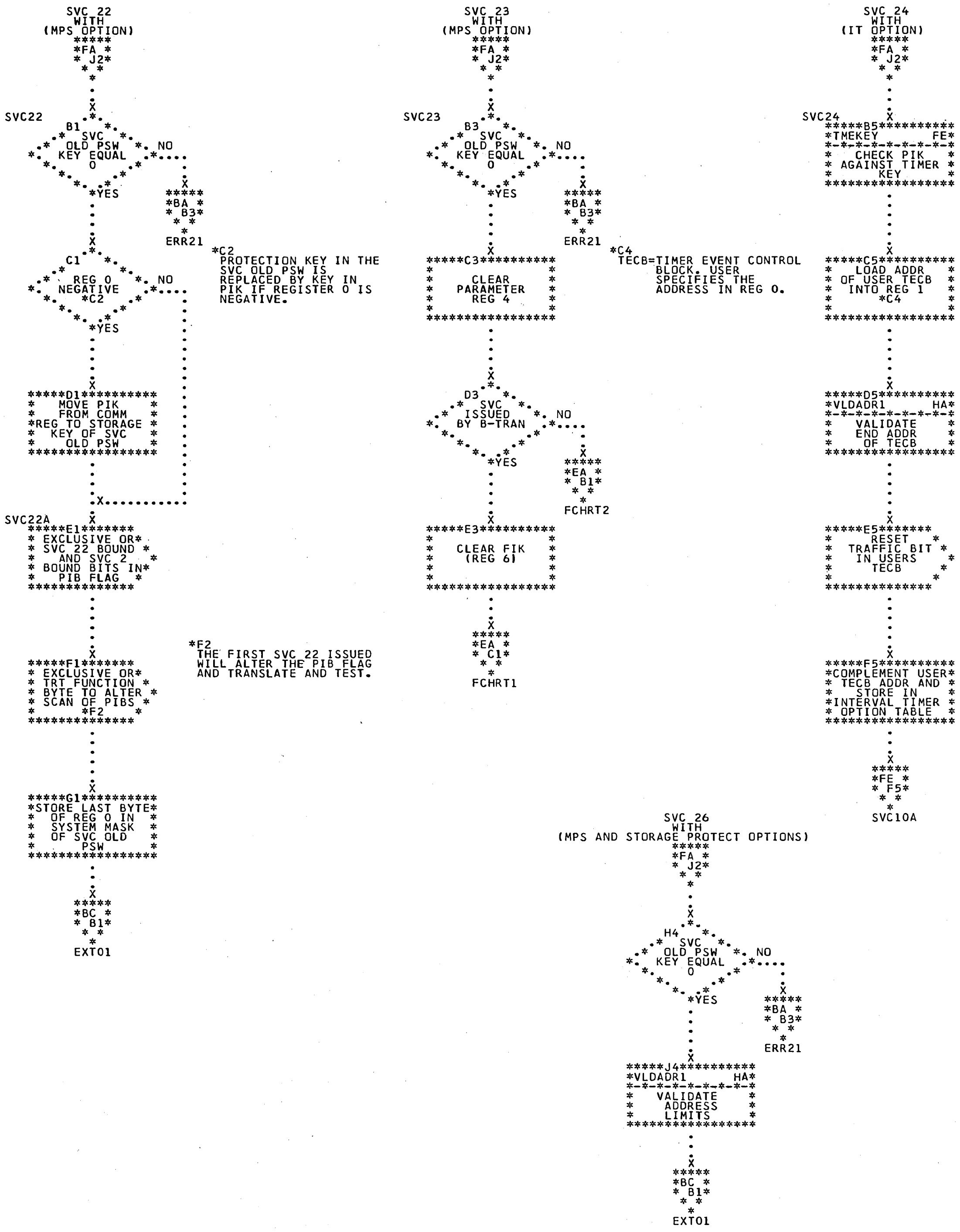
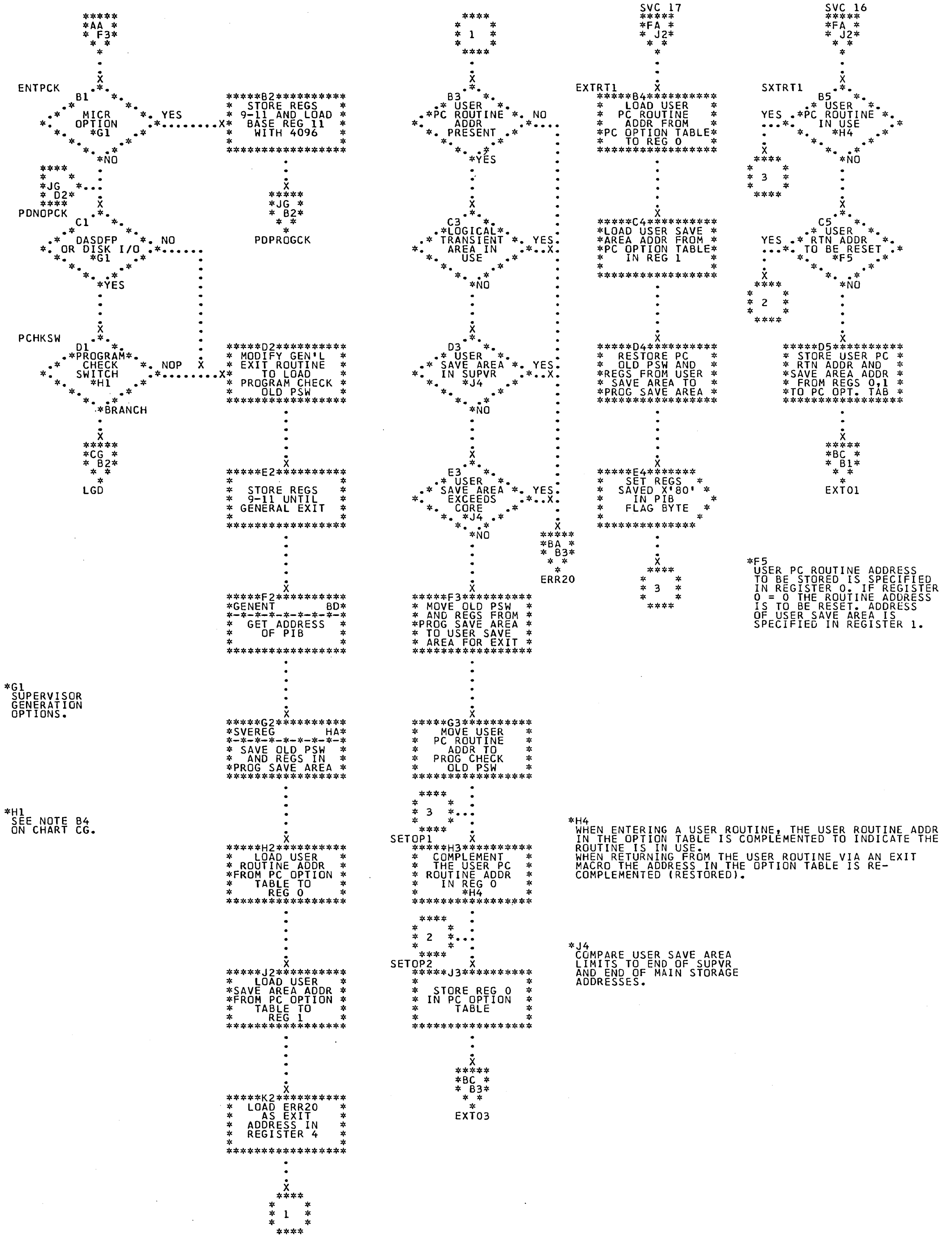

SVC 22 WITH (MPS OPTION)
FA J2
SVC22
B1 SVC OLD PSW KEY EQUAL 0
NO
YES
BA B3
ERR21
C1 REG 0 NEGATIVE *C2
NO
YES
*C2 PROTECTION KEY IN THE SVC OLD PSW IS REPLACED BY KEY IN PIK IF REGISTER 0 IS NEGATIVE.
D1 MOVE PIK FROM COMM REG TO STORAGE KEY OF SVC OLD PSW
SVC22A
E1 EXCLUSIVE OR SVC 22 BOUND AND SVC 2 BOUND BITS IN PIB FLAG
*F2 THE FIRST SVC 22 ISSUED WILL ALTER THE PIB FLAG AND TRANSLATE AND TEST.
F1 EXCLUSIVE OR TRT FUNCTION BYTE TO ALTER SCAN OF PIBS *F2
G1 STORE LAST BYTE OF REG 0 IN SYSTEM MASK OF SVC OLD PSW
BC B1
EXT01
SVC 23 WITH (MPS OPTION)
FA J2
SVC23
B3 SVC OLD PSW KEY EQUAL 0
NO
YES
BA B3
ERR21
*C4 TECB=TIMER EVENT CONTROL BLOCK. USER SPECIFIES THE ADDRESS IN REG 0.
C3 CLEAR PARAMETER REG 4
D3 SVC ISSUED BY B-TRAN
NO
YES
EA B1
FCHRT2
E3 CLEAR FIK (REG 6)
EA C1
FCHRT1
SVC 24 WITH (IT OPTION)
FA J2
SVC24
B5 TMEKEY FE CHECK PIK AGAINST TIMER KEY
C5 LOAD ADDR OF USER TECB INTO REG 1 *C4
D5 VLDADR1 HA VALIDATE END ADDR OF TECB
E5 RESET TRAFFIC BIT IN USERS TECB
F5 COMPLEMENT USER TECB ADDR AND STORE IN INTERVAL TIMER OPTION TABLE
FE F5
SVC10A
SVC 26 WITH (MPS AND STORAGE PROTECT OPTIONS)
FA J2
H4 SVC OLD PSW KEY EQUAL 0
NO
YES
BA B3
ERR21
J4 VLDADR1 HA VALIDATE ADDRESS LIMITS
BC B1
EXT01

Chart FG. $$A$SUP1 - SGSVC Macro, Program Check Interrupt and SVCs 16 and 17
Refer to Chart 03.


AA F3
ENTPCK
B1
MICR OPTION *G1
YES
NO
B2
STORE REGS 9-11 AND LOAD BASE REG 11 WITH 4096
JG B2
PDPROGCK
JG D2
PDNOPCK
C1
DASDFP OR DISK I/O *G1
NO
YES
PCHKSW
D1
PROGRAM CHECK SWITCH *H1
NOP
BRANCH
CG B2
LGD
D2
MODIFY GEN'L EXIT ROUTINE TO LOAD PROGRAM CHECK OLD PSW
E2
STORE REGS 9-11 UNTIL GENERAL EXIT
F2
GENENT BD
GET ADDRESS OF PIB
G2
SVEREG HA
SAVE OLD PSW AND REGS IN PROG SAVE AREA
H2
LOAD USER ROUTINE ADDR FROM PC OPTION TABLE TO REG 0
J2
LOAD USER SAVE AREA ADDR FROM PC OPTION TABLE TO REG 1
K2
LOAD ERR20 AS EXIT ADDRESS IN REGISTER 4
1
1
B3
USER PC ROUTINE ADDR PRESENT
NO
YES
C3
LOGICAL TRANSIENT AREA IN USE
YES
NO
D3
USER SAVE AREA IN SUPVR *J4
YES
NO
E3
USER SAVE AREA EXCEEDS CORE *J4
YES
NO
BA B3
ERR20
F3
MOVE OLD PSW AND REGS FROM PROG SAVE AREA TO USER SAVE AREA FOR EXIT
G3
MOVE USER PC ROUTINE ADDR TO PROG CHECK OLD PSW
3
SETOP1
H3
COMPLEMENT THE USER PC ROUTINE ADDR IN REG 0 *H4
2
SETOP2
J3
STORE REG 0 IN PC OPTION TABLE
BC B3
EXT03
SVC 17
FA J2
EXTRT1
B4
LOAD USER PC ROUTINE ADDR FROM PC OPTION TABLE TO REG 0
C4
LOAD USER SAVE AREA ADDR FROM PC OPTION TABLE IN REG 1
D4
RESTORE PC OLD PSW AND REGS FROM USER SAVE AREA TO PROG SAVE AREA
E4
SET REGS SAVED X'80' IN PIB FLAG BYTE
3
SVC 16
FA J2
SXTRT1
B5
USER PC ROUTINE IN USE *H4
YES
NO
3
C5
USER RTN ADDR TO BE RESET *F5
YES
NO
2
D5
STORE USER PC RTN ADDR AND SAVE AREA ADDR FROM REGS 0,1 TO PC OPT. TAB
BC B1
EXT01

*F5
USER PC ROUTINE ADDRESS TO BE STORED IS SPECIFIED IN REGISTER 0. IF REGISTER 0 = 0 THE ROUTINE ADDRESS IS TO BE RESET. ADDRESS OF USER SAVE AREA IS SPECIFIED IN REGISTER 1.

*G1
SUPERVISOR GENERATION OPTIONS.

*H1
SEE NOTE B4 ON CHART CG.

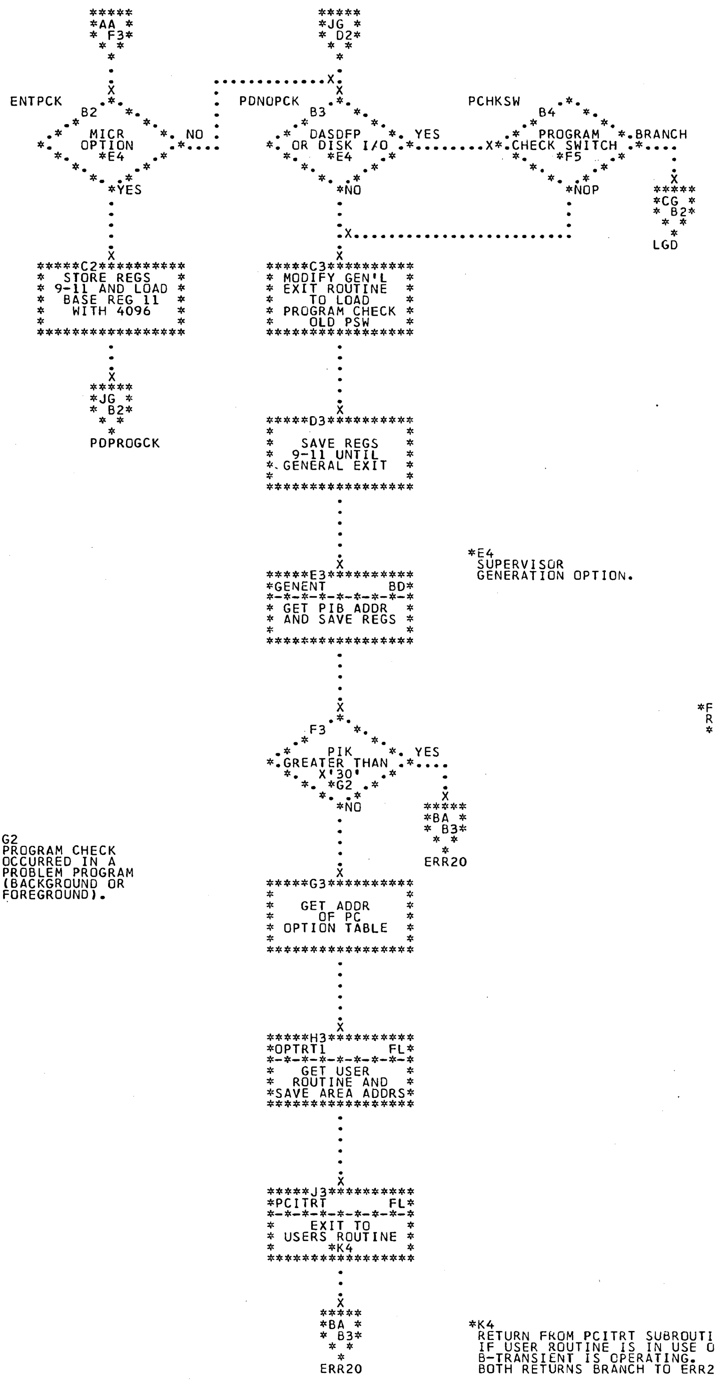
*H4
WHEN ENTERING A USER ROUTINE, THE USER ROUTINE ADDR IN THE OPTION TABLE IS COMPLEMENTED TO INDICATE THE ROUTINE IS IN USE.
WHEN RETURNING FROM THE USER ROUTINE VIA AN EXIT MACRO THE ADDRESS IN THE OPTION TABLE IS RE-COMPLEMENTED (RESTORED).

*J4
COMPARE USER SAVE AREA LIMITS TO END OF SUPVR AND END OF MAIN STORAGE ADDRESSES.

Chart FH. $$A$SUP1 - SGSVC Macro, Program Check Interrupt Refer to Chart 02.

OPTIONS--USER PC ROUTINE WITH (STORAGE PROTECT, INTERVAL TIMER, OR USER OC ROUTINE).

SEE CHART BA IF NO USER PC ROUTINE OPTION PRESENT. SEE CHART FG IF NONE OF THE OTHER OPTIONS LISTED ABOVE ARE PRESENT.


AA F3
JG D2
ENTPCK
B2
MICR OPTION *E4
NO
YES
PDNOPCK
B3
DASDFP OR DISK I/O *E4
YES
NO
PCHKSW
B4
PROGRAM CHECK SWITCH *F5
BRANCH
NOP
CG B2
LGD
C2
STORE REGS 9-11 AND LOAD BASE REG 11 WITH 4096
JG B2
PDPROGCK
C3
MODIFY GEN'L EXIT ROUTINE TO LOAD PROGRAM CHECK OLD PSW
D3
SAVE REGS 9-11 UNTIL GENERAL EXIT
E3
GENENT BD
GET PIB ADDR AND SAVE REGS
F3
PIK GREATER THAN X'30' *G2
YES
NO
BA B3
ERR20
G3
GET ADDR OF PC OPTION TABLE
H3
OPTRT1 FL
GET USER ROUTINE AND SAVE AREA ADDRS
J3
PCITRT FL
EXIT TO USERS ROUTINE *K4
BA B3
ERR20

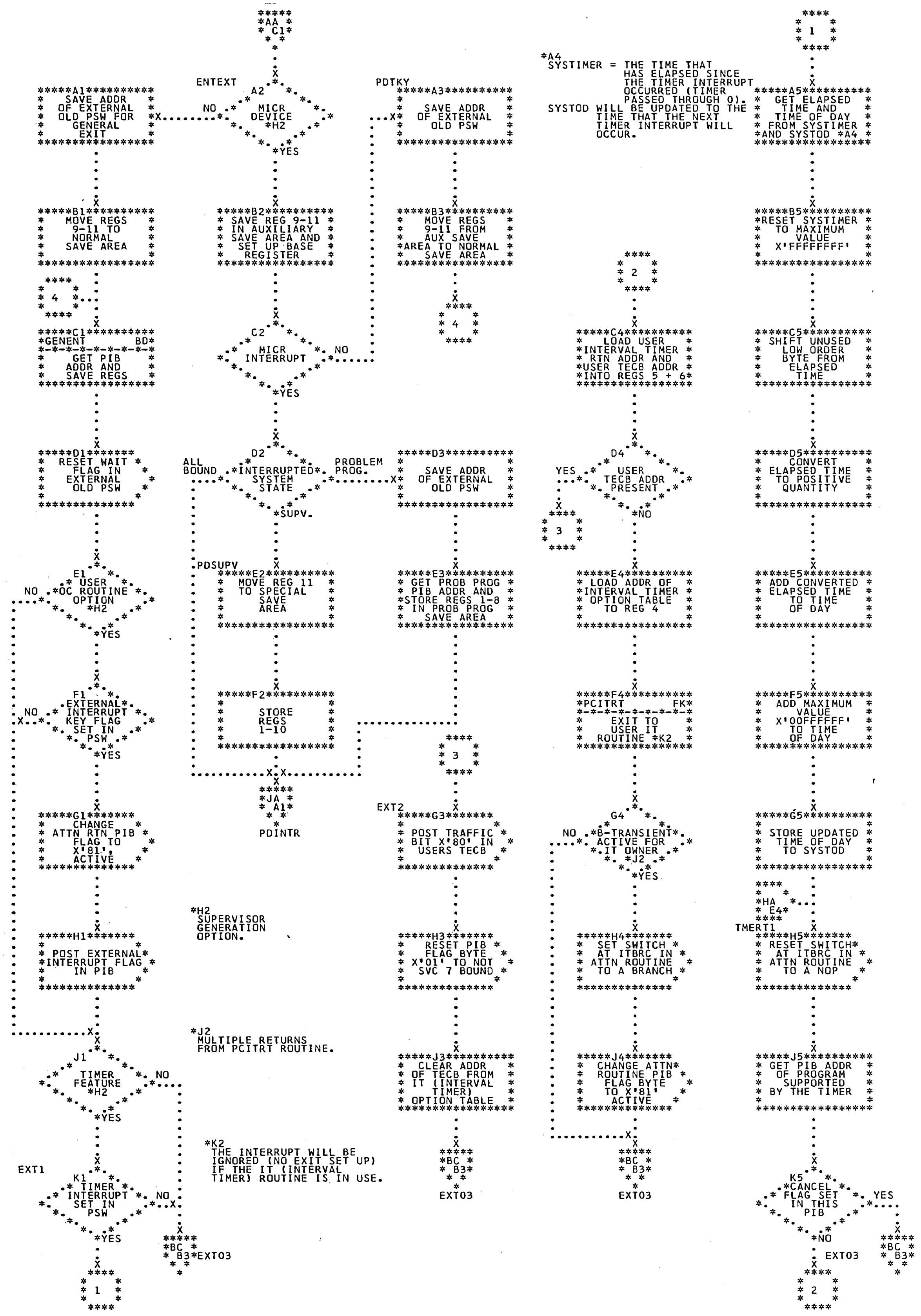
*E4 SUPERVISOR GENERATION OPTION.

*F5 REFER TO NOTE *B4 ON CHART CG.

*G2 PROGRAM CHECK OCCURRED IN A PROBLEM PROGRAM (BACKGROUND OR FOREGROUND).

*K4 RETURN FROM PCITRT SUBROUTINE IF USER ROUTINE IS IN USE OR B-TRANSIENT IS OPERATING. BOTH RETURNS BRANCH TO ERR20.

Chart FJ. $$A$SUP1 - SGSVC Macro, External Interrupt
Refer to Chart 05.


AA C1
A1 SAVE ADDR OF EXTERNAL OLD PSW FOR GENERAL EXIT
ENTEXT
A2 MICR DEVICE *H2
NO
YES
PDTKY
A3 SAVE ADDR OF EXTERNAL OLD PSW
1
A5 GET ELAPSED TIME AND TIME OF DAY FROM SYSTIMER AND SYSTOD *A4
B1 MOVE REGS 9-11 TO NORMAL SAVE AREA
B2 SAVE REG 9-11 IN AUXILIARY SAVE AREA AND SET UP BASE REGISTER
B3 MOVE REGS 9-11 FROM AUX SAVE AREA TO NORMAL SAVE AREA
B5 RESET SYSTIMER TO MAXIMUM VALUE X'FFFFFFFF'
4
2
4
C1 GENENT BD GET PIB ADDR AND SAVE REGS
C2 MICR INTERRUPT
NO
YES
C4 LOAD USER INTERVAL TIMER RTN ADDR AND USER TECB ADDR INTO REGS 5 + 6
C5 SHIFT UNUSED LOW ORDER BYTE FROM ELAPSED TIME
D1 RESET WAIT FLAG IN EXTERNAL OLD PSW
D2 INTERRUPTED SYSTEM STATE
ALL BOUND
PROBLEM PROG.
SUPV.
D3 SAVE ADDR OF EXTERNAL OLD PSW
D4 USER TECB ADDR PRESENT
YES
NO
3
D5 CONVERT ELAPSED TIME TO POSITIVE QUANTITY
E1 USER OC ROUTINE OPTION *H2
NO
YES
PDSUPV
E2 MOVE REG 11 TO SPECIAL SAVE AREA
E3 GET PROB PROG PIB ADDR AND STORE REGS 1-8 IN PROB PROG SAVE AREA
E4 LOAD ADDR OF INTERVAL TIMER OPTION TABLE TO REG 4
E5 ADD CONVERTED ELAPSED TIME TO TIME OF DAY
F1 EXTERNAL INTERRUPT KEY FLAG SET IN PSW
NO
YES
F2 STORE REGS 1-10
3
F4 PCITRT FK EXIT TO USER IT ROUTINE *K2
F5 ADD MAXIMUM VALUE X'00FFFFFF' TO TIME OF DAY
JA A1
PDINTR
EXT2
G1 CHANGE ATTN RTN PIB FLAG TO X'81', ACTIVE
G3 POST TRAFFIC BIT X'80' IN USERS TECB
G4 B-TRANSIENT ACTIVE FOR IT OWNER *J2
NO
YES
G5 STORE UPDATED TIME OF DAY TO SYSTOD
HA E4
TMERT1
H1 POST EXTERNAL INTERRUPT FLAG IN PIB
H3 RESET PIB FLAG BYTE X'01' TO NOT SVC 7 BOUND
H4 SET SWITCH AT ITBRC IN ATTN ROUTINE TO A BRANCH
H5 RESET SWITCH AT ITBRC IN ATTN ROUTINE TO A NOP
J1 TIMER FEATURE *H2
NO
YES
J3 CLEAR ADDR OF TECB FROM IT (INTERVAL TIMER) OPTION TABLE
J4 CHANGE ATTN ROUTINE PIB FLAG BYTE TO X'81' ACTIVE
J5 GET PIB ADDR OF PROGRAM SUPPORTED BY THE TIMER
BC B3
EXT03
BC B3
EXT03
EXT1
K1 TIMER INTERRUPT SET IN PSW
NO
YES
BC B3 EXT03
1
K5 CANCEL FLAG SET IN THIS PIB
YES
NO
EXT03
BC B3
2

*A4
SYSTIMER = THE TIME THAT HAS ELAPSED SINCE THE TIMER INTERRUPT OCCURRED (TIMER PASSED THROUGH 0). SYSTOD WILL BE UPDATED TO THE TIME THAT THE NEXT TIMER INTERRUPT WILL OCCUR.

*H2
SUPERVISOR GENERATION OPTION.

*J2
MULTIPLE RETURNS FROM PCITRT ROUTINE.

*K2
THE INTERRUPT WILL BE IGNORED (NO EXIT SET UP) IF THE IT (INTERVAL TIMER) ROUTINE IS IN USE.

Chart FK. $$A$SUP1 - SGSVC Macro, External Interrupt Subroutines Refer to Chart 05.

OPTIONS--USER PC ROUTINE WITH (STORAGE PROTECT, INTERVAL TIMER, OR USER OC ROUTINE).

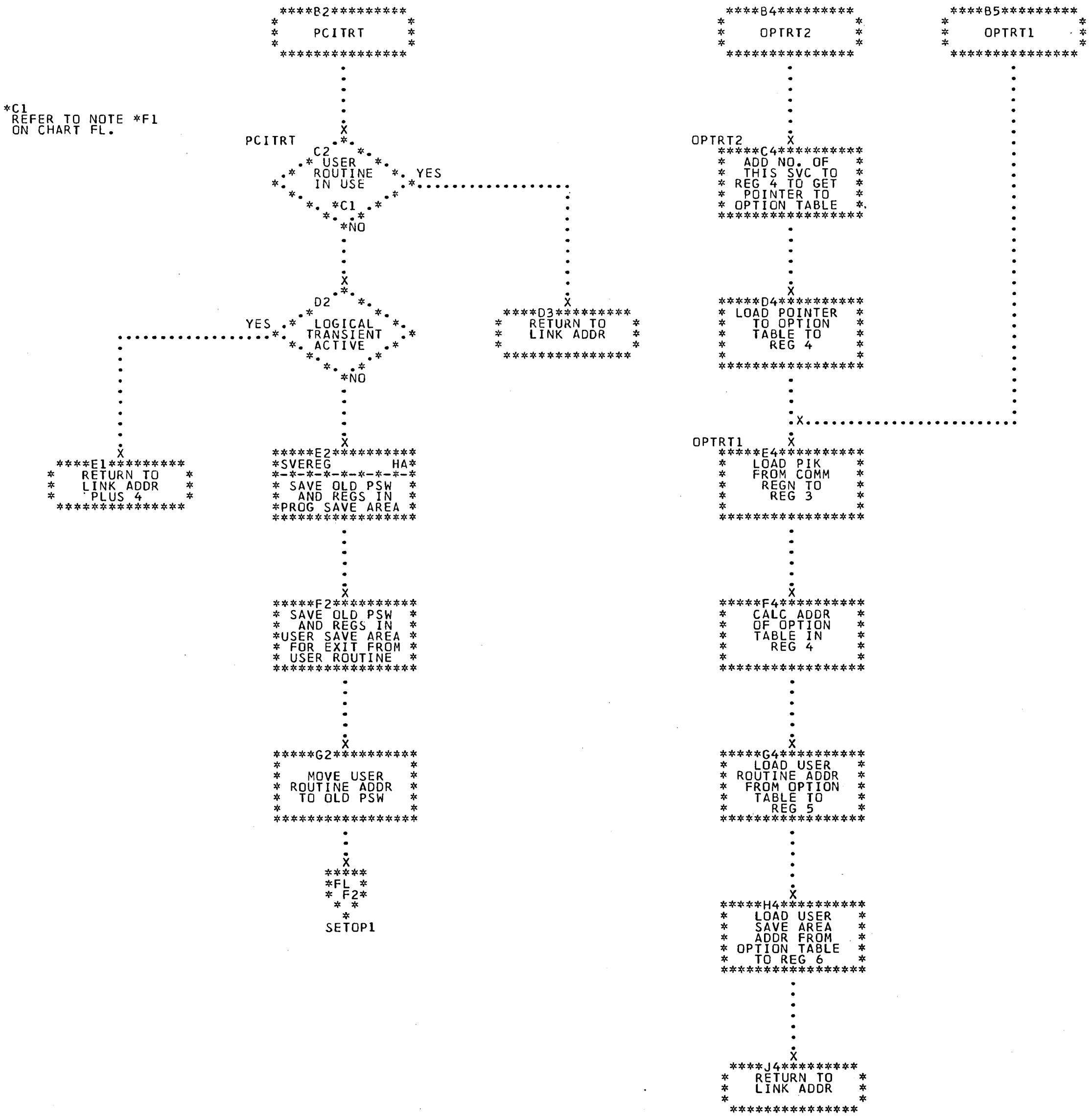
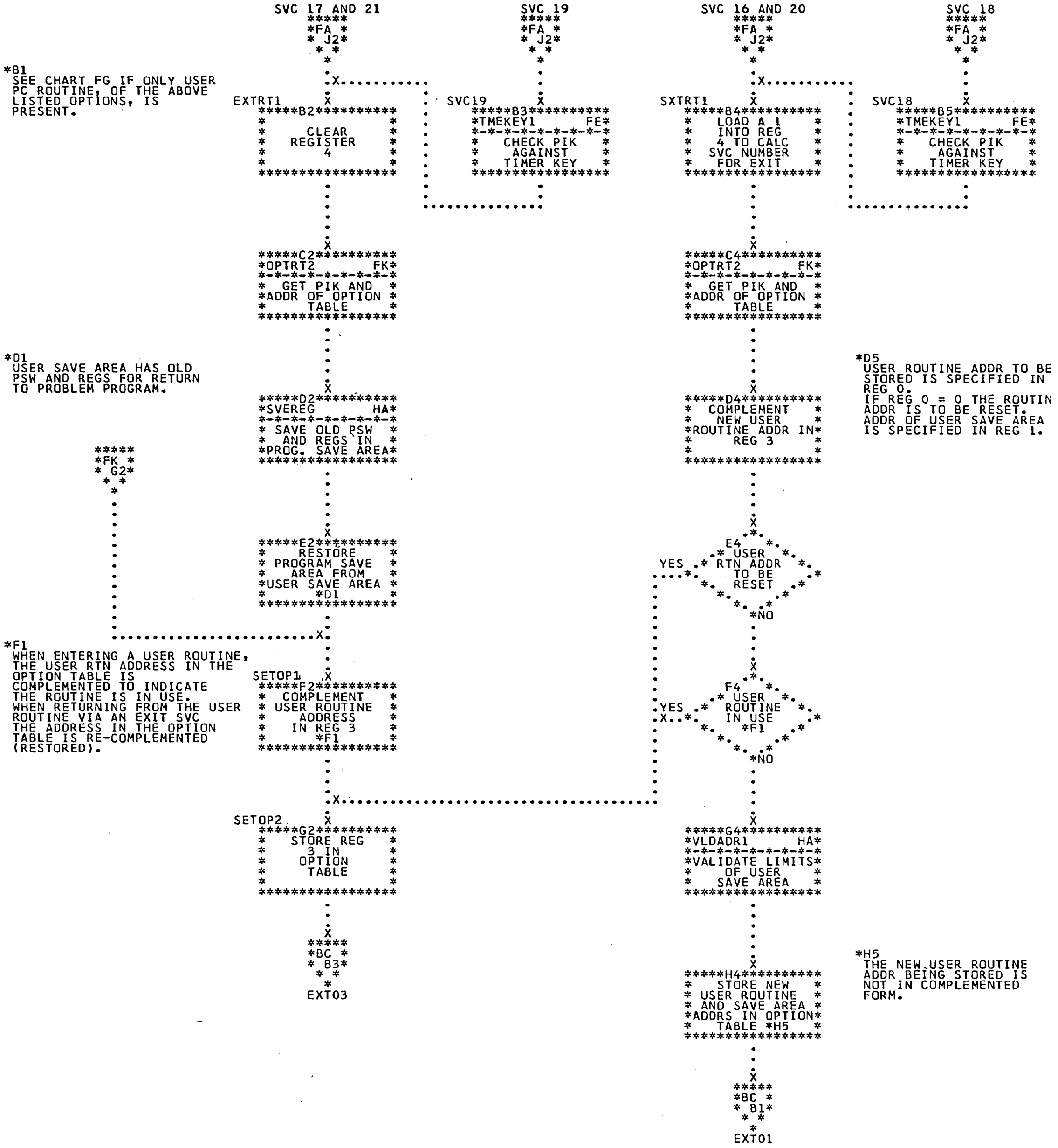

B2 PCITRT
*C1 REFER TO NOTE *F1 ON CHART FL.
PCITRT
C2 USER ROUTINE IN USE
*C1
YES
NO
D2 LOGICAL TRANSIENT ACTIVE
YES
NO
D3 RETURN TO LINK ADDR
E1 RETURN TO LINK ADDR PLUS 4
E2 SVEREG HA
SAVE OLD PSW AND REGS IN PROG SAVE AREA
F2 SAVE OLD PSW AND REGS IN USER SAVE AREA FOR EXIT FROM USER ROUTINE
G2 MOVE USER ROUTINE ADDR TO OLD PSW
FL F2
SETOP1
B4 OPTRT2
B5 OPTRT1
OPTRT2
C4 ADD NO. OF THIS SVC TO REG 4 TO GET POINTER TO OPTION TABLE
D4 LOAD POINTER TO OPTION TABLE TO REG 4
OPTRT1
E4 LOAD PIK FROM COMM REGN TO REG 3
F4 CALC ADDR OF OPTION TABLE IN REG 4
G4 LOAD USER ROUTINE ADDR FROM OPTION TABLE TO REG 5
H4 LOAD USER SAVE AREA ADDR FROM OPTION TABLE TO REG 6
J4 RETURN TO LINK ADDR

Chart FL. $$A$SUP1 - SGSVC Macro, Program Check Interrupt (SVCs 16 through 21 with Options)
Refer to Chart 03.

OPTIONS--USER PC ROUTINE WITH (STORAGE PROTECT, INTERVAL TIMER, OR USER OC ROUTINE). *B1



*B1
SEE CHART FG IF ONLY USER PC ROUTINE, OF THE ABOVE LISTED OPTIONS, IS PRESENT.

*D1
USER SAVE AREA HAS OLD PSW AND REGS FOR RETURN TO PROBLEM PROGRAM.

*F1
WHEN ENTERING A USER ROUTINE, THE USER RTN ADDRESS IN THE OPTION TABLE IS COMPLEMENTED TO INDICATE THE ROUTINE IS IN USE. WHEN RETURNING FROM THE USER ROUTINE VIA AN EXIT SVC THE ADDRESS IN THE OPTION TABLE IS RE-COMPLEMENTED (RESTORED).

*D5
USER ROUTINE ADDR TO BE STORED IS SPECIFIED IN REG 0.
IF REG 0 = 0 THE ROUTIN ADDR IS TO BE RESET.
ADDR OF USER SAVE AREA IS SPECIFIED IN REG 1.

*H5
THE NEW USER ROUTINE ADDR BEING STORED IS NOT IN COMPLEMENTED FORM.

SVC 17 AND 21
FA J2
SVC 19
FA J2
SVC 16 AND 20
FA J2
SVC 18
FA J2
EXTRT1
B2
CLEAR REGISTER 4
SVC19
B3
TMEKEY1 FE
CHECK PIK AGAINST TIMER KEY
SXTRT1
B4
LOAD A 1 INTO REG 4 TO CALC SVC NUMBER FOR EXIT
SVC18
B5
TMEKEY1 FE
CHECK PIK AGAINST TIMER KEY
C2
OPTRT2 FK
GET PIK AND ADDR OF OPTION TABLE
C4
OPTRT2 FK
GET PIK AND ADDR OF OPTION TABLE
D2
SVEREG HA
SAVE OLD PSW AND REGS IN PROG. SAVE AREA
D4
COMPLEMENT NEW USER ROUTINE ADDR IN REG 3
FK G2
E2
RESTORE PROGRAM SAVE AREA FROM USER SAVE AREA *D1
E4
USER RTN ADDR TO BE RESET
YES
NO
SETOP1
F2
COMPLEMENT USER ROUTINE ADDRESS IN REG 3 *F1
F4
USER ROUTINE IN USE *F1
YES
NO
SETOP2
G2
STORE REG 3 IN OPTION TABLE
G4
VLDADR1 HA
VALIDATE LIMITS OF USER SAVE AREA
BC B3
EXT03
H4
STORE NEW USER ROUTINE AND SAVE AREA ADDRS IN OPTION TABLE *H5
BC B1
EXT01

Chart GA. $$A$SUP1 - SGDSK Macro, Disk Error Recovery (Part 1 of 2)
Refer to Chart 06.

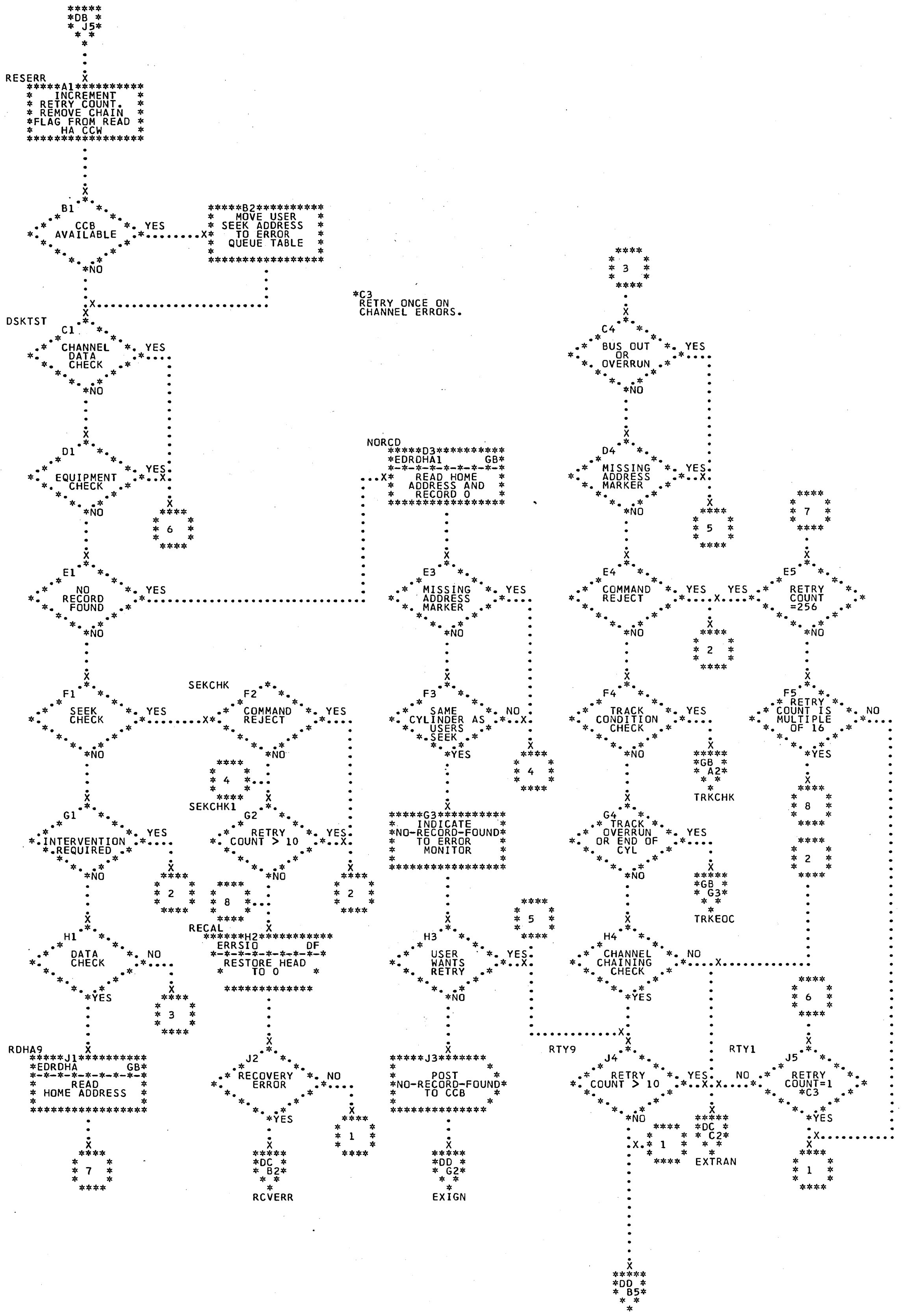


Chart GB. $$A$SUP1 - SGDSK Macro, Disk Error Recovery (Part 2 of 2)
Refer to Chart 06.

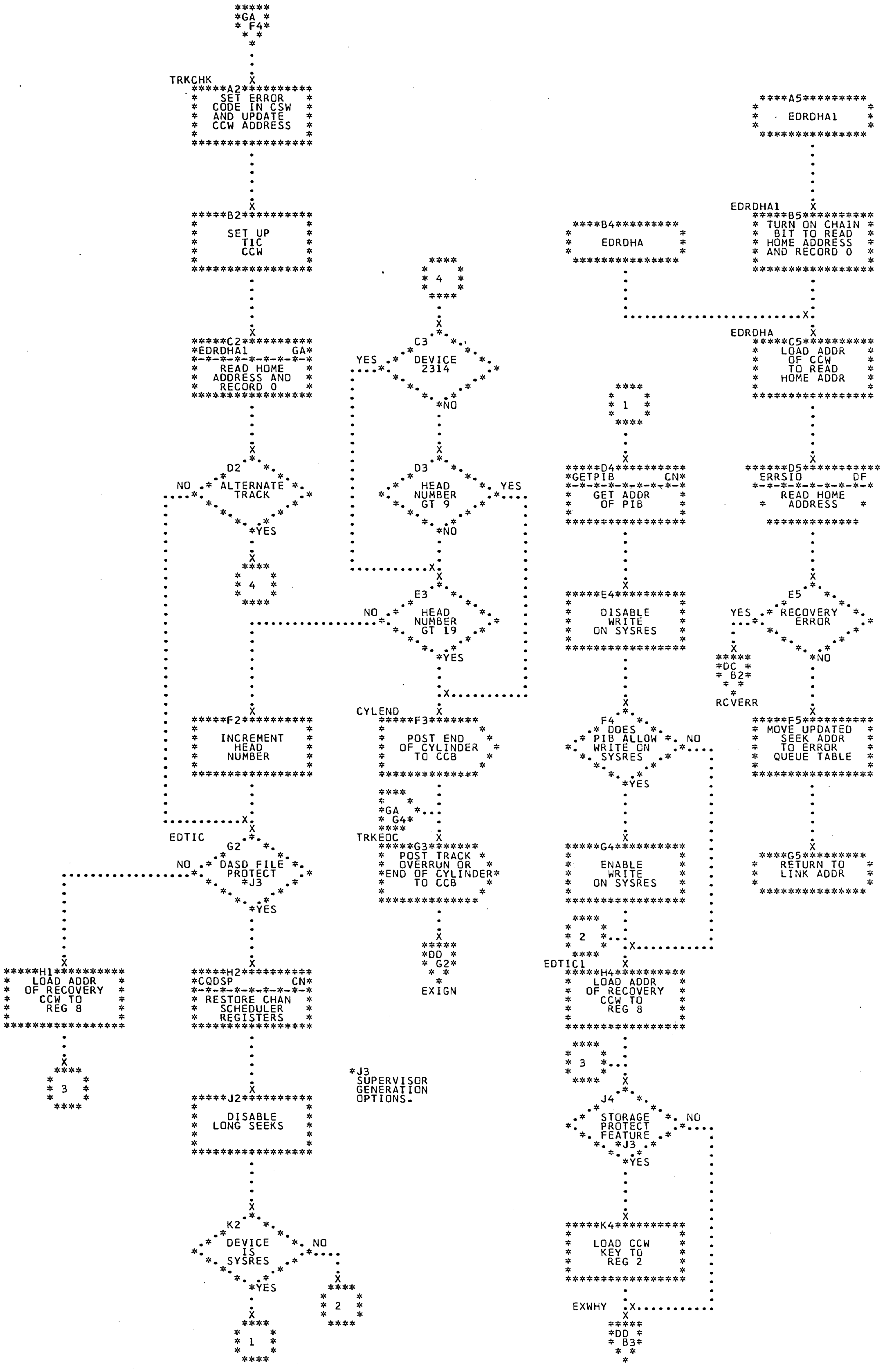


Chart HA. $$A$SUP1 - SGTCON Macro, Resident Attention, SVEREG and VLDADR Subroutines
Refer to Chart 02.

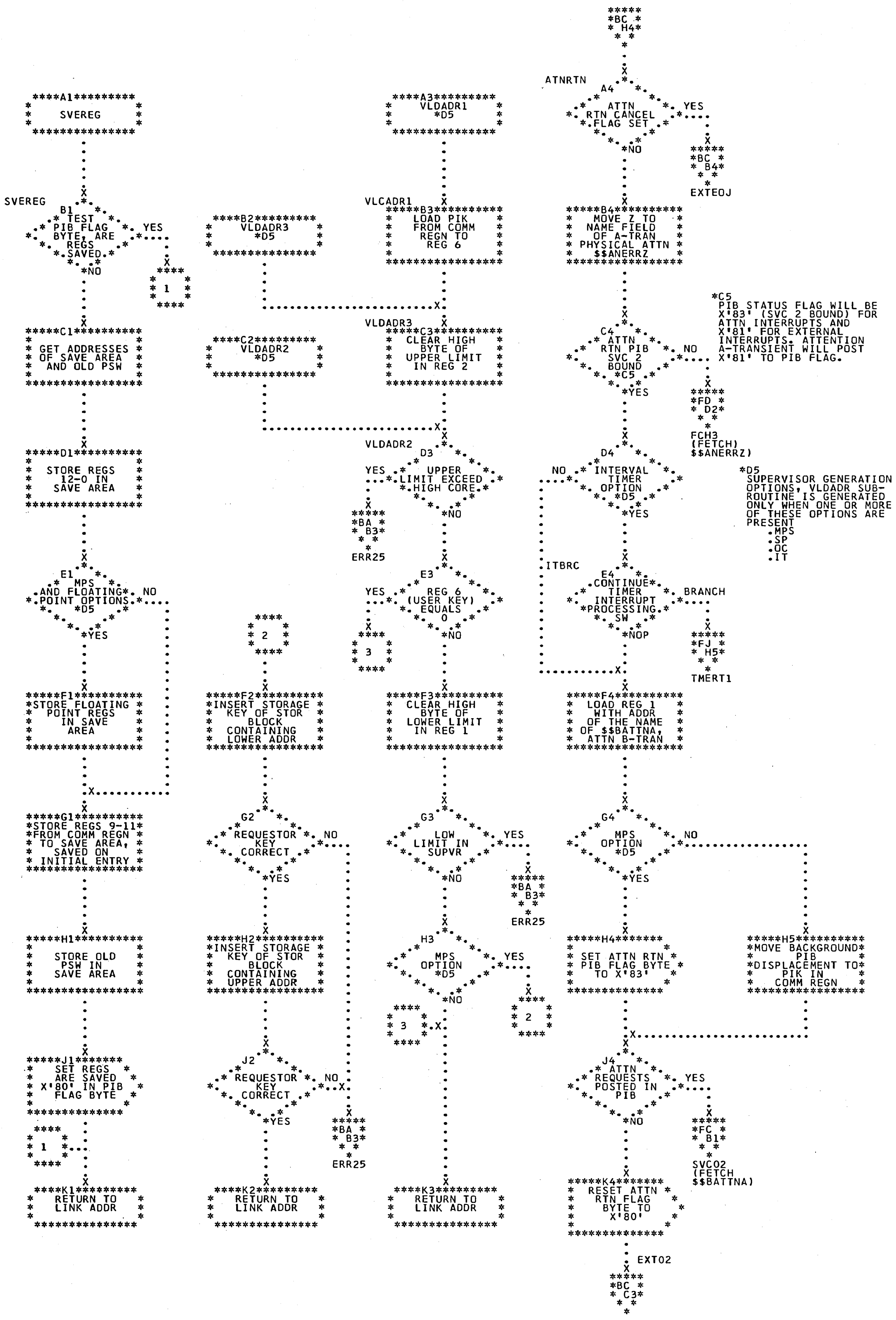
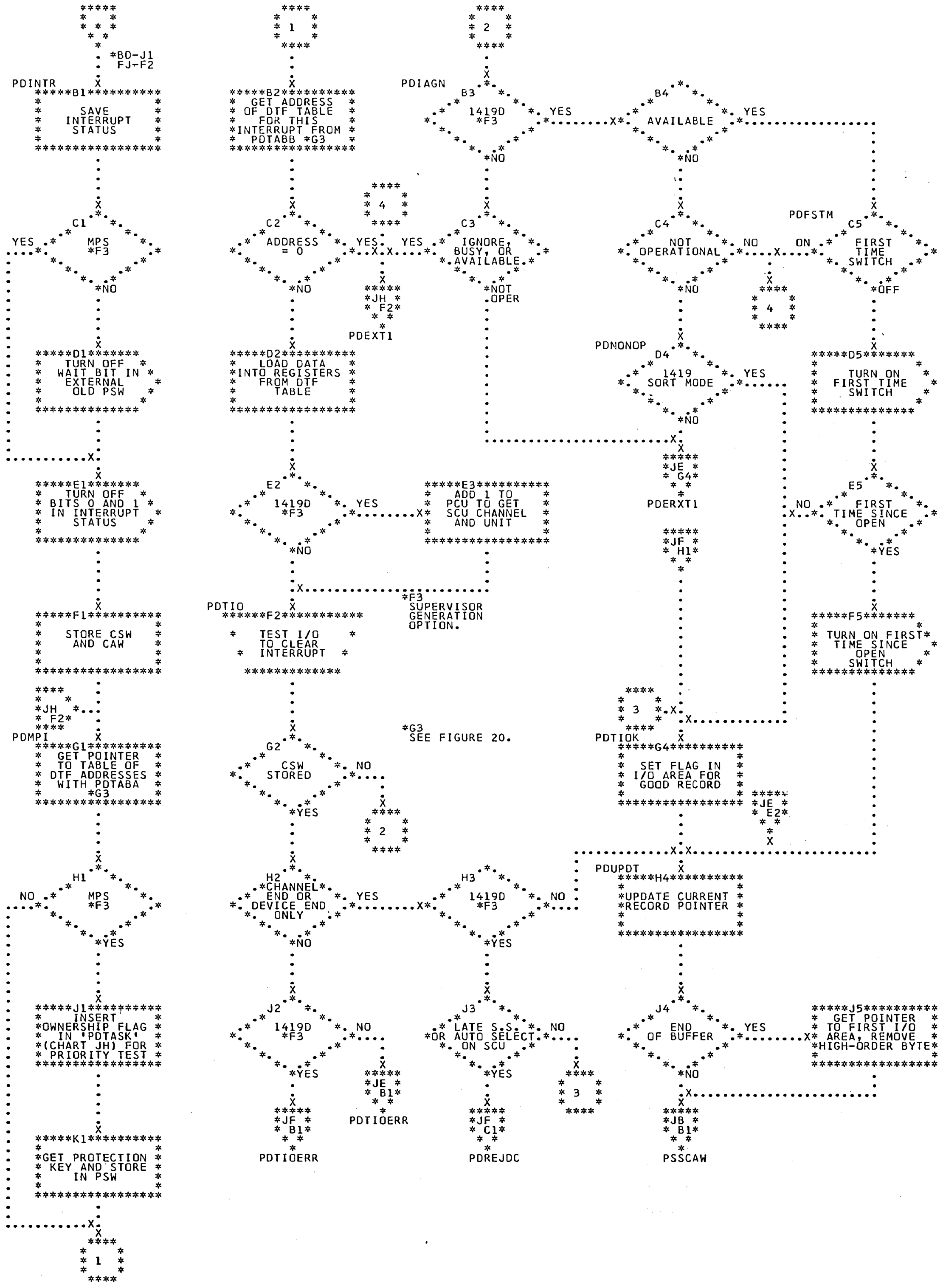

A1 SVEREG
SVEREG B1 TEST PIB FLAG BYTE, ARE REGS SAVED. YES NO
1
C1 GET ADDRESSES OF SAVE AREA AND OLD PSW
D1 STORE REGS 12-0 IN SAVE AREA
E1 MPS AND FLOATING POINT OPTIONS *D5 NO YES
F1 STORE FLOATING POINT REGS IN SAVE AREA
G1 STORE REGS 9-11 FROM COMM REGN TO SAVE AREA, SAVED ON INITIAL ENTRY
H1 STORE OLD PSW IN SAVE AREA
J1 SET REGS ARE SAVED X'80' IN PIB FLAG BYTE
1
K1 RETURN TO LINK ADDR
B2 VLDADR3 *D5
C2 VLDADR2 *D5
2
F2 INSERT STORAGE KEY OF STOR BLOCK CONTAINING LOWER ADDR
G2 REQUESTOR KEY CORRECT NO YES
H2 INSERT STORAGE KEY OF STOR BLOCK CONTAINING UPPER ADDR
J2 REQUESTOR KEY CORRECT NO YES
BA B3 ERR25
K2 RETURN TO LINK ADDR
A3 VLDADR1 *D5
VLCADR1 B3 LOAD PIK FROM COMM REGN TO REG 6
VLDADR3 C3 CLEAR HIGH BYTE OF UPPER LIMIT IN REG 2
VLDADR2 D3 UPPER LIMIT EXCEED HIGH CORE. YES NO
BA B3 ERR25
E3 REG 6 (USER KEY) EQUALS 0 YES NO
3
F3 CLEAR HIGH BYTE OF LOWER LIMIT IN REG 1
G3 LOW LIMIT IN SUPVR YES NO
BA B3 ERR25
H3 MPS OPTION *D5 YES NO
3
2
K3 RETURN TO LINK ADDR
BC H4
ATNRTN A4 ATTN RTN CANCEL FLAG SET YES NO
BC B4 EXTEOJ
B4 MOVE Z TO NAME FIELD OF A-TRAN PHYSICAL ATTN $$ANERRZ
*C5 PIB STATUS FLAG WILL BE X'83' (SVC 2 BOUND) FOR ATTN INTERRUPTS AND X'81' FOR EXTERNAL INTERRUPTS. ATTENTION A-TRANSIENT WILL POST X'81' TO PIB FLAG.
C4 ATTN RTN PIB SVC 2 BOUND *C5 NO YES
FD D2 FCH3 (FETCH) $$ANERRZ)
D4 INTERVAL TIMER OPTION *D5 NO YES
*D5 SUPERVISOR GENERATION OPTIONS, VLDADR SUB-ROUTINE IS GENERATED ONLY WHEN ONE OR MORE OF THESE OPTIONS ARE PRESENT .MPS .SP .OC .IT
ITBRC E4 CONTINUE TIMER INTERRUPT PROCESSING. SW BRANCH NOP
FJ H5 TMERT1
F4 LOAD REG 1 WITH ADDR OF THE NAME OF $$BATTNA, ATTN B-TRAN
G4 MPS OPTION *D5 NO YES
H4 SET ATTN RTN PIB FLAG BYTE TO X'83'
H5 MOVE BACKGROUND PIB DISPLACEMENT TO PIK IN COMM REGN
J4 ATTN REQUESTS POSTED IN PIB YES NO
FC B1 SVC02 (FETCH $$BATTNA)
K4 RESET ATTN RTN FLAG BYTE TO X'80'
EXT02
BC C3

Chart JA. $$A$SUP1 - SMICR Macro, External Interrupt (Part 1 of 3)
Refer to Chart 05.



*BD-J1 FJ-F2
PDINTR
B1 SAVE INTERRUPT STATUS
C1 MPS *F3
YES
NO
D1 TURN OFF WAIT BIT IN EXTERNAL OLD PSW
E1 TURN OFF BITS 0 AND 1 IN INTERRUPT STATUS
F1 STORE CSW AND CAW
JH F2
PDMPI
G1 GET POINTER TO TABLE OF DTF ADDRESSES WITH PDTABA *G3
H1 MPS *F3
NO
YES
J1 INSERT OWNERSHIP FLAG IN 'PDTASK' (CHART JH) FOR PRIORITY TEST
K1 GET PROTECTION KEY AND STORE IN PSW
1
B2 GET ADDRESS OF DTF TABLE FOR THIS INTERRUPT FROM PDTABB *G3
C2 ADDRESS = 0
YES
NO
4
JH F2
PDEXT1
D2 LOAD DATA INTO REGISTERS FROM DTF TABLE
E2 1419D *F3
YES
NO
E3 ADD 1 TO PCU TO GET SCU CHANNEL AND UNIT
*F3 SUPERVISOR GENERATION OPTION.
PDTIO
F2 TEST I/O TO CLEAR INTERRUPT
*G3 SEE FIGURE 20.
G2 CSW STORED
NO
2
YES
H2 CHANNEL END OR DEVICE END ONLY
YES
NO
J2 1419D *F3
NO
JE B1
PDTIOERR
YES
JF B1
PDTIOERR
2
PDIAGN
B3 1419D *F3
YES
NO
C3 IGNORE, BUSY, OR AVAILABLE
YES
NOT OPER
H3 1419D *F3
NO
YES
J3 LATE S.S. OR AUTO SELECT. ON SCU
NO
3
YES
JF C1
PDREJDC
B4 AVAILABLE
YES
NO
C4 NOT OPERATIONAL
NO
4
PDNONOP
D4 1419 SORT MODE
YES
NO
JE G4
PDERXT1
JF H1
3
PDTIOK
G4 SET FLAG IN I/O AREA FOR GOOD RECORD
JE E2
PDUPDT
H4 UPDATE CURRENT RECORD POINTER
J4 END OF BUFFER
YES
NO
JB B1
PSSCAW
PDFSTM
C5 FIRST TIME SWITCH
ON
OFF
D5 TURN ON FIRST TIME SWITCH
E5 FIRST TIME SINCE OPEN
NO
YES
F5 TURN ON FIRST TIME SINCE OPEN SWITCH
J5 GET POINTER TO FIRST I/O AREA, REMOVE HIGH-ORDER BYTE

Chart JB. $$A$SUP1 - SMICR Macro, External Interrupt (Part 2 of 3)
Refer to Chart 05.

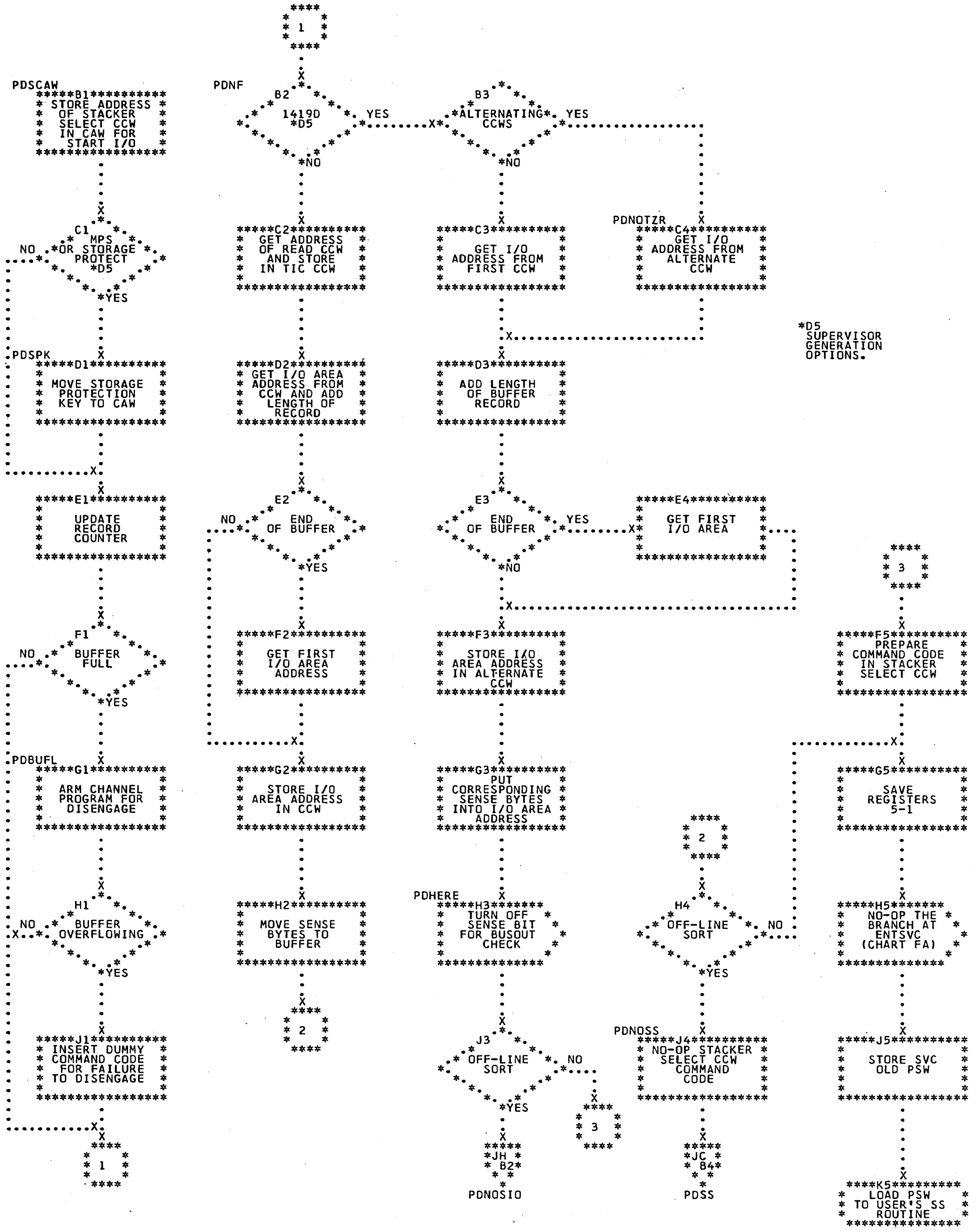

1
PDSCAW
B1
STORE ADDRESS OF STACKER SELECT CCW IN CAW FOR START I/O
PDNF
B2
1419D *D5
YES
NO
B3
ALTERNATING CCWS
YES
NO
C1
MPS OR STORAGE PROTECT *D5
NO
YES
C2
GET ADDRESS OF READ CCW AND STORE IN TIC CCW
C3
GET I/O ADDRESS FROM FIRST CCW
PDNOTZR
C4
GET I/O ADDRESS FROM ALTERNATE CCW
*D5 SUPERVISOR GENERATION OPTIONS.
PDSPK
D1
MOVE STORAGE PROTECTION KEY TO CAW
D2
GET I/O AREA ADDRESS FROM CCW AND ADD LENGTH OF RECORD
D3
ADD LENGTH OF BUFFER RECORD
E1
UPDATE RECORD COUNTER
E2
END OF BUFFER
NO
YES
E3
END OF BUFFER
YES
NO
E4
GET FIRST I/O AREA
3
F1
BUFFER FULL
NO
YES
F2
GET FIRST I/O AREA ADDRESS
F3
STORE I/O AREA ADDRESS IN ALTERNATE CCW
F5
PREPARE COMMAND CODE IN STACKER SELECT CCW
PDBUFL
G1
ARM CHANNEL PROGRAM FOR DISENGAGE
G2
STORE I/O AREA ADDRESS IN CCW
G3
PUT CORRESPONDING SENSE BYTES INTO I/O AREA ADDRESS
G5
SAVE REGISTERS 5-1
2
H1
BUFFER OVERFLOWING
NO
YES
H2
MOVE SENSE BYTES TO BUFFER
PDHERE
H3
TURN OFF SENSE BIT FOR BUSOUT CHECK
H4
OFF-LINE SORT
NO
YES
H5
NO-OP THE BRANCH AT ENTSVC (CHART FA)
2
J1
INSERT DUMMY COMMAND CODE FOR FAILURE TO DISENGAGE
J3
OFF-LINE SORT
NO
YES
3
PDNOSS
J4
NO-OP STACKER SELECT CCW COMMAND CODE
J5
STORE SVC OLD PSW
1
JH B2
PDNOSIO
JC B4
PDSS
K5
LOAD PSW TO USER'S SS ROUTINE

Chart JC. $$A$SUP1 - SMICR Macro, External Interrupt (Part 3 of 3)
Refer to Chart 05.

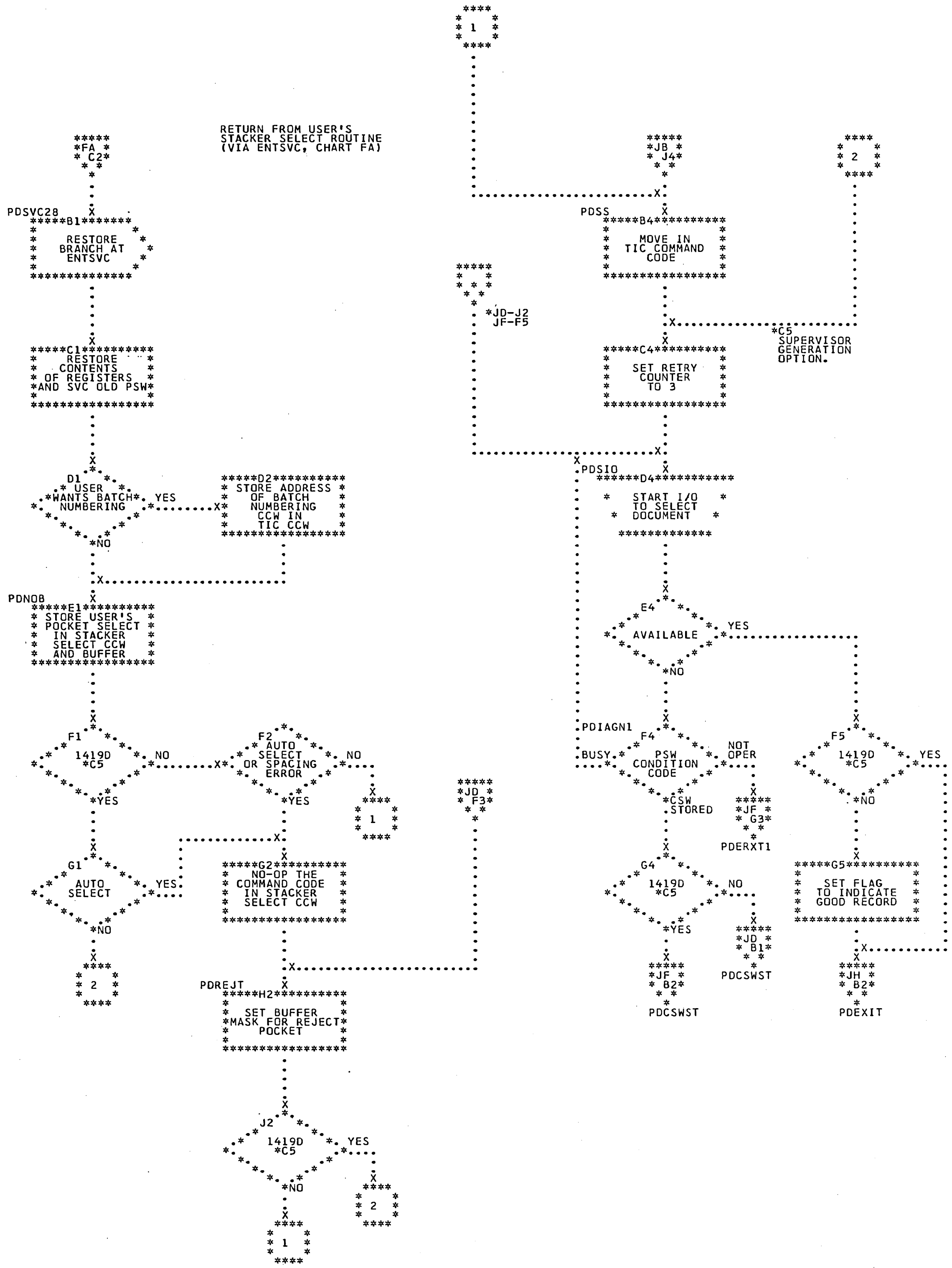
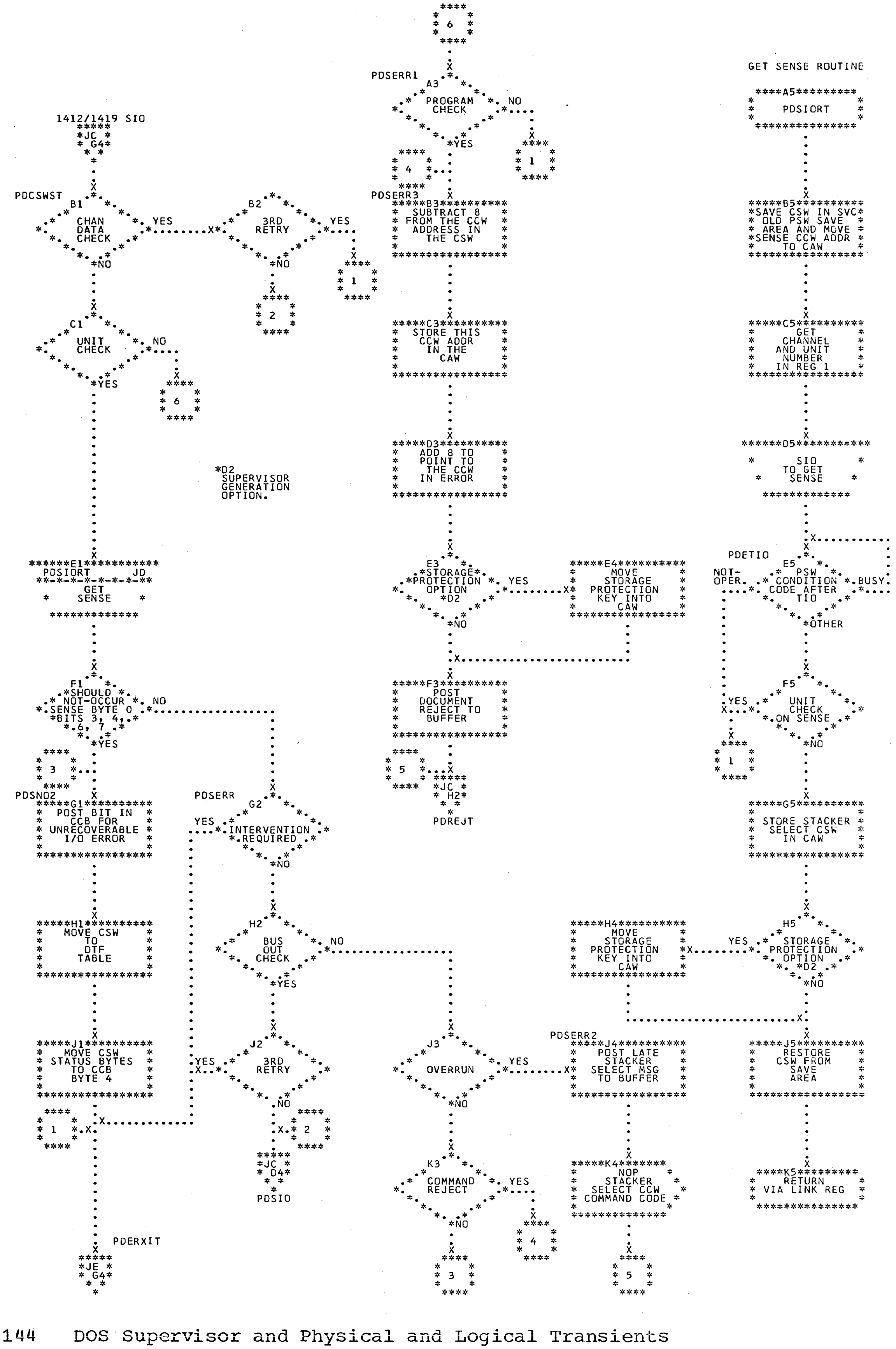

1
FA C2
RETURN FROM USER'S STACKER SELECT ROUTINE (VIA ENTSVC, CHART FA)
JB J4
2
PDSVC28
B1
RESTORE BRANCH AT ENTSVC
PDSS
B4
MOVE IN TIC COMMAND CODE
JD-J2
JF-F5
C1
RESTORE CONTENTS OF REGISTERS AND SVC OLD PSW
C4
SET RETRY COUNTER TO 3
C5
SUPERVISOR GENERATION OPTION.
D1
USER WANTS BATCH NUMBERING
YES
NO
D2
STORE ADDRESS OF BATCH NUMBERING CCW IN TIC CCW
PDSIO
D4
START I/O TO SELECT DOCUMENT
PDNOB
E1
STORE USER'S POCKET SELECT IN STACKER SELECT CCW AND BUFFER
E4
AVAILABLE
YES
NO
F1
1419D C5
NO
YES
F2
AUTO SELECT OR SPACING ERROR
NO
YES
1
JD F3
PDIAGN1
F4
BUSY
PSW CONDITION CODE
NOT OPER
CSW STORED
JF G3
PDERXT1
F5
1419D C5
YES
NO
G1
AUTO SELECT
YES
NO
2
G2
NO-OP THE COMMAND CODE IN STACKER SELECT CCW
G4
1419D C5
NO
YES
JD B1
PDCSWST
JF B2
PDCSWST
G5
SET FLAG TO INDICATE GOOD RECORD
JH B2
PDEXIT
PDREJT
H2
SET BUFFER MASK FOR REJECT POCKET
J2
1419D C5
YES
NO
2
1

Chart JD. $$A$SUP1 - SMICR Macro, ERP for 1412/1419 Start I/O
Refer to Chart 05.



6
PDSERR1
A3
PROGRAM CHECK
NO
YES
1
4
GET SENSE ROUTINE
A5
PDSIORT
1412/1419 SIO
JC G4
PDCSWST
B1
CHAN DATA CHECK
YES
NO
B2
3RD RETRY
YES
NO
1
2
PDSERR3
B3
SUBTRACT 8 FROM THE CCW ADDRESS IN THE CSW
B5
SAVE CSW IN SVC OLD PSW SAVE AREA AND MOVE SENSE CCW ADDR TO CAW
C1
UNIT CHECK
NO
YES
6
C3
STORE THIS CCW ADDR IN THE CAW
C5
GET CHANNEL AND UNIT NUMBER IN REG 1
D3
ADD 8 TO POINT TO THE CCW IN ERROR
D5
SIO TO GET SENSE
*D2 SUPERVISOR GENERATION OPTION.
E1
PDSIORT JD
GET SENSE
E3
STORAGE PROTECTION OPTION *D2
YES
NO
E4
MOVE STORAGE PROTECTION KEY INTO CAW
PDETIO
E5
NOT-OPER.
PSW CONDITION CODE AFTER TIO
BUSY
OTHER
F1
SHOULD NOT-OCCUR SENSE BYTE 0 BITS 3, 4, 6, 7
NO
YES
3
F3
POST DOCUMENT REJECT TO BUFFER
5
JC H2
PDREJT
F5
UNIT CHECK ON SENSE
YES
NO
1
PDSNO2
G1
POST BIT IN CCB FOR UNRECOVERABLE I/O ERROR
PDSERR
G2
INTERVENTION REQUIRED
YES
NO
G5
STORE STACKER SELECT CSW IN CAW
H1
MOVE CSW TO DTF TABLE
H2
BUS OUT CHECK
NO
YES
H4
MOVE STORAGE PROTECTION KEY INTO CAW
H5
STORAGE PROTECTION OPTION *D2
YES
NO
J1
MOVE CSW STATUS BYTES TO CCB BYTE 4
J2
3RD RETRY
YES
NO
J3
OVERRUN
YES
PDSERR2
J4
POST LATE STACKER SELECT MSG TO BUFFER
J5
RESTORE CSW FROM SAVE AREA
1
2
JC D4
PDSIO
K3
COMMAND REJECT
YES
NO
4
K4
NOP STACKER SELECT CCW COMMAND CODE
K5
RETURN VIA LINK REG
PDERXIT
JE G4
3
5

Chart JE. $$A$SUP1 - SMICR Macro, ERP for 1412/1419 Test I/O
Refer to Chart 05.

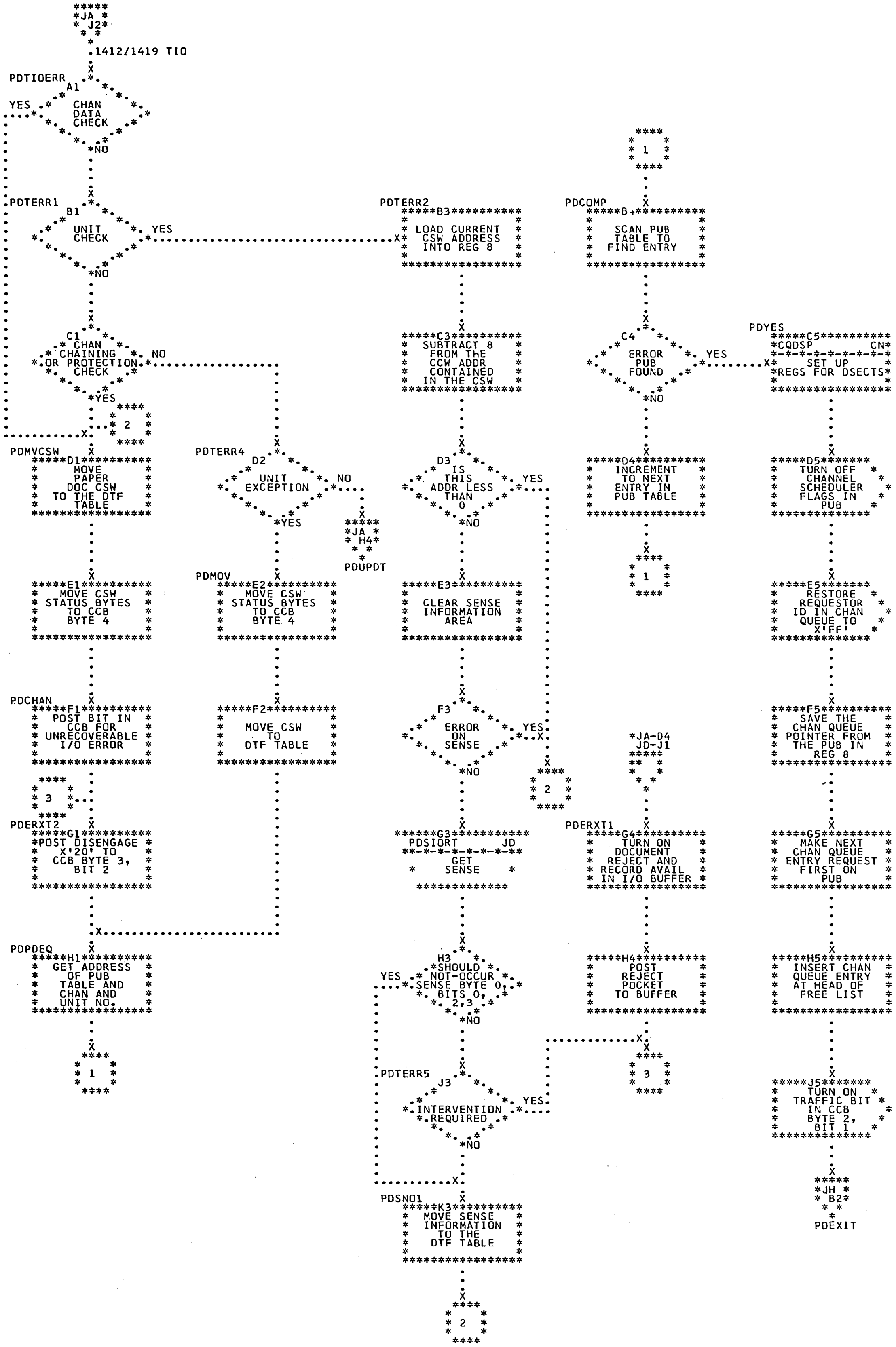
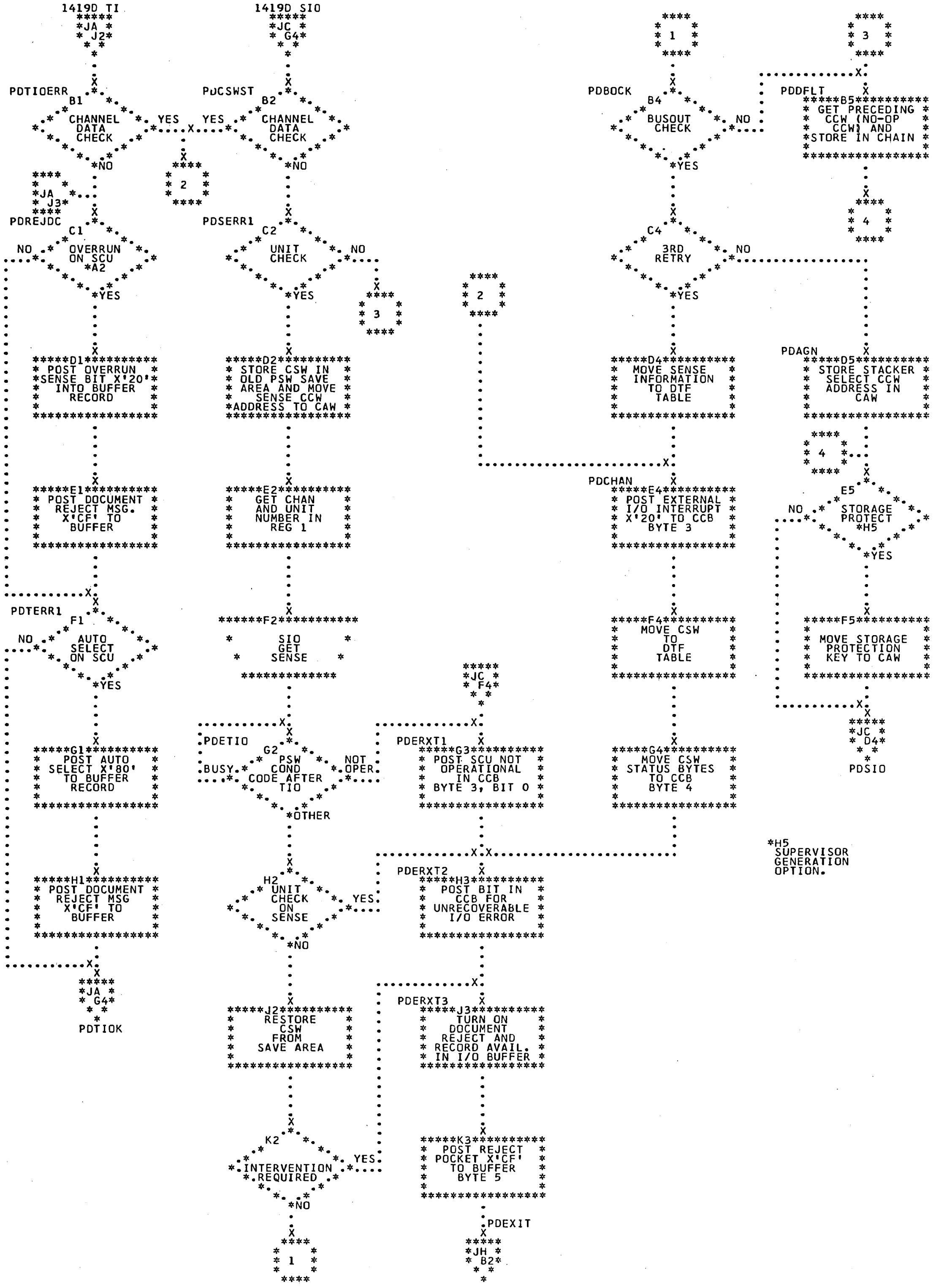
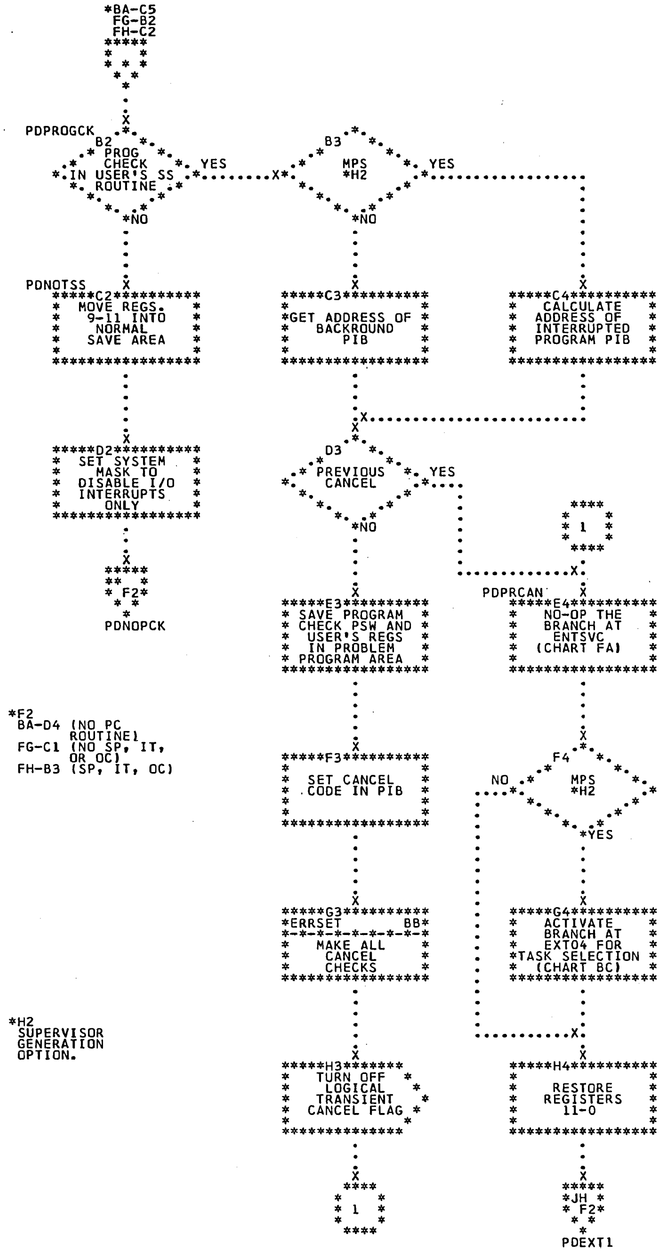

JA J2
1412/1419 TIO
PDTIOERR A1 CHAN DATA CHECK
YES
NO
PDTERR1 B1 UNIT CHECK
YES
NO
C1 CHAN CHAINING OR PROTECTION CHECK
NO
YES
2
PDMVCSW D1 MOVE PAPER DOC CSW TO THE DTF TABLE
E1 MOVE CSW STATUS BYTES TO CCB BYTE 4
PDCHAN F1 POST BIT IN CCB FOR UNRECOVERABLE I/O ERROR
3
PDERXT2 G1 POST DISENGAGE X'20' TO CCB BYTE 3, BIT 2
PDPDEQ H1 GET ADDRESS OF PUB TABLE AND CHAN AND UNIT NO.
1
PDTERR4 D2 UNIT EXCEPTION
NO
YES
JA H4 PDUPDT
PDMOV E2 MOVE CSW STATUS BYTES TO CCB BYTE 4
F2 MOVE CSW TO DTF TABLE
PDTERR2 B3 LOAD CURRENT CSW ADDRESS INTO REG 8
C3 SUBTRACT 8 FROM THE CCW ADDR CONTAINED IN THE CSW
D3 IS THIS ADDR LESS THAN 0
YES
NO
E3 CLEAR SENSE INFORMATION AREA
F3 ERROR ON SENSE
YES
NO
2
G3 PDSIORT JD GET SENSE
H3 SHOULD NOT-OCCUR SENSE BYTE 0, BITS 0, 2,3
YES
NO
PDTERR5 J3 INTERVENTION REQUIRED
YES
NO
PDSNO1 K3 MOVE SENSE INFORMATION TO THE DTF TABLE
2
1
PDCOMP B4 SCAN PUB TABLE TO FIND ENTRY
C4 ERROR PUB FOUND
YES
NO
D4 INCREMENT TO NEXT ENTRY IN PUB TABLE
1
JA-D4 JD-J1
PDERXT1 G4 TURN ON DOCUMENT REJECT AND RECORD AVAIL IN I/O BUFFER
H4 POST REJECT POCKET TO BUFFER
3
PDYES C5 CQDSP CN SET UP REGS FOR DSECTS
D5 TURN OFF CHANNEL SCHEDULER FLAGS IN PUB
E5 RESTORE REQUESTOR ID IN CHAN QUEUE TO X'FF'
F5 SAVE THE CHAN QUEUE POINTER FROM THE PUB IN REG 8
G5 MAKE NEXT CHAN QUEUE ENTRY REQUEST FIRST ON PUB
H5 INSERT CHAN QUEUE ENTRY AT HEAD OF FREE LIST
J5 TURN ON TRAFFIC BIT IN CCB BYTE 2, BIT 1
JH B2 PDEXIT

Chart JF. $$A$SUP1 - SMICR Macro, ERP for 1419D SIO and TIO
Refer to Chart 05.



1419D TI
JA J2
1419D SIO
JC G4
1
3
PDTIOERR
B1
CHANNEL DATA CHECK
YES
NO
PDCSWST
B2
CHANNEL DATA CHECK
YES
NO
2
PDBOCK
B4
BUSOUT CHECK
NO
YES
PDDFLT
B5
GET PRECEDING CCW (NO-OP CCW) AND STORE IN CHAIN
4
JA J3
PDREJDC
C1
OVERRUN ON SCU *A2
NO
YES
PDSERR1
C2
UNIT CHECK
NO
YES
3
2
C4
3RD RETRY
NO
YES
D1
POST OVERRUN SENSE BIT X'20' INTO BUFFER RECORD
D2
STORE CSW IN OLD PSW SAVE AREA AND MOVE SENSE CCW ADDRESS TO CAW
D4
MOVE SENSE INFORMATION TO DTF TABLE
PDAGN
D5
STORE STACKER SELECT CCW ADDRESS IN CAW
4
E1
POST DOCUMENT REJECT MSG. X'CF' TO BUFFER
E2
GET CHAN AND UNIT NUMBER IN REG 1
PDCHAN
E4
POST EXTERNAL I/O INTERRUPT X'20' TO CCB BYTE 3
E5
STORAGE PROTECT *H5
NO
YES
PDTERR1
F1
AUTO SELECT ON SCU
NO
YES
F2
SIO GET SENSE
F4
MOVE CSW TO DTF TABLE
F5
MOVE STORAGE PROTECTION KEY TO CAW
JC F4
JC D4
PDSIO
G1
POST AUTO SELECT X'80' TO BUFFER RECORD
PDETIO
G2
PSW COND CODE AFTER TIO
BUSY
NOT OPER
OTHER
PDERXT1
G3
POST SCU NOT OPERATIONAL IN CCB BYTE 3, BIT 0
G4
MOVE CSW STATUS BYTES TO CCB BYTE 4
*H5 SUPERVISOR GENERATION OPTION.
H1
POST DOCUMENT REJECT MSG X'CF' TO BUFFER
H2
UNIT CHECK ON SENSE
YES
NO
PDERXT2
H3
POST BIT IN CCB FOR UNRECOVERABLE I/O ERROR
JA G4
PDTIOK
J2
RESTORE CSW FROM SAVE AREA
PDERXT3
J3
TURN ON DOCUMENT REJECT AND RECORD AVAIL. IN I/O BUFFER
K2
INTERVENTION REQUIRED
YES
NO
K3
POST REJECT POCKET X'CF' TO BUFFER BYTE 5
PDEXIT
1
JH B2

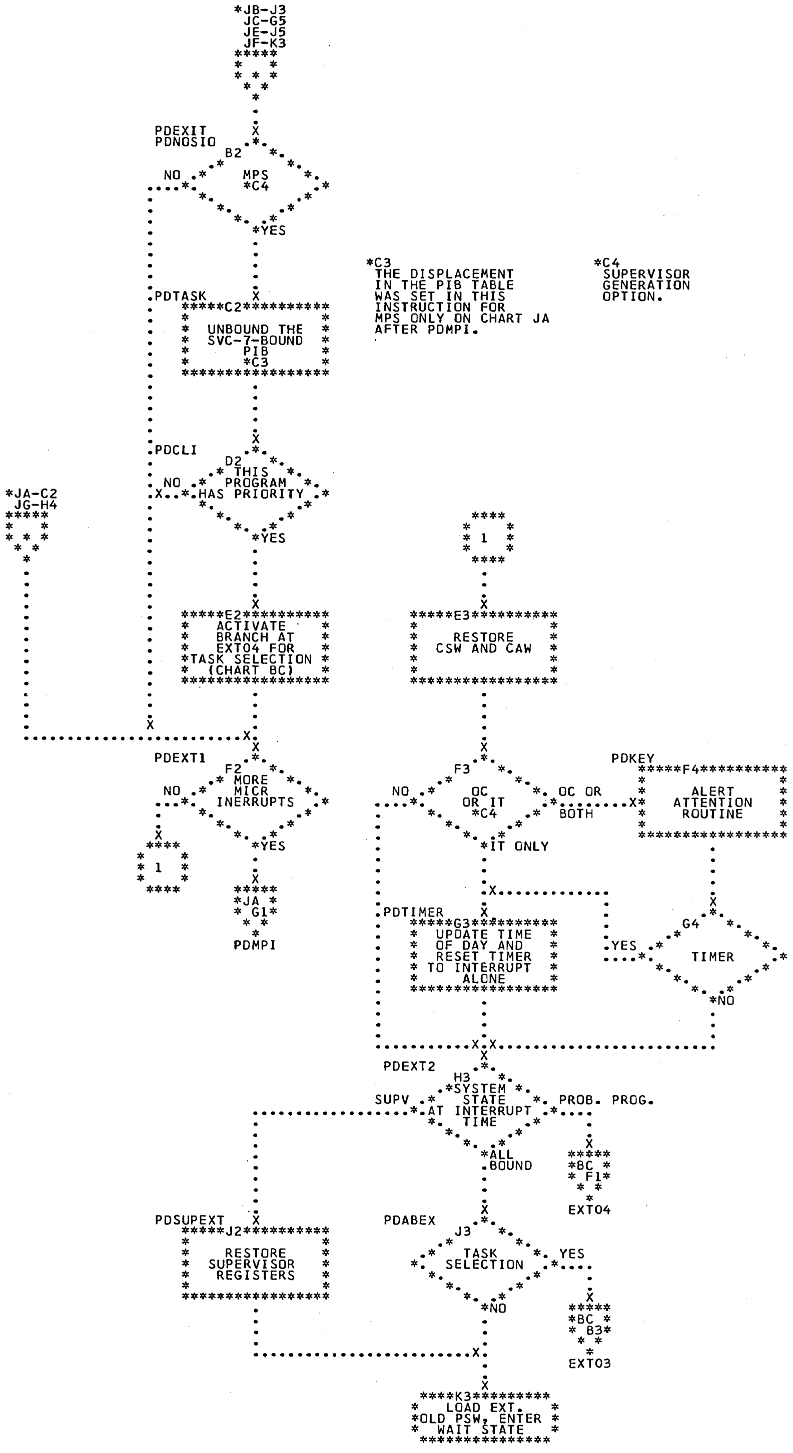
Chart JG. $$A$SUP1 - SMICR Macro, Program Check Interrupt
Refer to Chart 03.


BA-C5
FG-B2
FH-C2
PDPROGCK
B2
PROG CHECK IN USER'S SS ROUTINE
YES
NO
B3
MPS *H2
YES
NO
PDNOTSS
C2
MOVE REGS. 9-11 INTO NORMAL SAVE AREA
C3
GET ADDRESS OF BACKROUND PIB
C4
CALCULATE ADDRESS OF INTERRUPTED PROGRAM PIB
D2
SET SYSTEM MASK TO DISABLE I/O INTERRUPTS ONLY
D3
PREVIOUS CANCEL
YES
NO
1
F2
PDNOPCK
PDPRCAN
E3
SAVE PROGRAM CHECK PSW AND USER'S REGS IN PROBLEM PROGRAM AREA
E4
NO-OP THE BRANCH AT ENTSVC (CHART FA)
F3
SET CANCEL CODE IN PIB
F4
NO
MPS *H2
YES
G3
ERRSET BB
MAKE ALL CANCEL CHECKS
G4
ACTIVATE BRANCH AT EXT04 FOR TASK SELECTION (CHART BC)
H3
TURN OFF LOGICAL TRANSIENT CANCEL FLAG
H4
RESTORE REGISTERS 11-0
1
JH F2
PDEXT1

*F2
BA-D4 (NO PC ROUTINE)
FG-C1 (NO SP, IT, OR OC)
FH-B3 (SP, IT, OC)

*H2
SUPERVISOR GENERATION OPTION.

Chart JH. $$A$SUP1 - SMICR Macro, Exit Routine
Refer to Charts 03 and 05.

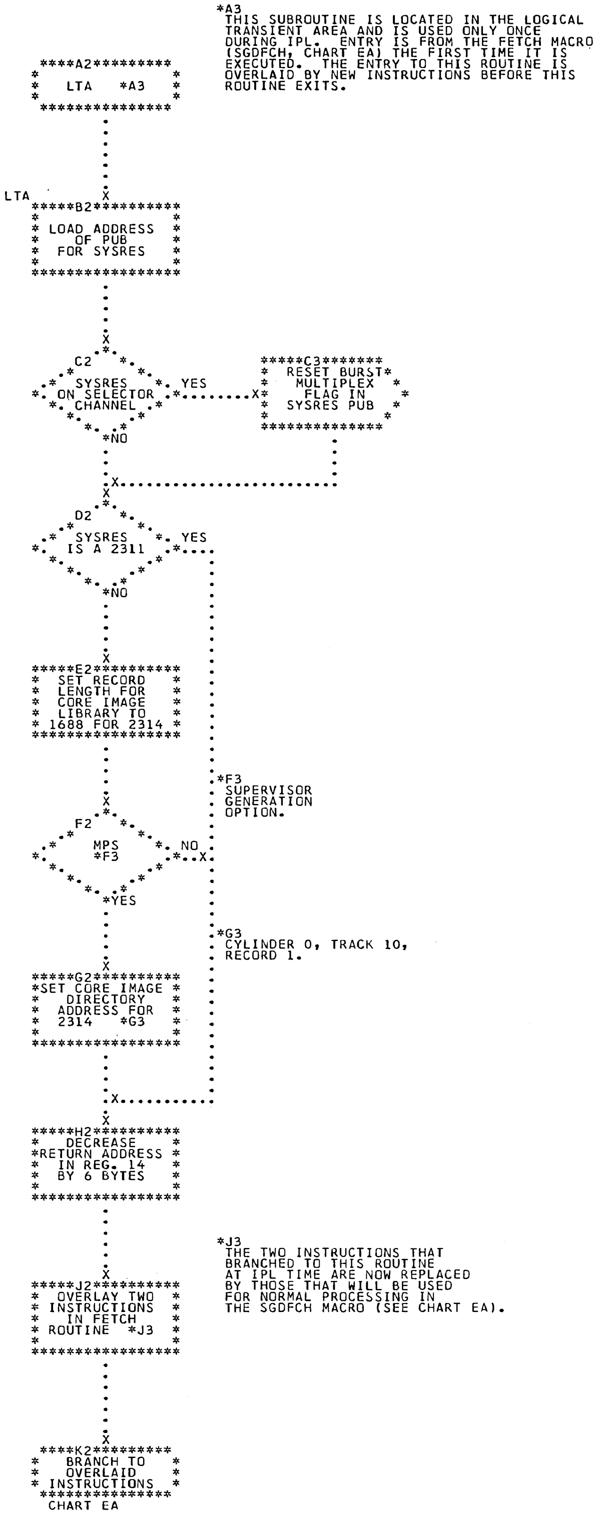


*JB-J3 JC-G5 JE-J5 JF-K3
PDEXIT PDNOSIO
B2 MPS *C4
NO
YES
PDTASK
C2 UNBOUND THE SVC-7-BOUND PIB *C3
PDCLI
D2 THIS PROGRAM HAS PRIORITY
NO
YES
*JA-C2 JG-H4
E2 ACTIVATE BRANCH AT EXT04 FOR TASK SELECTION (CHART BC)
1
E3 RESTORE CSW AND CAW
PDEXT1
F2 MORE MICR INTERRUPTS
NO
YES
1
JA G1
PDMPI
F3 OC OR IT *C4
NO
OC OR BOTH
IT ONLY
PDKEY
F4 ALERT ATTENTION ROUTINE
PDTIMER
G3 UPDATE TIME OF DAY AND RESET TIMER TO INTERRUPT ALONE
G4 TIMER
YES
NO
PDEXT2
H3 SYSTEM STATE AT INTERRUPT TIME
SUPV
PROB. PROG.
ALL BOUND
BC F1
EXT04
PDSUPEXT
J2 RESTORE SUPERVISOR REGISTERS
PDABEX
J3 TASK SELECTION
YES
NO
BC B3
EXT03
K3 LOAD EXT. OLD PSW, ENTER WAIT STATE

*C3
THE DISPLACEMENT
IN THE PIB TABLE
WAS SET IN THIS
INSTRUCTION FOR
MPS ONLY ON CHART JA
AFTER PDMPI.

*C4
SUPERVISOR
GENERATION
OPTION.

Chart KA. $$A$SUP1 - SEND Macro, LTA Subroutine


A2
LTA *A3
LTA
B2
LOAD ADDRESS OF PUB FOR SYSRES
C2
SYSRES ON SELECTOR CHANNEL
YES
NO
C3
RESET BURST MULTIPLEX FLAG IN SYSRES PUB
D2
SYSRES IS A 2311
YES
NO
E2
SET RECORD LENGTH FOR CORE IMAGE LIBRARY TO 1688 FOR 2314
F2
MPS *F3
NO
YES
G2
SET CORE IMAGE DIRECTORY ADDRESS FOR 2314 *G3
H2
DECREASE RETURN ADDRESS IN REG. 14 BY 6 BYTES
J2
OVERLAY TWO INSTRUCTIONS IN FETCH ROUTINE *J3
K2
BRANCH TO OVERLAID INSTRUCTIONS
CHART EA

*A3
THIS SUBROUTINE IS LOCATED IN THE LOGICAL TRANSIENT AREA AND IS USED ONLY ONCE DURING IPL. ENTRY IS FROM THE FETCH MACRO (SGDFCH, CHART EA) THE FIRST TIME IT IS EXECUTED. THE ENTRY TO THIS ROUTINE IS OVERLAID BY NEW INSTRUCTIONS BEFORE THIS ROUTINE EXITS.

*F3
SUPERVISOR GENERATION OPTION.

*G3
CYLINDER 0, TRACK 10, RECORD 1.

*J3
THE TWO INSTRUCTIONS THAT BRANCHED TO THIS ROUTINE AT IPL TIME ARE NOW REPLACED BY THOSE THAT WILL BE USED FOR NORMAL PROCESSING IN THE SGDFCH MACRO (SEE CHART EA).

Chart LA. $$ANERRA - ERP Monitor (Part 1 of 2)
Refer to Chart 07.

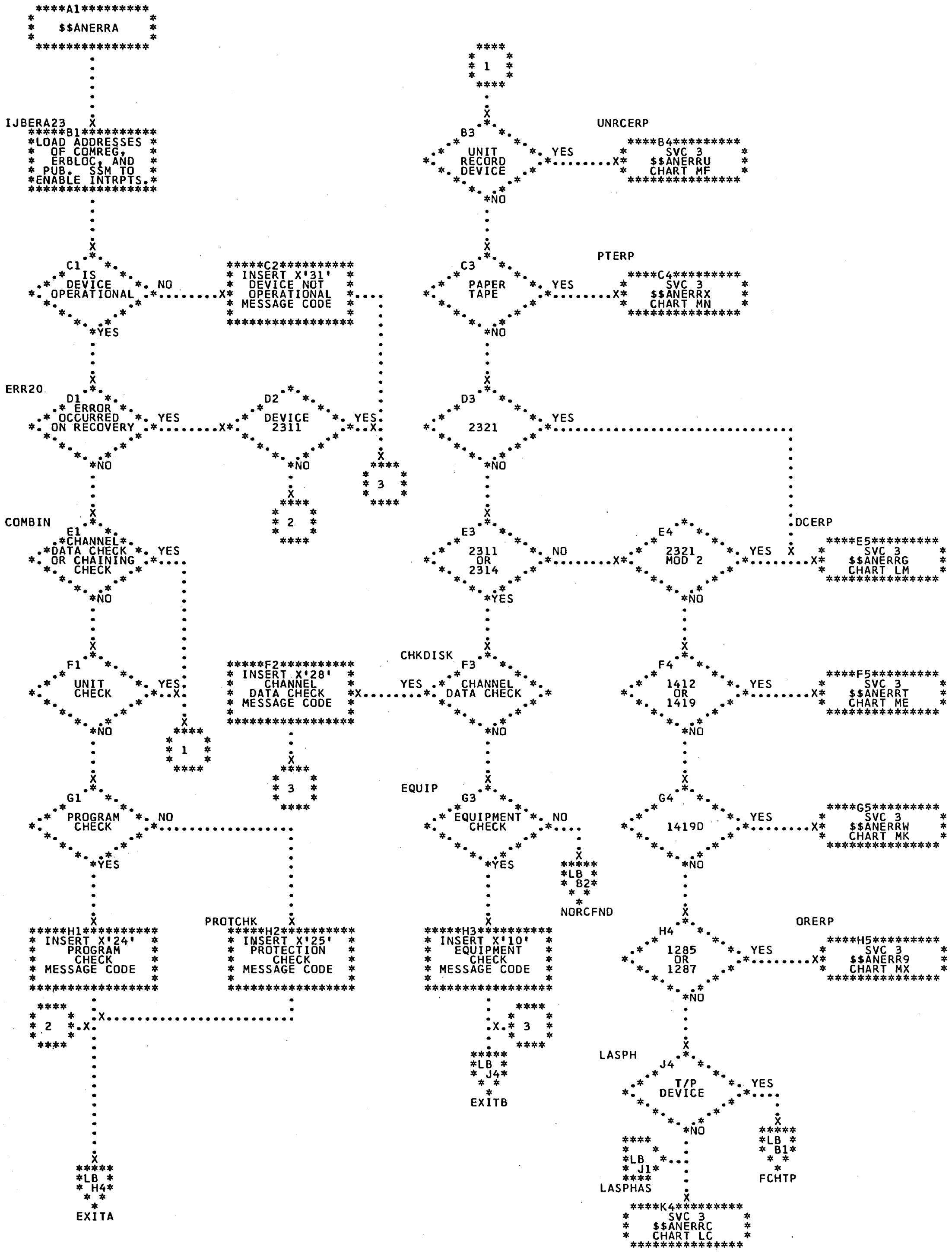


Chart LB. $$ANERRA - ERP Monitor (Part 2 of 2)
Refer to Chart 07.

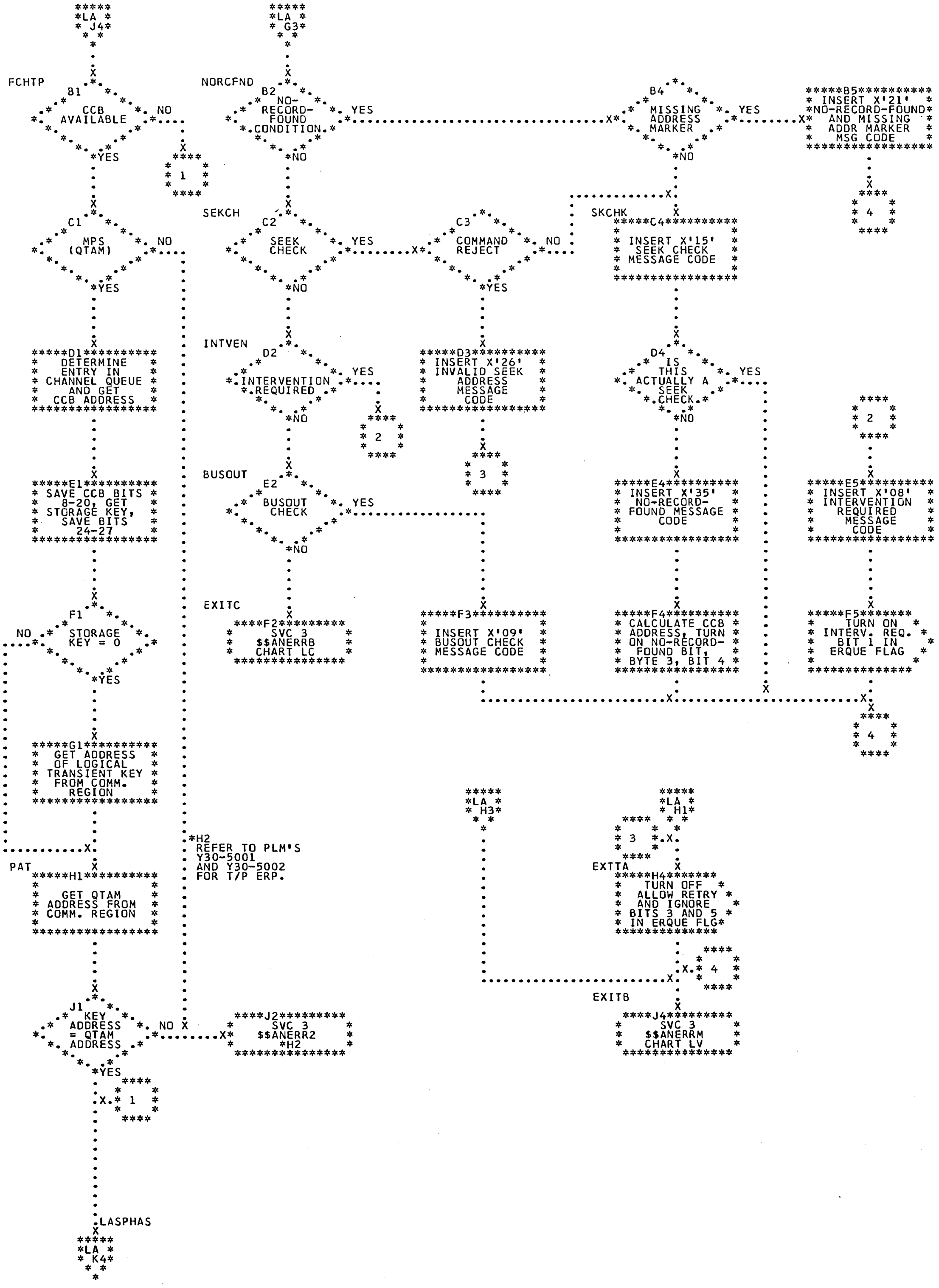

LA J4
FCHTP
B1
CCB AVAILABLE
NO
YES
1
C1
MPS (QTAM)
NO
YES
D1
DETERMINE ENTRY IN CHANNEL QUEUE AND GET CCB ADDRESS
E1
SAVE CCB BITS 8-20, GET STORAGE KEY, SAVE BITS 24-27
F1
STORAGE KEY = 0
NO
YES
G1
GET ADDRESS OF LOGICAL TRANSIENT KEY FROM COMM. REGION
PAT
H1
GET QTAM ADDRESS FROM COMM. REGION
J1
KEY ADDRESS = QTAM ADDRESS
NO
YES
1
LASPHAS
LA K4
*H2
REFER TO PLM'S Y30-5001 AND Y30-5002 FOR T/P ERP.
J2
SVC 3
$$ANERR2
*H2
LA G3
NORCFND
B2
NO-RECORD-FOUND CONDITION
YES
NO
SEKCH
C2
SEEK CHECK
YES
NO
INTVEN
D2
INTERVENTION REQUIRED
YES
NO
2
BUSOUT
E2
BUSOUT CHECK
YES
NO
EXITC
F2
SVC 3
$$ANERRB
CHART LC
C3
COMMAND REJECT
NO
YES
D3
INSERT X'26' INVALID SEEK ADDRESS MESSAGE CODE
3
F3
INSERT X'09' BUSOUT CHECK MESSAGE CODE
LA H3
B4
MISSING ADDRESS MARKER
YES
NO
SKCHK
C4
INSERT X'15' SEEK CHECK MESSAGE CODE
D4
IS THIS ACTUALLY A SEEK CHECK
YES
NO
E4
INSERT X'35' NO-RECORD-FOUND MESSAGE CODE
F4
CALCULATE CCB ADDRESS, TURN ON NO-RECORD-FOUND BIT, BYTE 3, BIT 4
LA H1
3
EXTTA
H4
TURN OFF ALLOW RETRY AND IGNORE BITS 3 AND 5 IN ERQUE FLG
4
EXITB
J4
SVC 3
$$ANERRM
CHART LV
B5
INSERT X'21' NO-RECORD-FOUND AND MISSING ADDR MARKER MSG CODE
4
2
E5
INSERT X'08' INTERVENTION REQUIRED MESSAGE CODE
F5
TURN ON INTERV. REQ. BIT 1 IN ERQUE FLAG
4

Chart LC. $$ANERRB - Disk ERP
Refer to Chart 07.

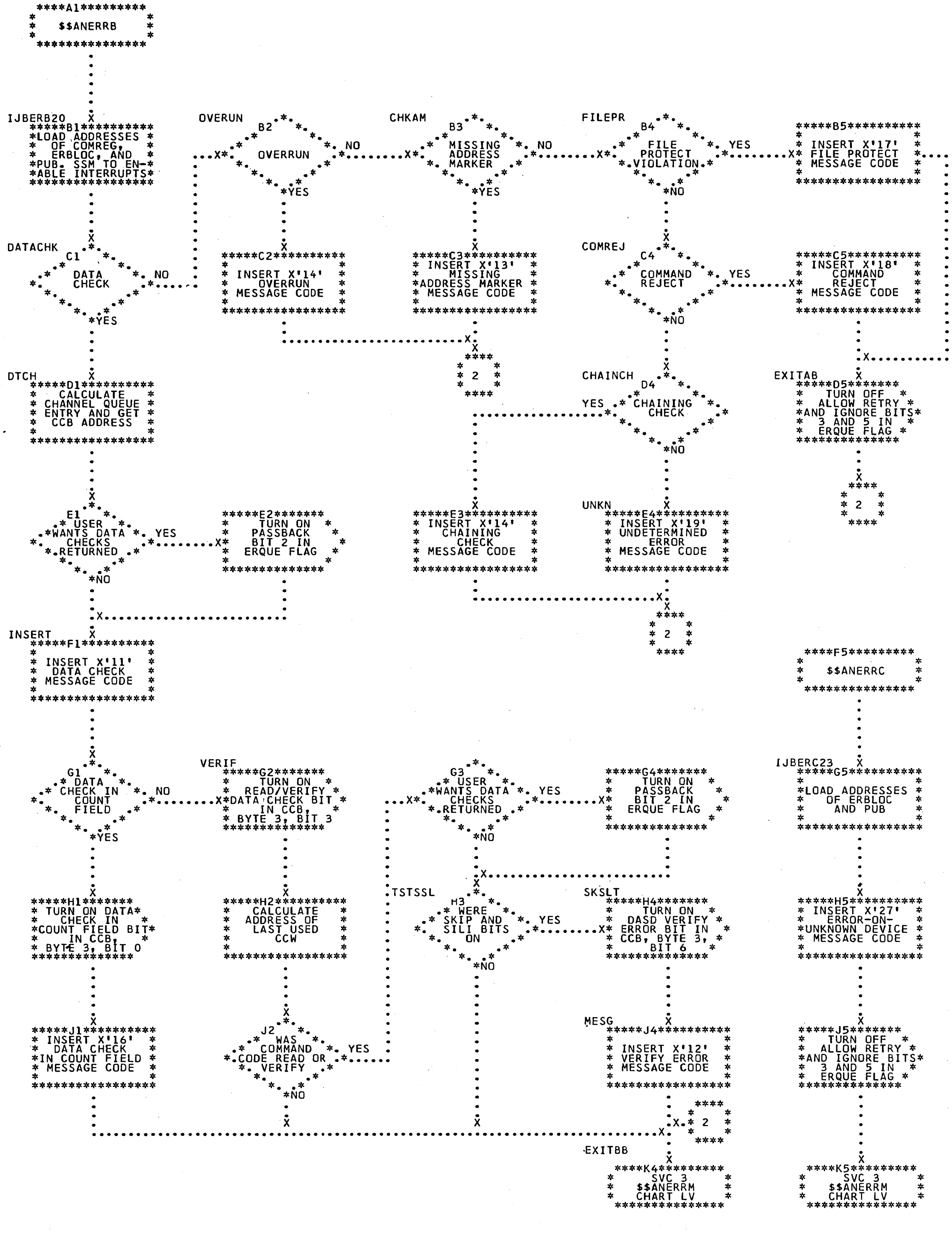

A1 $$ANERRB
IJBERB20
B1 LOAD ADDRESSES OF COMREG, ERBLOC, AND PUB. SSM TO ENABLE INTERRUPTS
OVERUN
B2 OVERRUN
NO
YES
CHKAM
B3 MISSING ADDRESS MARKER
NO
YES
FILEPR
B4 FILE PROTECT VIOLATION
YES
NO
B5 INSERT X'17' FILE PROTECT MESSAGE CODE
DATACHK
C1 DATA CHECK
NO
YES
C2 INSERT X'14' OVERRUN MESSAGE CODE
C3 INSERT X'13' MISSING ADDRESS MARKER MESSAGE CODE
COMREJ
C4 COMMAND REJECT
YES
NO
C5 INSERT X'18' COMMAND REJECT MESSAGE CODE
2
DTCH
D1 CALCULATE CHANNEL QUEUE ENTRY AND GET CCB ADDRESS
CHAINCH
D4 CHAINING CHECK
YES
NO
EXITAB
D5 TURN OFF ALLOW RETRY AND IGNORE BITS 3 AND 5 IN ERQUE FLAG
2
E1 USER WANTS DATA CHECKS RETURNED
YES
NO
E2 TURN ON PASSBACK BIT 2 IN ERQUE FLAG
E3 INSERT X'14' CHAINING CHECK MESSAGE CODE
UNKN
E4 INSERT X'19' UNDETERMINED ERROR MESSAGE CODE
2
INSERT
F1 INSERT X'11' DATA CHECK MESSAGE CODE
F5 $$ANERRC
G1 DATA CHECK IN COUNT FIELD
NO
YES
VERIF
G2 TURN ON READ/VERIFY DATA CHECK BIT IN CCB, BYTE 3, BIT 3
G3 USER WANTS DATA CHECKS RETURNED
YES
NO
G4 TURN ON PASSBACK BIT 2 IN ERQUE FLAG
IJBERC23
G5 LOAD ADDRESSES OF ERBLOC AND PUB
H1 TURN ON DATA CHECK IN COUNT FIELD BIT IN CCB, BYTE 3, BIT 0
H2 CALCULATE ADDRESS OF LAST USED CCW
TSTSSL
H3 WERE SKIP AND SILI BITS ON
YES
NO
SKSLT
H4 TURN ON DASD VERIFY ERROR BIT IN CCB, BYTE 3, BIT 6
H5 INSERT X'27' ERROR-ON-UNKNOWN DEVICE MESSAGE CODE
J1 INSERT X'16' DATA CHECK IN COUNT FIELD MESSAGE CODE
J2 WAS COMMAND CODE READ OR VERIFY
YES
NO
MESG
J4 INSERT X'12' VERIFY ERROR MESSAGE CODE
J5 TURN OFF ALLOW RETRY AND IGNORE BITS 3 AND 5 IN ERQUE FLAG
2
EXITBB
K4 SVC 3 $$ANERRM CHART LV
K5 SVC 3 $$ANERRM CHART LV

Chart LD. $$ANERRD - Phase 1 of Tape ERP (Part 1 of 2)
Refer to Chart 07.

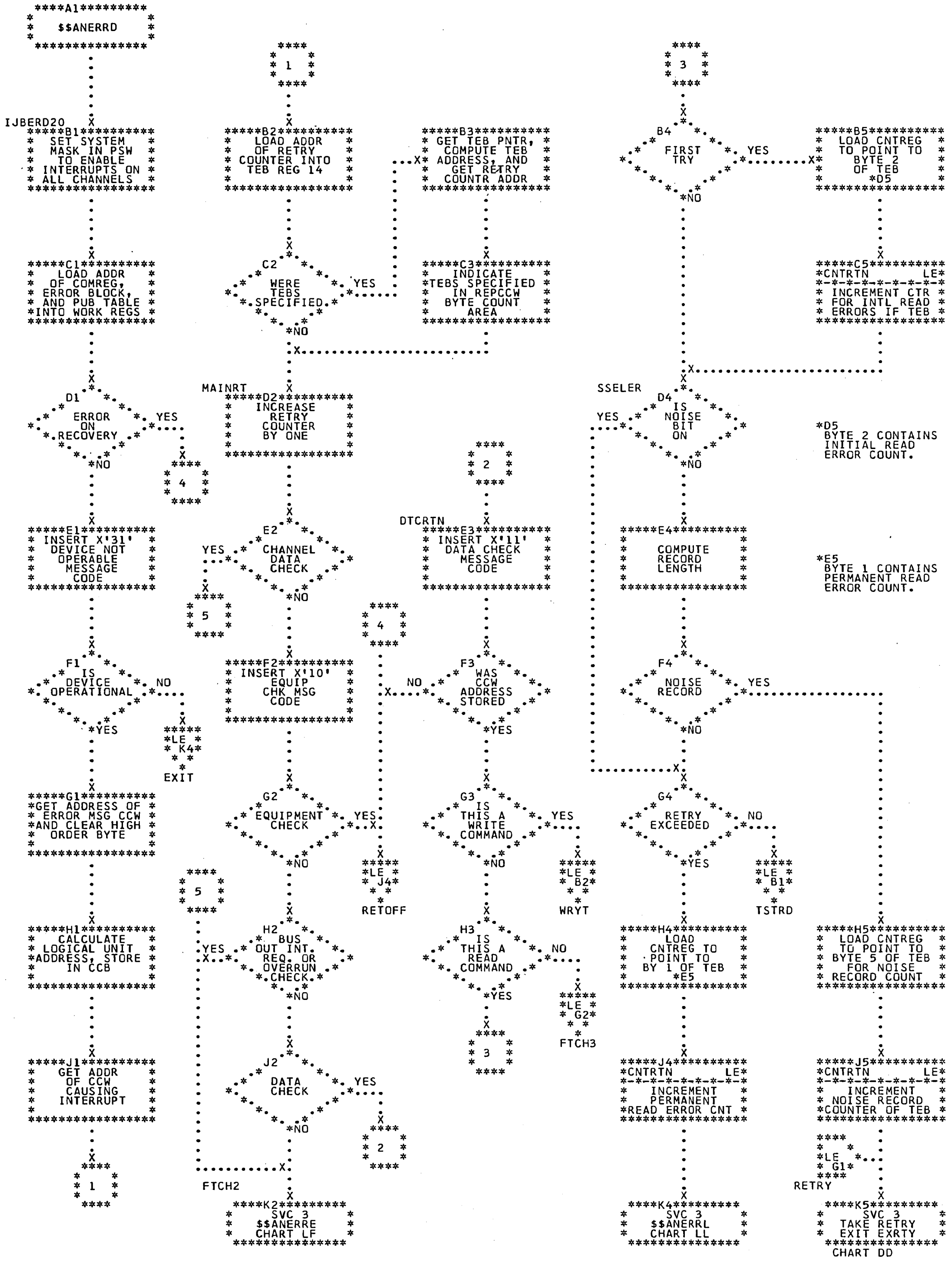
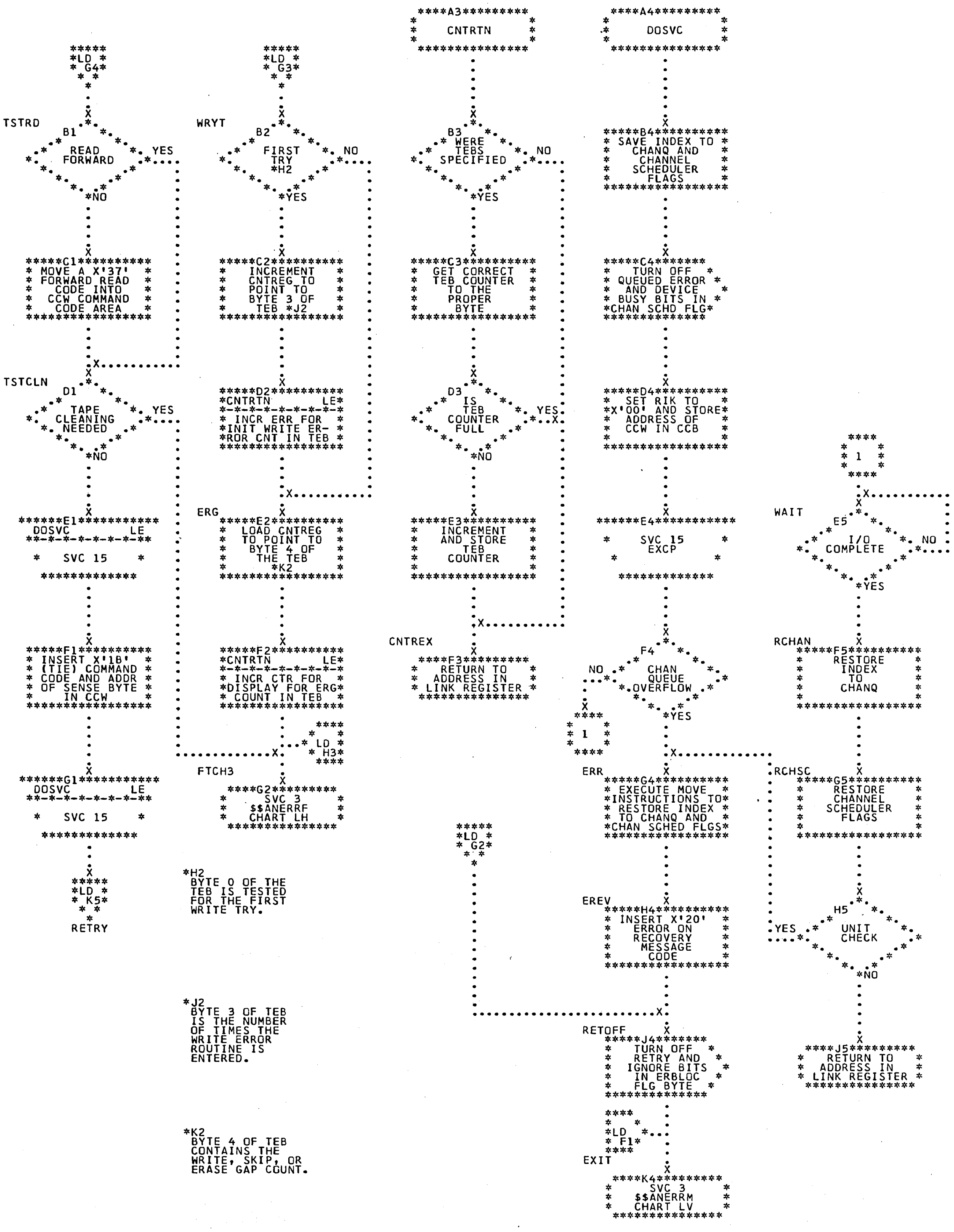

A1
$$ANERRD
IJBERD20
B1
SET SYSTEM MASK IN PSW TO ENABLE INTERRUPTS ON ALL CHANNELS
C1
LOAD ADDR OF COMREG, ERROR BLOCK, AND PUB TABLE INTO WORK REGS
D1
ERROR ON RECOVERY
YES
NO
4
E1
INSERT X'31' DEVICE NOT OPERABLE MESSAGE CODE
F1
IS DEVICE OPERATIONAL
NO
YES
LE K4
EXIT
G1
GET ADDRESS OF ERROR MSG CCW AND CLEAR HIGH ORDER BYTE
H1
CALCULATE LOGICAL UNIT ADDRESS, STORE IN CCB
J1
GET ADDR OF CCW CAUSING INTERRUPT
1
1
B2
LOAD ADDR OF RETRY COUNTER INTO TEB REG 14
C2
WERE TEBS SPECIFIED
YES
NO
MAINRT
D2
INCREASE RETRY COUNTER BY ONE
E2
CHANNEL DATA CHECK
YES
NO
5
F2
INSERT X'10' EQUIP CHK MSG CODE
G2
EQUIPMENT CHECK
YES
NO
LE J4
RETOFF
5
H2
BUS OUT INT. REQ. OR OVERRUN CHECK
YES
NO
J2
DATA CHECK
YES
NO
2
FTCH2
K2
SVC 3 $$ANERRE CHART LF
4
B3
GET TEB PNTR, COMPUTE TEB ADDRESS, AND GET RETRY COUNTR ADDR
C3
INDICATE TEBS SPECIFIED IN REPCCW BYTE COUNT AREA
2
DTCRTN
E3
INSERT X'11' DATA CHECK MESSAGE CODE
F3
WAS CCW ADDRESS STORED
NO
YES
G3
IS THIS A WRITE COMMAND
YES
NO
LE B2
WRYT
H3
IS THIS A READ COMMAND
NO
YES
LE G2
FTCH3
3
3
B4
FIRST TRY
YES
NO
SSELER
D4
IS NOISE BIT ON
YES
NO
E4
COMPUTE RECORD LENGTH
F4
NOISE RECORD
YES
NO
G4
RETRY EXCEEDED
NO
YES
LE B1
TSTRD
H4
LOAD CNTREG TO POINT TO BY 1 OF TEB *E5
J4
CNTRTN LE
INCREMENT PERMANENT READ ERROR CNT
K4
SVC 3 $$ANERRL CHART LL
B5
LOAD CNTREG TO POINT TO BYTE 2 OF TEB *D5
C5
CNTRTN LE
INCREMENT CTR FOR INTL READ ERRORS IF TEB
*D5
BYTE 2 CONTAINS INITIAL READ ERROR COUNT.
*E5
BYTE 1 CONTAINS PERMANENT READ ERROR COUNT.
H5
LOAD CNTREG TO POINT TO BYTE 5 OF TEB FOR NOISE RECORD COUNT
J5
CNTRTN LE
INCREMENT NOISE RECORD COUNTER OF TEB
LE G1
RETRY
K5
SVC 3 TAKE RETRY EXIT EXRTY
CHART DD

Chart LE. $$ANERRD - Phase 1 of Tape ERP (Part 2 of 2)
Refer to Chart 07.


A3 CNTRTN
A4 DOSVC
LD G4
TSTRD
B1 READ FORWARD
YES
NO
C1 MOVE A X'37' FORWARD READ CODE INTO CCW COMMAND CODE AREA
TSTCLN
D1 TAPE CLEANING NEEDED
YES
NO
E1 DOSVC LE
SVC 15
F1 INSERT X'1B' (TIE) COMMAND CODE AND ADDR OF SENSE BYTE IN CCW
G1 DOSVC LE
SVC 15
LD K5
RETRY
LD G3
WRYT
B2 FIRST TRY *H2
NO
YES
C2 INCREMENT CNTREG TO POINT TO BYTE 3 OF TEB *J2
D2 CNTRTN LE
INCR ERR FOR INIT WRITE ERROR CNT IN TEB
ERG
E2 LOAD CNTREG TO POINT TO BYTE 4 OF THE TEB *K2
F2 CNTRTN LE
INCR CTR FOR DISPLAY FOR ERG COUNT IN TEB
LD H3
FTCH3
G2 SVC 3 $$ANERRF CHART LH
B3 WERE TEBS SPECIFIED
NO
YES
C3 GET CORRECT TEB COUNTER TO THE PROPER BYTE
D3 IS TEB COUNTER FULL
YES
NO
E3 INCREMENT AND STORE TEB COUNTER
CNTREX
F3 RETURN TO ADDRESS IN LINK REGISTER
LD G2
B4 SAVE INDEX TO CHANQ AND CHANNEL SCHEDULER FLAGS
C4 TURN OFF QUEUED ERROR AND DEVICE BUSY BITS IN CHAN SCHD FLG
D4 SET RIK TO X'00' AND STORE ADDRESS OF CCW IN CCB
E4 SVC 15 EXCP
F4 CHAN QUEUE OVERFLOW
NO
YES
1
ERR
G4 EXECUTE MOVE INSTRUCTIONS TO RESTORE INDEX TO CHANQ AND CHAN SCHED FLGS
EREV
H4 INSERT X'20' ERROR ON RECOVERY MESSAGE CODE
RETOFF
J4 TURN OFF RETRY AND IGNORE BITS IN ERBLOC FLG BYTE
LD F1
EXIT
K4 SVC 3 $$ANERRM CHART LV
1
WAIT
E5 I/O COMPLETE
NO
YES
RCHAN
F5 RESTORE INDEX TO CHANQ
RCHSC
G5 RESTORE CHANNEL SCHEDULER FLAGS
H5 UNIT CHECK
YES
NO
J5 RETURN TO ADDRESS IN LINK REGISTER

*H2
BYTE 0 OF THE TEB IS TESTED FOR THE FIRST WRITE TRY.

*J2
BYTE 3 OF TEB IS THE NUMBER OF TIMES THE WRITE ERROR ROUTINE IS ENTERED.

*K2
BYTE 4 OF TEB CONTAINS THE WRITE, SKIP, OR ERASE GAP COUNT.

Chart LF. $$ANERRE - Phase 2 of Tape ERP (Part 1 of 2)
Refer to Chart 07.

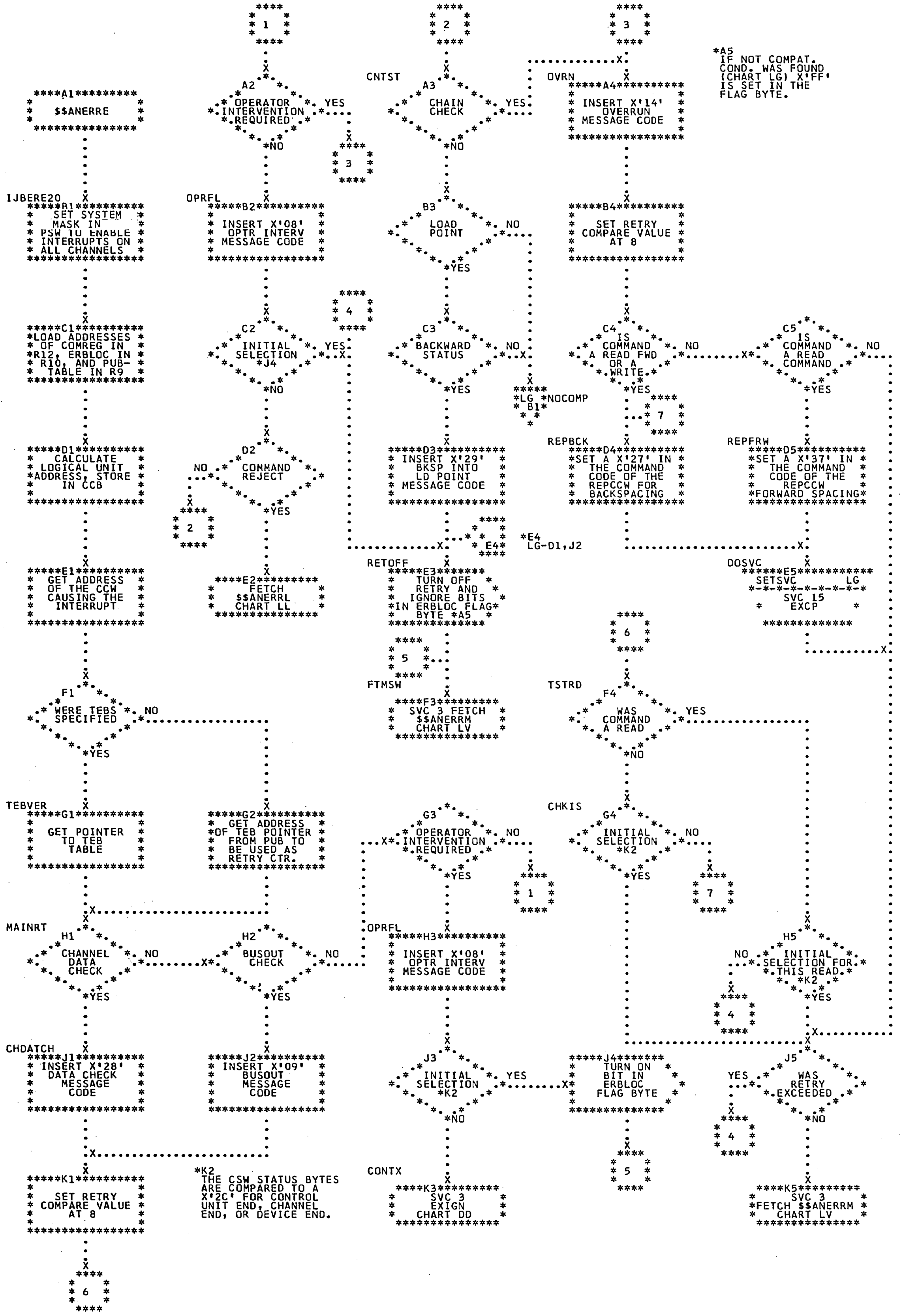

1
2
3
A1 $$ANERRE
A2 OPERATOR INTERVENTION REQUIRED YES NO
3
CNTST
A3 CHAIN CHECK YES NO
OVRN
A4 INSERT X'14' OVERRUN MESSAGE CODE
*A5 IF NOT COMPAT. COND. WAS FOUND (CHART LG) X'FF' IS SET IN THE FLAG BYTE.
IJBERE20
B1 SET SYSTEM MASK IN PSW TO ENABLE INTERRUPTS ON ALL CHANNELS
OPRFL
B2 INSERT X'08' OPTR INTERV MESSAGE CODE
B3 LOAD POINT NO YES
B4 SET RETRY COMPARE VALUE AT 8
C1 LOAD ADDRESSES OF COMREG IN R12, ERBLOC IN R10, AND PUB-TABLE IN R9
4
C2 INITIAL SELECTION *J4 YES NO
C3 BACKWARD STATUS NO YES
LG B1 NOCOMP
C4 IS COMMAND A READ FWD OR A WRITE NO YES
7
C5 IS COMMAND A READ COMMAND NO YES
D1 CALCULATE LOGICAL UNIT ADDRESS, STORE IN CCB
D2 COMMAND REJECT NO YES
2
D3 INSERT X'29' BKSP INTO LD POINT MESSAGE CODE
REPBCK
D4 SET A X'27' IN THE COMMAND CODE OF THE REPCCW FOR BACKSPACING
REPFRW
D5 SET A X'37' IN THE COMMAND CODE OF THE REPCCW FORWARD SPACING
E4
*E4 LG-D1,J2
E1 GET ADDRESS OF THE CCW CAUSING THE INTERRUPT
E2 FETCH $$ANERRL CHART LL
RETOFF
E3 TURN OFF RETRY AND IGNORE BITS IN ERBLOC FLAG BYTE *A5
DOSVC
E5 SETSVC LG SVC 15 EXCP
5
6
F1 WERE TEBS SPECIFIED NO YES
FTMSW
F3 SVC 3 FETCH $$ANERRM CHART LV
TSTRD
F4 WAS COMMAND A READ YES NO
TEBVER
G1 GET POINTER TO TEB TABLE
G2 GET ADDRESS OF TEB POINTER FROM PUB TO BE USED AS RETRY CTR.
G3 OPERATOR INTERVENTION REQUIRED NO YES
1
CHKIS
G4 INITIAL SELECTION *K2 NO YES
7
MAINRT
H1 CHANNEL DATA CHECK NO YES
H2 BUSOUT CHECK NO YES
OPRFL
H3 INSERT X'08' OPTR INTERV MESSAGE CODE
H5 INITIAL SELECTION FOR THIS READ. *K2 NO YES
4
CHDATCH
J1 INSERT X'28' DATA CHECK MESSAGE CODE
J2 INSERT X'09' BUSOUT MESSAGE CODE
J3 INITIAL SELECTION *K2 YES NO
J4 TURN ON BIT IN ERBLOC FLAG BYTE
J5 WAS RETRY EXCEEDED YES NO
4
5
K1 SET RETRY COMPARE VALUE AT 8
*K2 THE CSW STATUS BYTES ARE COMPARED TO A X'2C' FOR CONTROL UNIT END, CHANNEL END, OR DEVICE END.
CONTX
K3 SVC 3 EXIGN CHART DD
K5 SVC 3 FETCH $$ANERRM CHART LV
6

Chart LG. $$ANERRE - Phase 2 of Tape ERP (Part 2 of 2)
Refer to Chart 07.

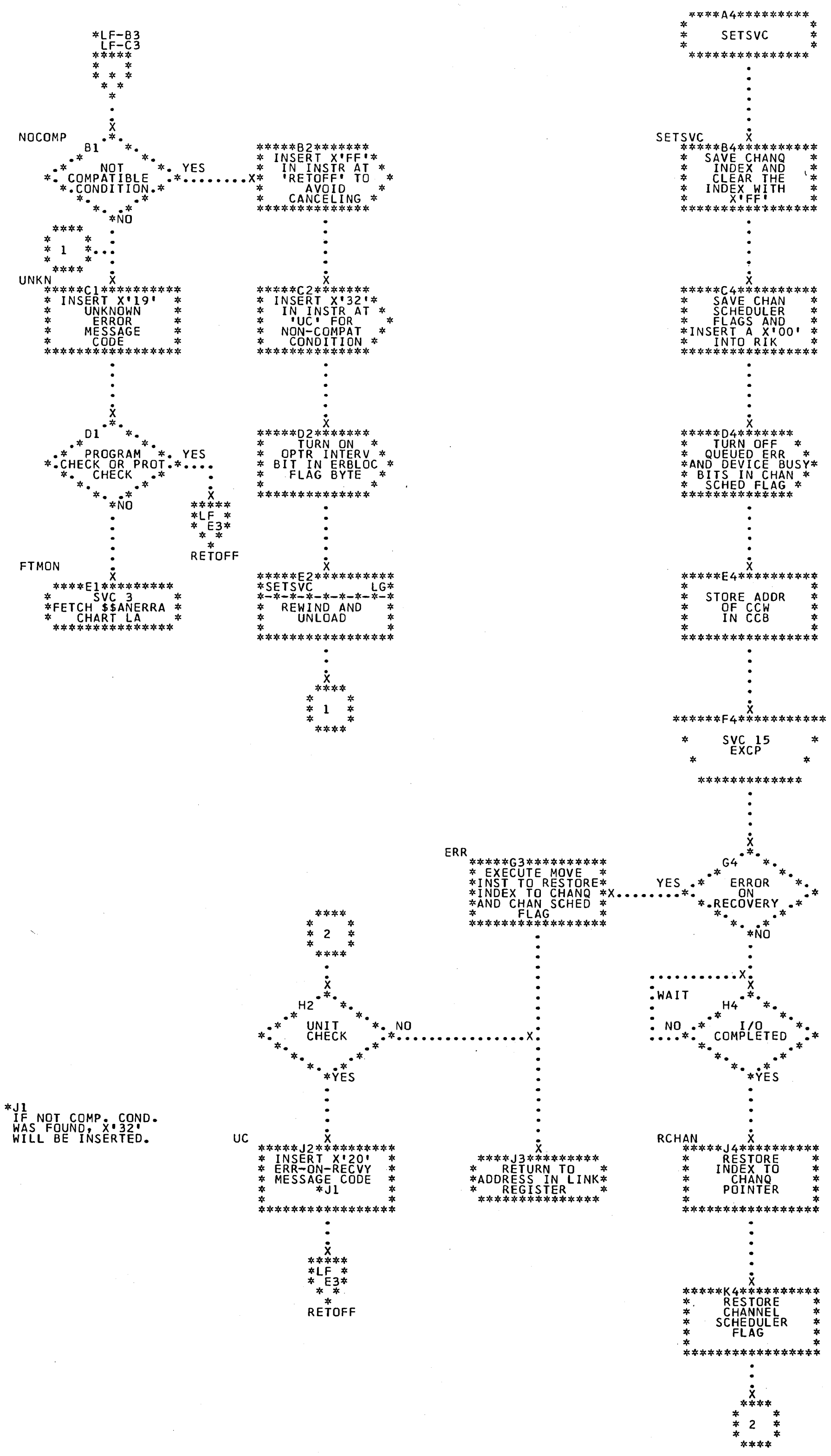

*LF-B3
LF-C3
NOCOMP
B1
NOT COMPATIBLE CONDITION
YES
NO
B2
INSERT X'FF' IN INSTR AT 'RETOFF' TO AVOID CANCELING
1
UNKN
C1
INSERT X'19' UNKNOWN ERROR MESSAGE CODE
C2
INSERT X'32' IN INSTR AT 'UC' FOR NON-COMPAT CONDITION
D1
PROGRAM CHECK OR PROT. CHECK
YES
NO
LF E3
RETOFF
D2
TURN ON OPTR INTERV BIT IN ERBLOC FLAG BYTE
FTMON
E1
SVC 3
FETCH $$ANERRA CHART LA
E2
SETSVC LG
REWIND AND UNLOAD
1
A4
SETSVC
SETSVC
B4
SAVE CHANQ INDEX AND CLEAR THE INDEX WITH X'FF'
C4
SAVE CHAN SCHEDULER FLAGS AND INSERT A X'00' INTO RIK
D4
TURN OFF QUEUED ERR AND DEVICE BUSY BITS IN CHAN SCHED FLAG
E4
STORE ADDR OF CCW IN CCB
F4
SVC 15
EXCP
ERR
G3
EXECUTE MOVE INST TO RESTORE INDEX TO CHANQ AND CHAN SCHED FLAG
G4
ERROR ON RECOVERY
YES
NO
2
H2
UNIT CHECK
NO
YES
WAIT
H4
I/O COMPLETED
NO
YES
*J1
IF NOT COMP. COND. WAS FOUND, X'32' WILL BE INSERTED.
UC
J2
INSERT X'20' ERR-ON-RECVY MESSAGE CODE
*J1
LF E3
RETOFF
J3
RETURN TO ADDRESS IN LINK REGISTER
RCHAN
J4
RESTORE INDEX TO CHANQ POINTER
K4
RESTORE CHANNEL SCHEDULER FLAG
2

Chart LH. $$ANERRF - Phase 3 of Tape ERP (Part 1 of 3)
Refer to Chart 07.

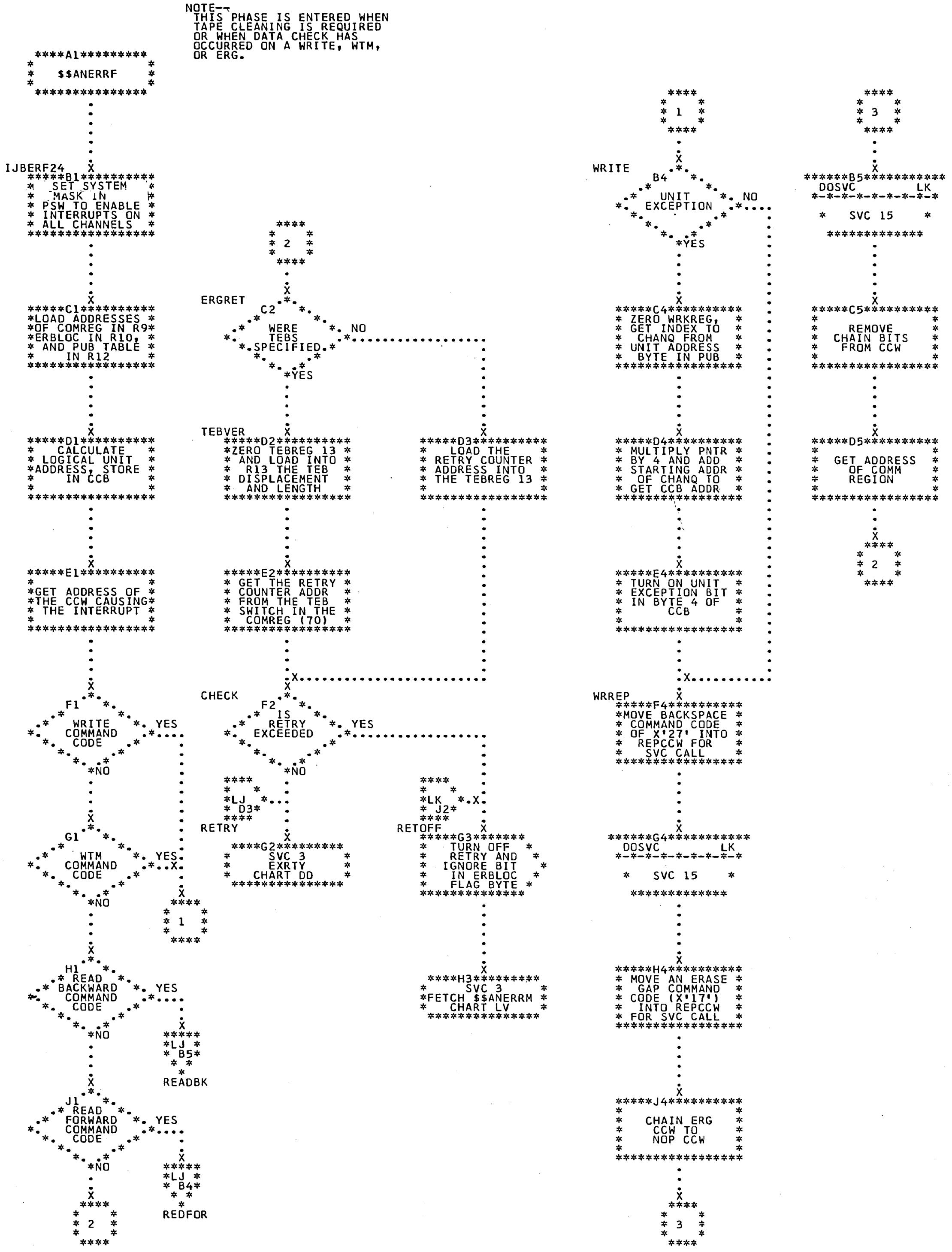

NOTE--
THIS PHASE IS ENTERED WHEN TAPE CLEANING IS REQUIRED OR WHEN DATA CHECK HAS OCCURRED ON A WRITE, WTM, OR ERG.
A1
$$ANERRF
IJBERF24
B1
SET SYSTEM MASK IN PSW TO ENABLE INTERRUPTS ON ALL CHANNELS
C1
LOAD ADDRESSES OF COMREG IN R9 ERBLOC IN R10, AND PUB TABLE IN R12
D1
CALCULATE LOGICAL UNIT ADDRESS, STORE IN CCB
E1
GET ADDRESS OF THE CCW CAUSING THE INTERRUPT
F1
WRITE COMMAND CODE
YES
NO
G1
WTM COMMAND CODE
YES
NO
1
H1
READ BACKWARD COMMAND CODE
YES
NO
LJ B5
READBK
J1
READ FORWARD COMMAND CODE
YES
NO
LJ B4
REDFOR
2
2
ERGRET
C2
WERE TEBS SPECIFIED
NO
YES
TEBVER
D2
ZERO TEBREG 13 AND LOAD INTO R13 THE TEB DISPLACEMENT AND LENGTH
E2
GET THE RETRY COUNTER ADDR FROM THE TEB SWITCH IN THE COMREG (70)
CHECK
F2
IS RETRY EXCEEDED
YES
NO
LJ D3
RETRY
G2
SVC 3
EXRTY
CHART DD
D3
LOAD THE RETRY COUNTER ADDRESS INTO THE TEBREG 13
LK J2
RETOFF
G3
TURN OFF RETRY AND IGNORE BIT IN ERBLOC FLAG BYTE
H3
SVC 3
FETCH $$ANERRM
CHART LV
1
WRITE
B4
UNIT EXCEPTION
NO
YES
C4
ZERO WRKREG, GET INDEX TO CHANQ FROM UNIT ADDRESS BYTE IN PUB
D4
MULTIPLY PNTR BY 4 AND ADD STARTING ADDR OF CHANQ TO GET CCB ADDR
E4
TURN ON UNIT EXCEPTION BIT IN BYTE 4 OF CCB
WRREP
F4
MOVE BACKSPACE COMMAND CODE OF X'27' INTO REPCCW FOR SVC CALL
G4
DOSVC LK
SVC 15
H4
MOVE AN ERASE GAP COMMAND CODE (X'17') INTO REPCCW FOR SVC CALL
J4
CHAIN ERG CCW TO NOP CCW
3
3
B5
DOSVC LK
SVC 15
C5
REMOVE CHAIN BITS FROM CCW
D5
GET ADDRESS OF COMM REGION
2

Chart LJ. $$ANERRF - Phase 3 of Tape ERP (Part 2 of 3)
Refer to Chart 07.

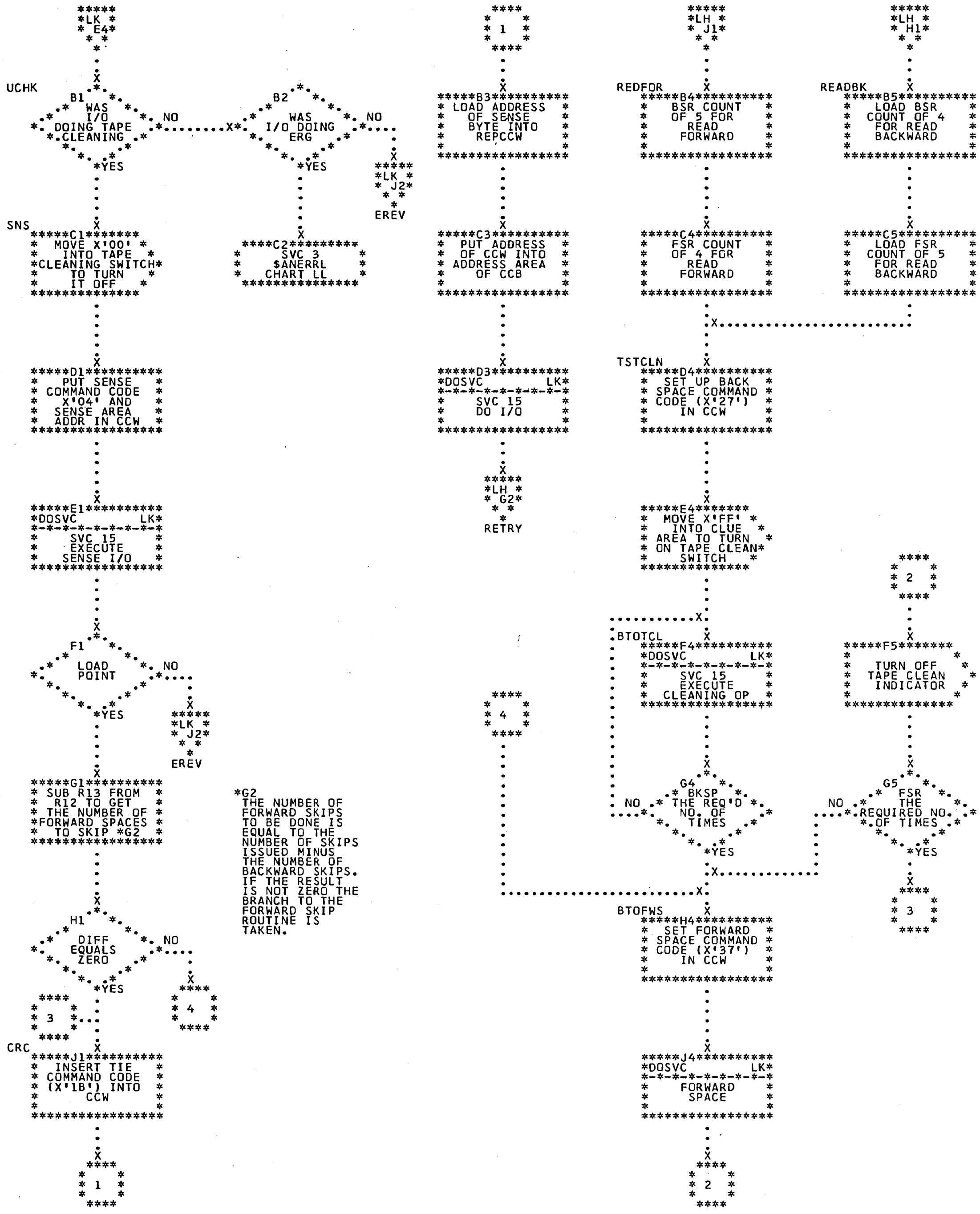

LK E4
UCHK
B1 WAS I/O DOING TAPE CLEANING
NO
YES
B2 WAS I/O DOING ERG
NO
YES
LK J2
EREV
SNS
C1 MOVE X'00' INTO TAPE CLEANING SWITCH TO TURN IT OFF
C2 SVC 3 $ANERRL CHART LL
D1 PUT SENSE COMMAND CODE X'04' AND SENSE AREA ADDR IN CCW
E1 DOSVC LK SVC 15 EXECUTE SENSE I/O
F1 LOAD POINT
NO
YES
LK J2
EREV
G1 SUB R13 FROM R12 TO GET THE NUMBER OF FORWARD SPACES TO SKIP *G2
*G2 THE NUMBER OF FORWARD SKIPS TO BE DONE IS EQUAL TO THE NUMBER OF SKIPS ISSUED MINUS THE NUMBER OF BACKWARD SKIPS. IF THE RESULT IS NOT ZERO THE BRANCH TO THE FORWARD SKIP ROUTINE IS TAKEN.
H1 DIFF EQUALS ZERO
NO
YES
3
4
CRC
J1 INSERT TIE COMMAND CODE (X'1B') INTO CCW
1
1
B3 LOAD ADDRESS OF SENSE BYTE INTO REPCCW
C3 PUT ADDRESS OF CCW INTO ADDRESS AREA OF CCB
D3 DOSVC LK SVC 15 DO I/O
LH G2
RETRY
4
LH J1
REDFOR
B4 BSR COUNT OF 5 FOR READ FORWARD
C4 FSR COUNT OF 4 FOR READ FORWARD
TSTCLN
D4 SET UP BACK SPACE COMMAND CODE (X'27') IN CCW
E4 MOVE X'FF' INTO CLUE AREA TO TURN ON TAPE CLEAN SWITCH
BTOTCL
F4 DOSVC LK SVC 15 EXECUTE CLEANING OP
G4 BKSP THE REQ'D NO. OF TIMES
NO
YES
BTOFWS
H4 SET FORWARD SPACE COMMAND CODE (X'37') IN CCW
J4 DOSVC LK FORWARD SPACE
2
LH H1
READBK
B5 LOAD BSR COUNT OF 4 FOR READ BACKWARD
C5 LOAD FSR COUNT OF 5 FOR READ BACKWARD
2
F5 TURN OFF TAPE CLEAN INDICATOR
G5 FSR THE REQUIRED NO. OF TIMES
NO
YES
3

Chart LK. $$ANERRF - Phase 3 of Tape ERP (Part 3 of 3)
Refer to Chart 07.

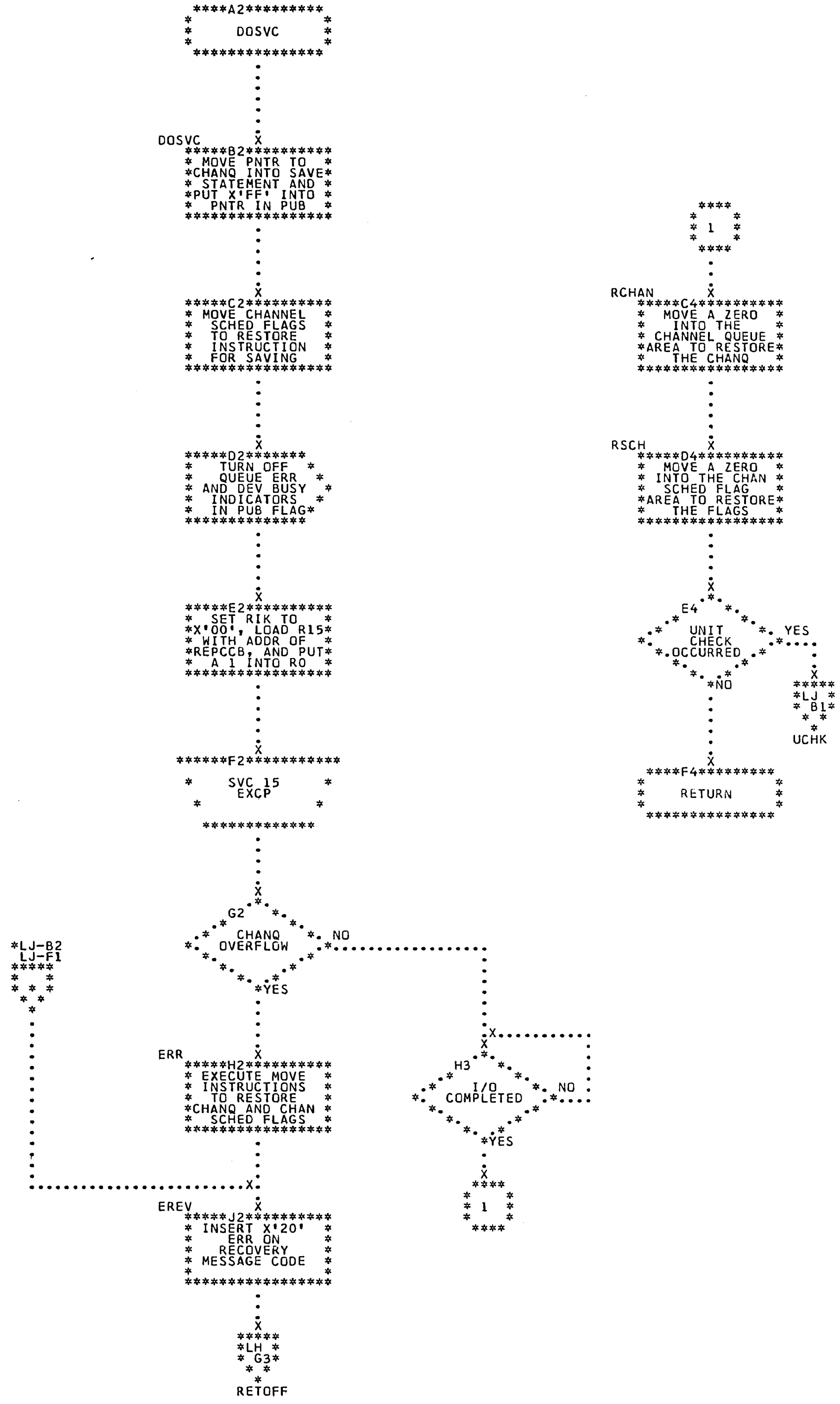
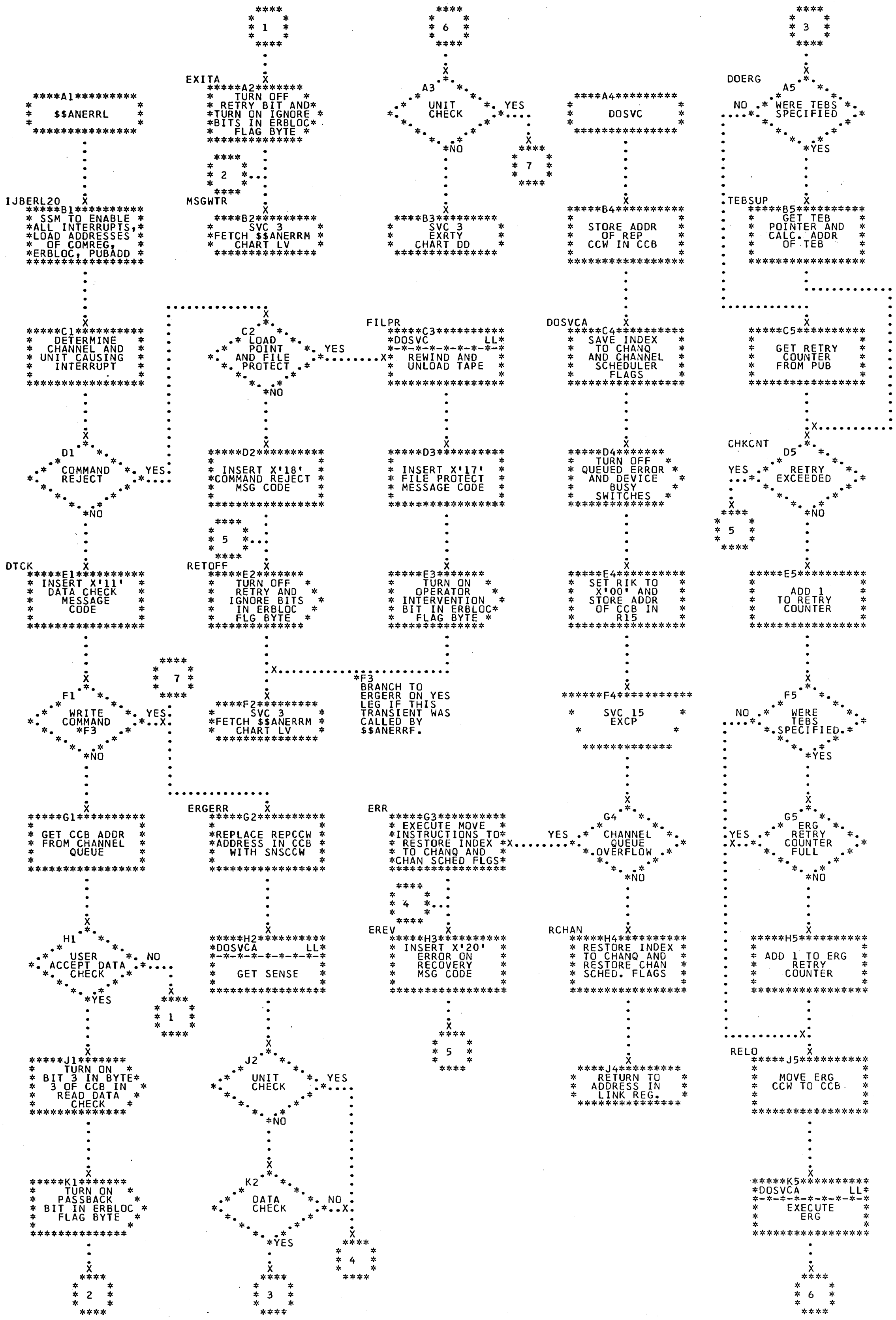

A2
DOSVC
DOSVC
B2
MOVE PNTR TO CHANQ INTO SAVE STATEMENT AND PUT X'FF' INTO PNTR IN PUB
C2
MOVE CHANNEL SCHED FLAGS TO RESTORE INSTRUCTION FOR SAVING
D2
TURN OFF QUEUE ERR AND DEV BUSY INDICATORS IN PUB FLAG
E2
SET RIK TO X'00', LOAD R15 WITH ADDR OF REPCCB, AND PUT A 1 INTO RO
F2
SVC 15 EXCP
G2
CHANQ OVERFLOW
NO
YES
LJ-B2
LJ-F1
ERR
H2
EXECUTE MOVE INSTRUCTIONS TO RESTORE CHANQ AND CHAN SCHED FLAGS
EREV
J2
INSERT X'20' ERR ON RECOVERY MESSAGE CODE
LH
G3
RETOFF
H3
I/O COMPLETED
NO
YES
1
1
RCHAN
C4
MOVE A ZERO INTO THE CHANNEL QUEUE AREA TO RESTORE THE CHANQ
RSCH
D4
MOVE A ZERO INTO THE CHAN SCHED FLAG AREA TO RESTORE THE FLAGS
E4
UNIT CHECK OCCURRED
YES
NO
LJ
B1
UCHK
F4
RETURN

Chart LL. $$ANERRL - Phase 4 of Tape ERP
Refer to Chart 07.



A1
$$ANERRL
IJBERL20
B1
SSM TO ENABLE ALL INTERRUPTS, LOAD ADDRESSES OF COMREG, ERBLOC, PUBADD
C1
DETERMINE CHANNEL AND UNIT CAUSING INTERRUPT
D1
COMMAND REJECT
YES
NO
DTCK
E1
INSERT X'11' DATA CHECK MESSAGE CODE
F1
WRITE COMMAND *F3
YES
NO
G1
GET CCB ADDR FROM CHANNEL QUEUE
H1
USER ACCEPT DATA CHECK
NO
YES
1
J1
TURN ON BIT 3 IN BYTE 3 OF CCB IN READ DATA CHECK
K1
TURN ON PASSBACK BIT IN ERBLOC FLAG BYTE
2
1
EXITA
A2
TURN OFF RETRY BIT AND TURN ON IGNORE BITS IN ERBLOC FLAG BYTE
2
MSGWTR
B2
SVC 3 FETCH $$ANERRM CHART LV
C2
LOAD POINT AND FILE PROTECT
YES
NO
D2
INSERT X'18' COMMAND REJECT MSG CODE
5
RETOFF
E2
TURN OFF RETRY AND IGNORE BITS IN ERBLOC FLG BYTE
7
F2
SVC 3 FETCH $$ANERRM CHART LV
ERGERR
G2
REPLACE REPCCW ADDRESS IN CCB WITH SNSCCW
H2
DOSVCA LL
GET SENSE
J2
UNIT CHECK
YES
NO
K2
DATA CHECK
NO
YES
4
3
6
A3
UNIT CHECK
YES
NO
7
B3
SVC 3 EXRTY CHART DD
FILPR
C3
DOSVC LL
REWIND AND UNLOAD TAPE
D3
INSERT X'17' FILE PROTECT MESSAGE CODE
E3
TURN ON OPERATOR INTERVENTION BIT IN ERBLOC FLAG BYTE
F3
BRANCH TO ERGERR ON YES LEG IF THIS TRANSIENT WAS CALLED BY $$ANERRF.
ERR
G3
EXECUTE MOVE INSTRUCTIONS TO RESTORE INDEX TO CHANQ AND CHAN SCHED FLGS
4
EREV
H3
INSERT X'20' ERROR ON RECOVERY MSG CODE
5
A4
DOSVC
B4
STORE ADDR OF REP CCW IN CCB
DOSVCA
C4
SAVE INDEX TO CHANQ AND CHANNEL SCHEDULER FLAGS
D4
TURN OFF QUEUED ERROR AND DEVICE BUSY SWITCHES
E4
SET RIK TO X'00' AND STORE ADDR OF CCB IN R15
F4
SVC 15 EXCP
G4
CHANNEL QUEUE OVERFLOW
YES
NO
RCHAN
H4
RESTORE INDEX TO CHANQ AND RESTORE CHAN SCHED. FLAGS
J4
RETURN TO ADDRESS IN LINK REG.
3
DOERG
A5
WERE TEBS SPECIFIED
NO
YES
TEBSUP
B5
GET TEB POINTER AND CALC. ADDR OF TEB
C5
GET RETRY COUNTER FROM PUB
CHKCNT
D5
RETRY EXCEEDED
YES
NO
5
E5
ADD 1 TO RETRY COUNTER
F5
WERE TEBS SPECIFIED
NO
YES
G5
ERG RETRY COUNTER FULL
YES
NO
H5
ADD 1 TO ERG RETRY COUNTER
RELO
J5
MOVE ERG CCW TO CCB
K5
DOSVCA LL
EXECUTE ERG
6

Chart LM. $$ANERRG - Phase 1 of Data Cell ERP (Part 1 of 3)
Refer to Chart 07.

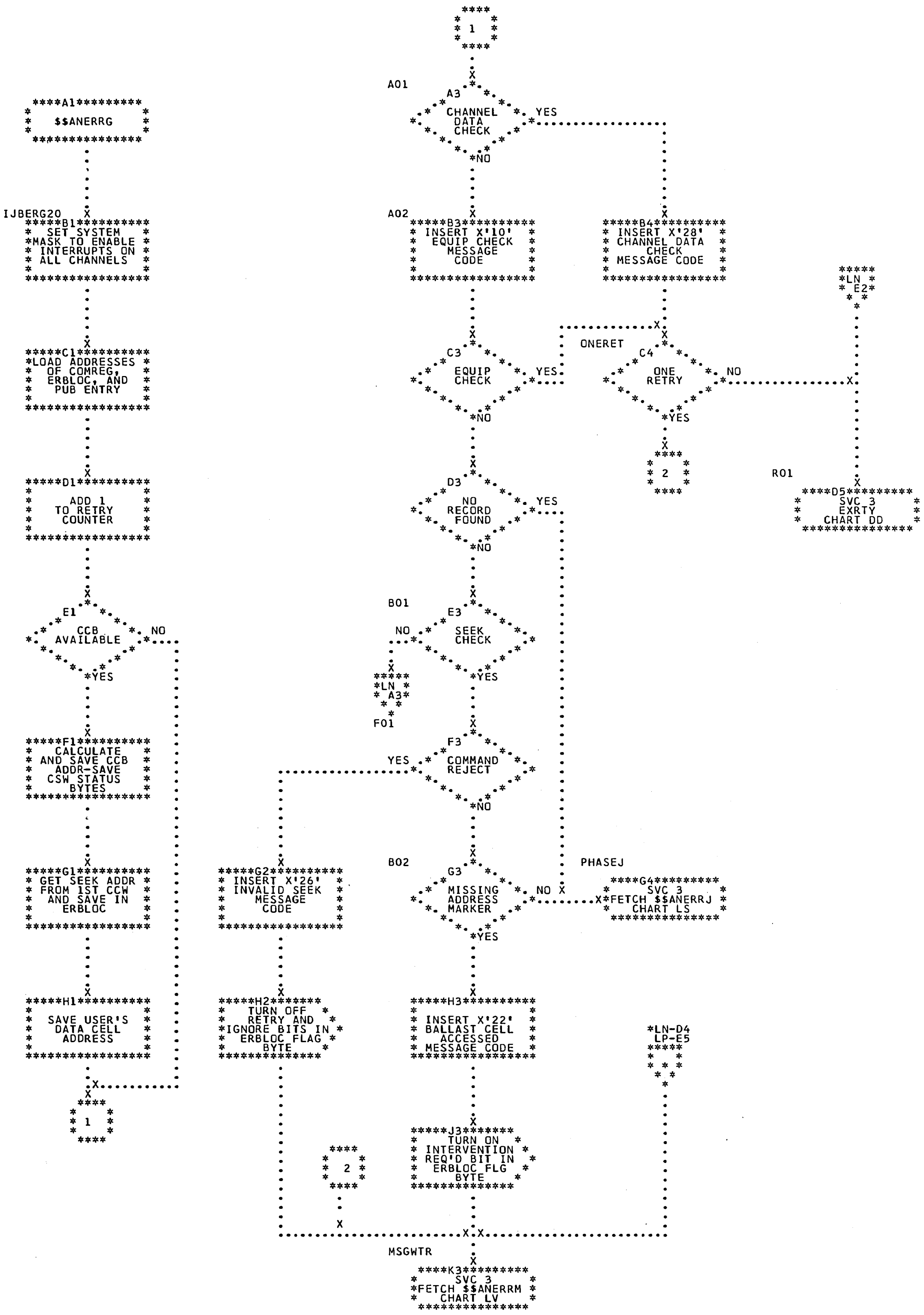

1
A1
$$ANERRG
IJBERG20
B1
SET SYSTEM MASK TO ENABLE INTERRUPTS ON ALL CHANNELS
C1
LOAD ADDRESSES OF COMREG, ERBLOC, AND PUB ENTRY
D1
ADD 1 TO RETRY COUNTER
E1
CCB AVAILABLE
NO
YES
F1
CALCULATE AND SAVE CCB ADDR-SAVE CSW STATUS BYTES
G1
GET SEEK ADDR FROM 1ST CCW AND SAVE IN ERBLOC
H1
SAVE USER'S DATA CELL ADDRESS
1
A01
A3
CHANNEL DATA CHECK
YES
NO
A02
B3
INSERT X'10' EQUIP CHECK MESSAGE CODE
B4
INSERT X'28' CHANNEL DATA CHECK MESSAGE CODE
C3
EQUIP CHECK
YES
NO
ONERET
C4
ONE RETRY
NO
YES
2
LN E2
R01
D5
SVC 3 EXRTY CHART DD
D3
NO RECORD FOUND
YES
NO
B01
E3
SEEK CHECK
NO
YES
LN A3
F01
F3
COMMAND REJECT
YES
NO
G2
INSERT X'26' INVALID SEEK MESSAGE CODE
B02
G3
MISSING ADDRESS MARKER
NO
YES
PHASEJ
G4
SVC 3 FETCH $$ANERRJ CHART LS
H2
TURN OFF RETRY AND IGNORE BITS IN ERBLOC FLAG BYTE
H3
INSERT X'22' BALLAST CELL ACCESSED MESSAGE CODE
LN-D4
LP-E5
2
J3
TURN ON INTERVENTION REQ'D BIT IN ERBLOC FLG BYTE
MSGWTR
K3
SVC 3 FETCH $$ANERRM CHART LV

Chart LN. $$ANERRG - Phase 1 of Data Cell ERP (Part 2 of 3)
Refer to Chart 07.

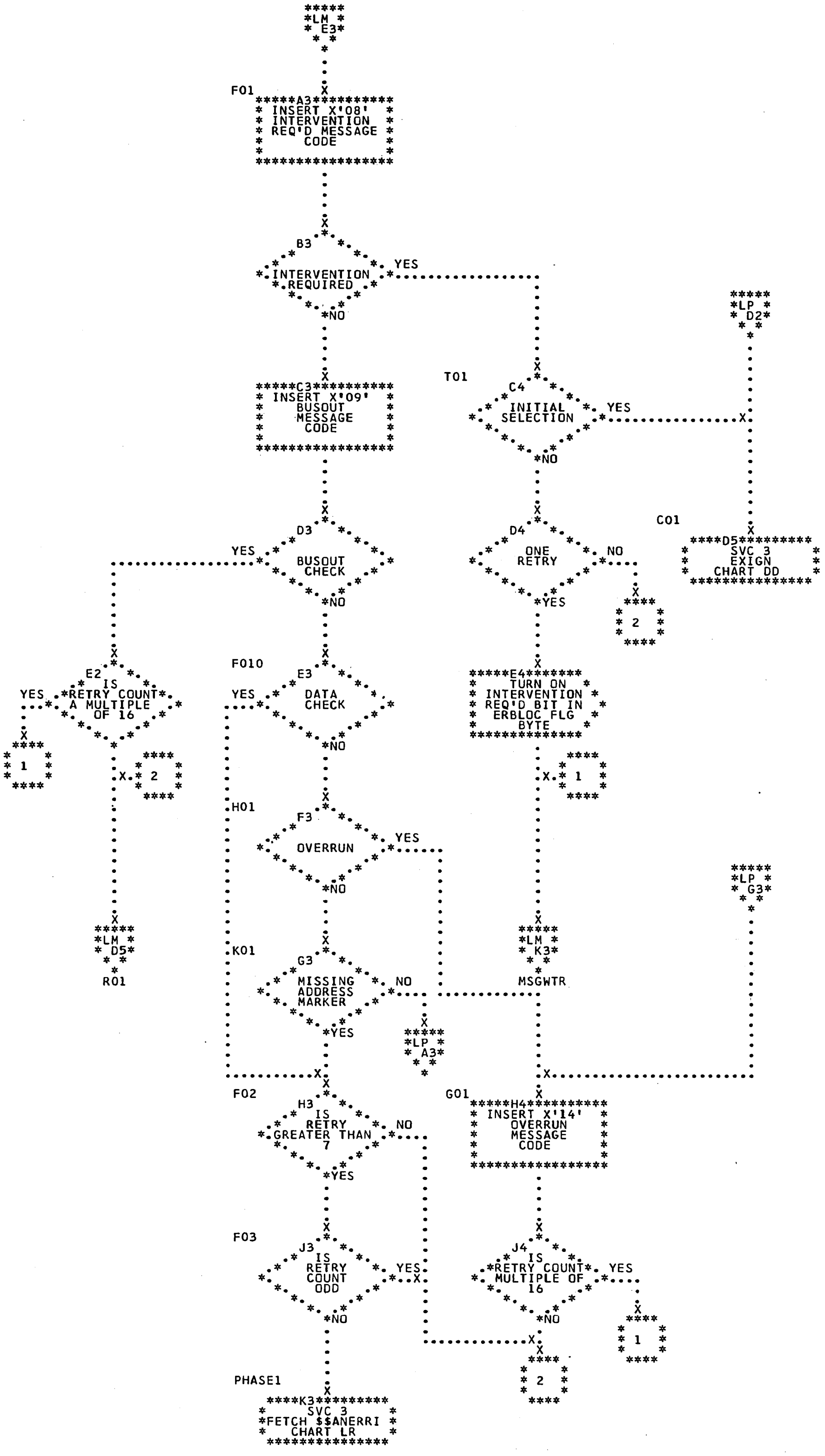
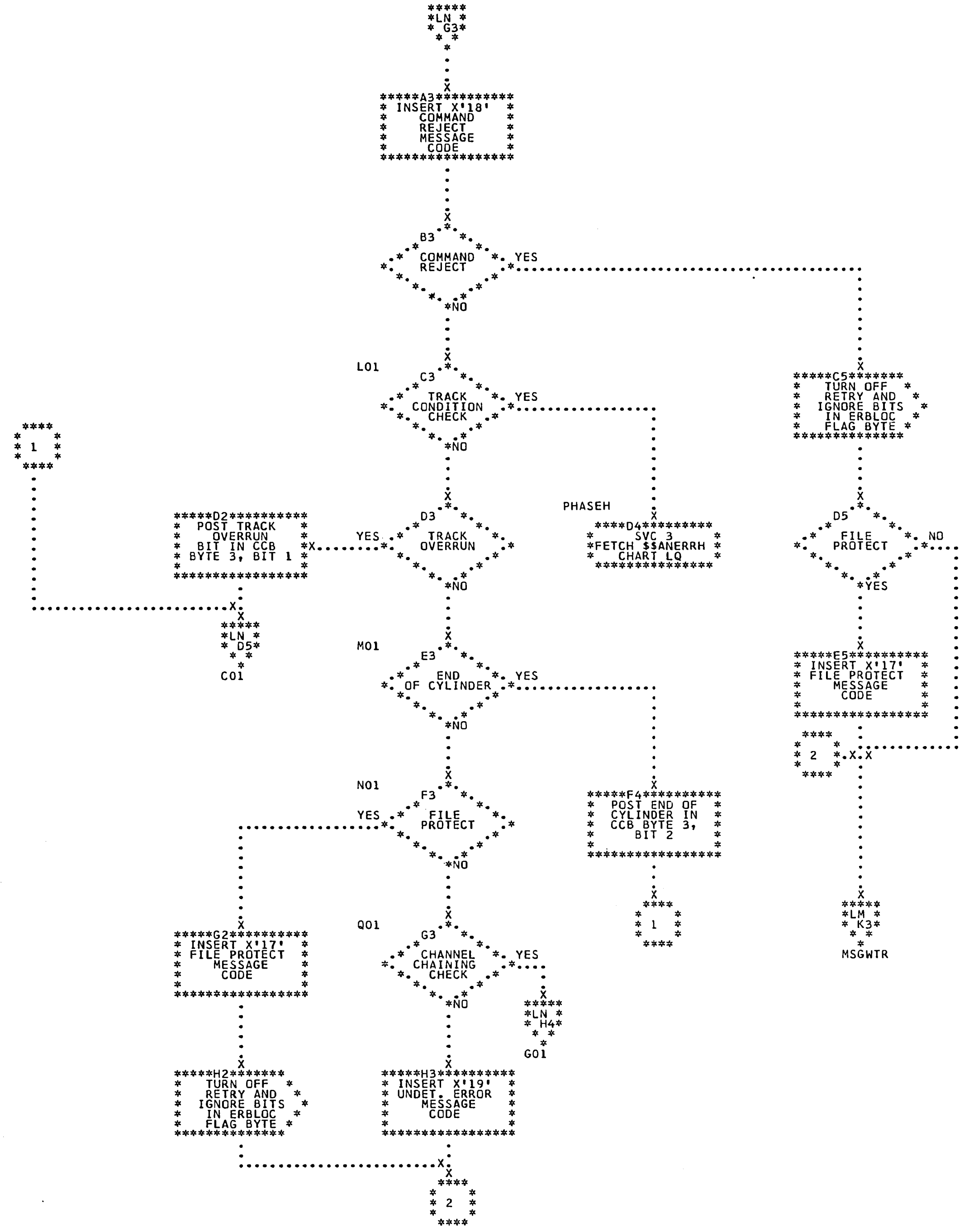

LM E3
F01
A3 INSERT X'08' INTERVENTION REQ'D MESSAGE CODE
B3 INTERVENTION REQUIRED
YES
NO
LP D2
T01
C4 INITIAL SELECTION
YES
NO
C3 INSERT X'09' BUSOUT MESSAGE CODE
C01
D5 SVC 3 EXIGN CHART DD
D3 BUSOUT CHECK
YES
NO
D4 ONE RETRY
NO
YES
2
E2 IS RETRY COUNT A MULTIPLE OF 16
YES
1
2
F010
E3 DATA CHECK
YES
NO
E4 TURN ON INTERVENTION REQ'D BIT IN ERBLOC FLG BYTE
1
H01
F3 OVERRUN
YES
NO
LP G3
LM D5
R01
K01
G3 MISSING ADDRESS MARKER
NO
YES
LP A3
LM K3
MSGWTR
F02
H3 IS RETRY GREATER THAN 7
NO
YES
G01
H4 INSERT X'14' OVERRUN MESSAGE CODE
F03
J3 IS RETRY COUNT ODD
YES
NO
J4 IS RETRY COUNT MULTIPLE OF 16
YES
NO
1
2
PHASE1
K3 SVC 3 FETCH $$ANERRI CHART LR

Chart LP. $$ANERRG - Phase 1 of Data Cell ERP (Part 3 of 3)
Refer to Chart 07.



LN G3
A3
INSERT X'18' COMMAND REJECT MESSAGE CODE
B3
COMMAND REJECT
YES
NO
L01
C3
TRACK CONDITION CHECK
YES
NO
C5
TURN OFF RETRY AND IGNORE BITS IN ERBLOC FLAG BYTE
1
D2
POST TRACK OVERRUN BIT IN CCB BYTE 3, BIT 1
YES
D3
TRACK OVERRUN
NO
PHASEH
D4
SVC 3 FETCH $$ANERRH CHART LQ
D5
FILE PROTECT
NO
YES
LN D5
C01
M01
E3
END OF CYLINDER
YES
NO
E5
INSERT X'17' FILE PROTECT MESSAGE CODE
2
N01
F3
FILE PROTECT
YES
NO
F4
POST END OF CYLINDER IN CCB BYTE 3, BIT 2
1
LM K3
MSGWTR
Q01
G2
INSERT X'17' FILE PROTECT MESSAGE CODE
G3
CHANNEL CHAINING CHECK
YES
NO
LN H4
G01
H2
TURN OFF RETRY AND IGNORE BITS IN ERBLOC FLAG BYTE
H3
INSERT X'19' UNDET. ERROR MESSAGE CODE
2

Chart LQ. $$ANERRH - Phase 2 of Data Cell ERP
Refer to Chart 07.

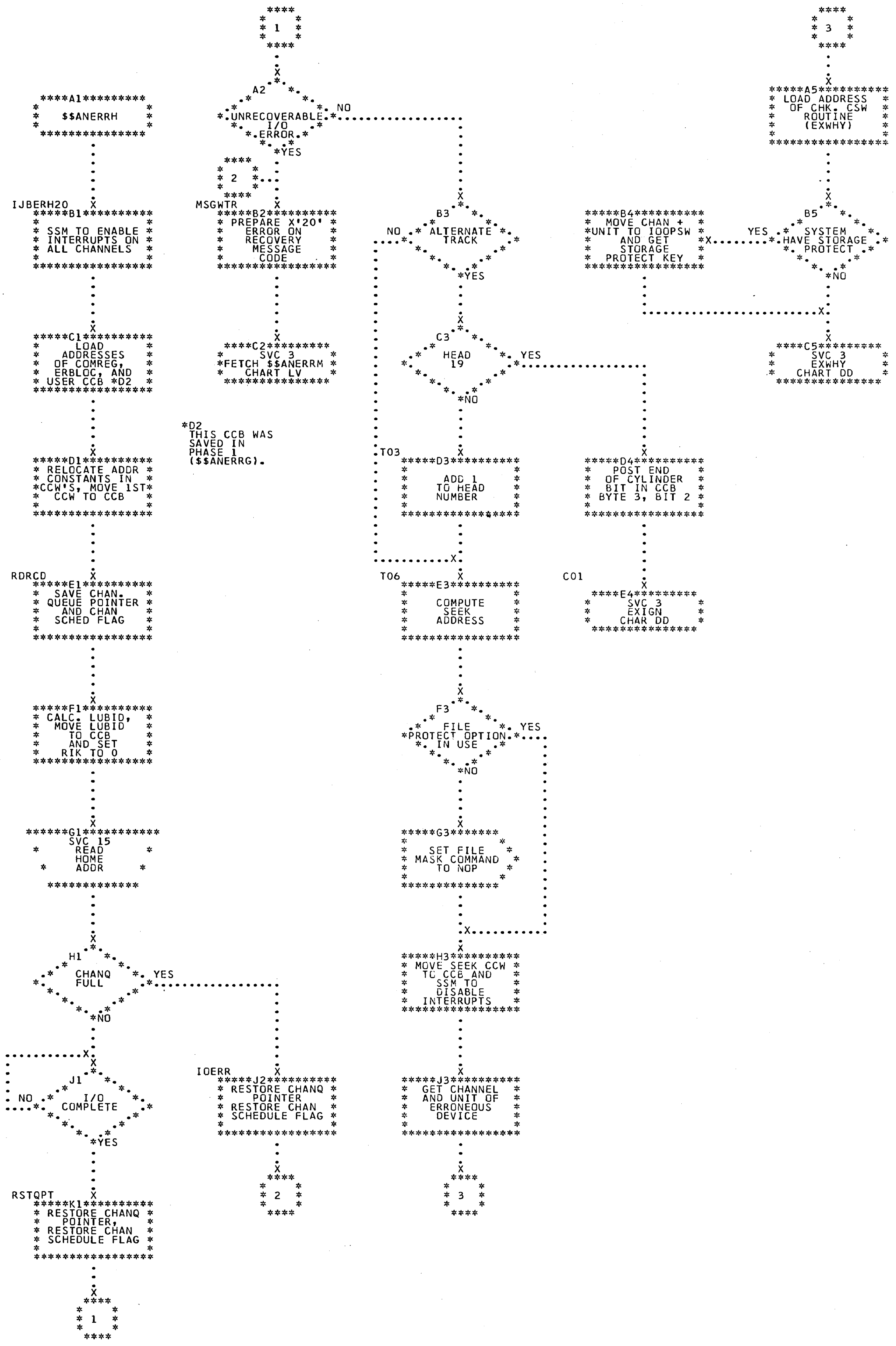

1
A1 $$ANERRH
IJBERH20
B1 SSM TO ENABLE INTERRUPTS ON ALL CHANNELS
C1 LOAD ADDRESSES OF COMREG, ERBLOC, AND USER CCB *D2
D1 RELOCATE ADDR CONSTANTS IN CCW'S, MOVE 1ST CCW TO CCB
RDRCD
E1 SAVE CHAN. QUEUE POINTER AND CHAN SCHED FLAG
F1 CALC. LUBID, MOVE LUBID TO CCB AND SET RIK TO 0
G1 SVC 15 READ HOME ADDR
H1 CHANQ FULL
YES
NO
J1 I/O COMPLETE
NO
YES
RSTQPT
K1 RESTORE CHANQ POINTER, RESTORE CHAN SCHEDULE FLAG
1
A2 UNRECOVERABLE I/O ERROR
NO
YES
2
MSGWTR
B2 PREPARE X'20' ERROR ON RECOVERY MESSAGE CODE
C2 SVC 3 FETCH $$ANERRM CHART LV
*D2 THIS CCB WAS SAVED IN PHASE 1 ($$ANERRG).
IOERR
J2 RESTORE CHANQ POINTER RESTORE CHAN SCHEDULE FLAG
2
B3 ALTERNATE TRACK
NO
YES
C3 HEAD 19
YES
NO
T03
D3 ADD 1 TO HEAD NUMBER
T06
E3 COMPUTE SEEK ADDRESS
F3 FILE PROTECT OPTION IN USE
YES
NO
G3 SET FILE MASK COMMAND TO NOP
H3 MOVE SEEK CCW TO CCB AND SSM TO DISABLE INTERRUPTS
J3 GET CHANNEL AND UNIT OF ERRONEOUS DEVICE
3
B4 MOVE CHAN + UNIT TO IOOPSW AND GET STORAGE PROTECT KEY
D4 POST END OF CYLINDER BIT IN CCB BYTE 3, BIT 2
C01
E4 SVC 3 EXIGN CHAR DD
3
A5 LOAD ADDRESS OF CHK. CSW ROUTINE (EXWHY)
B5 SYSTEM HAVE STORAGE PROTECT
YES
NO
C5 SVC 3 EXWHY CHART DD

Chart LR. $$ANERRI - Phase 3 of Data Cell ERP
Refer to Chart 07.

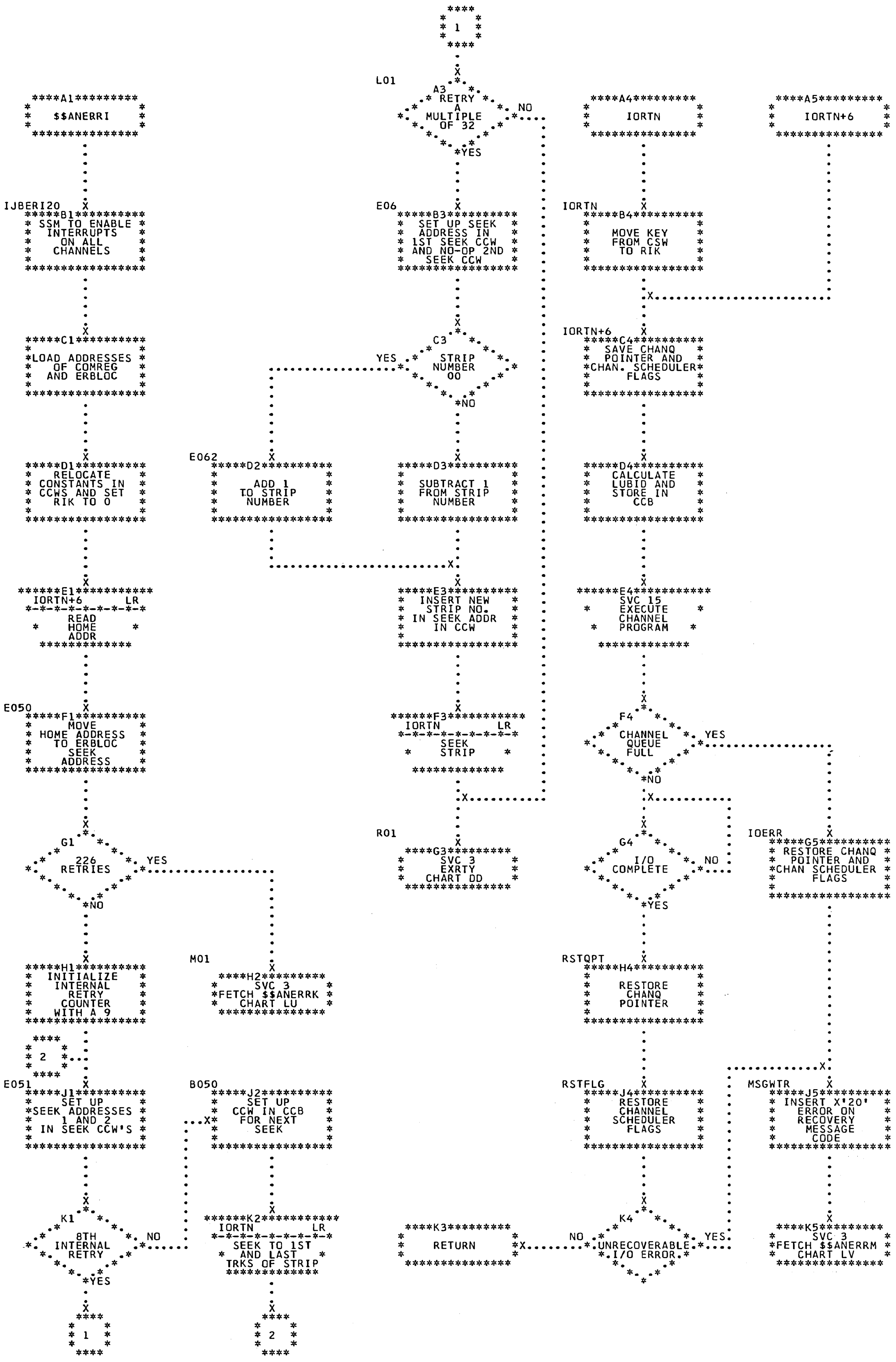

1
A1 $$ANERRI
IJBERI20
B1 SSM TO ENABLE INTERRUPTS ON ALL CHANNELS
C1 LOAD ADDRESSES OF COMREG AND ERBLOC
D1 RELOCATE CONSTANTS IN CCWS AND SET RIK TO 0
E1 IORTN+6 LR READ HOME ADDR
E050
F1 MOVE HOME ADDRESS TO ERBLOC SEEK ADDRESS
G1 226 RETRIES
YES
NO
H1 INITIALIZE INTERNAL RETRY COUNTER WITH A 9
2
E051
J1 SET UP SEEK ADDRESSES 1 AND 2 IN SEEK CCW'S
K1 8TH INTERNAL RETRY
NO
YES
1
L01
A3 RETRY A MULTIPLE OF 32
NO
YES
E06
B3 SET UP SEEK ADDRESS IN 1ST SEEK CCW AND NO-OP 2ND SEEK CCW
C3 STRIP NUMBER 00
YES
NO
E062
D2 ADD 1 TO STRIP NUMBER
D3 SUBTRACT 1 FROM STRIP NUMBER
E3 INSERT NEW STRIP NO. IN SEEK ADDR IN CCW
F3 IORTN LR SEEK STRIP
R01
G3 SVC 3 EXRTY CHART DD
M01
H2 SVC 3 FETCH $$ANERRK CHART LU
B050
J2 SET UP CCW IN CCB FOR NEXT SEEK
K2 IORTN LR SEEK TO 1ST AND LAST TRKS OF STRIP
2
A4 IORTN
IORTN
B4 MOVE KEY FROM CSW TO RIK
IORTN+6
C4 SAVE CHANQ POINTER AND CHAN. SCHEDULER FLAGS
D4 CALCULATE LUBID AND STORE IN CCB
E4 SVC 15 EXECUTE CHANNEL PROGRAM
F4 CHANNEL QUEUE FULL
YES
NO
G4 I/O COMPLETE
NO
YES
RSTQPT
H4 RESTORE CHANQ POINTER
RSTFLG
J4 RESTORE CHANNEL SCHEDULER FLAGS
K4 UNRECOVERABLE I/O ERROR
NO
YES
K3 RETURN
A5 IORTN+6
IOERR
G5 RESTORE CHANQ POINTER AND CHAN SCHEDULER FLAGS
MSGWTR
J5 INSERT X'20' ERROR ON RECOVERY MESSAGE CODE
K5 SVC 3 FETCH $$ANERRM CHART LV

Chart LS. $$ANERRJ - Phase 4 of Data Cell ERP (Part 1 of 2)
Refer to Chart 07.

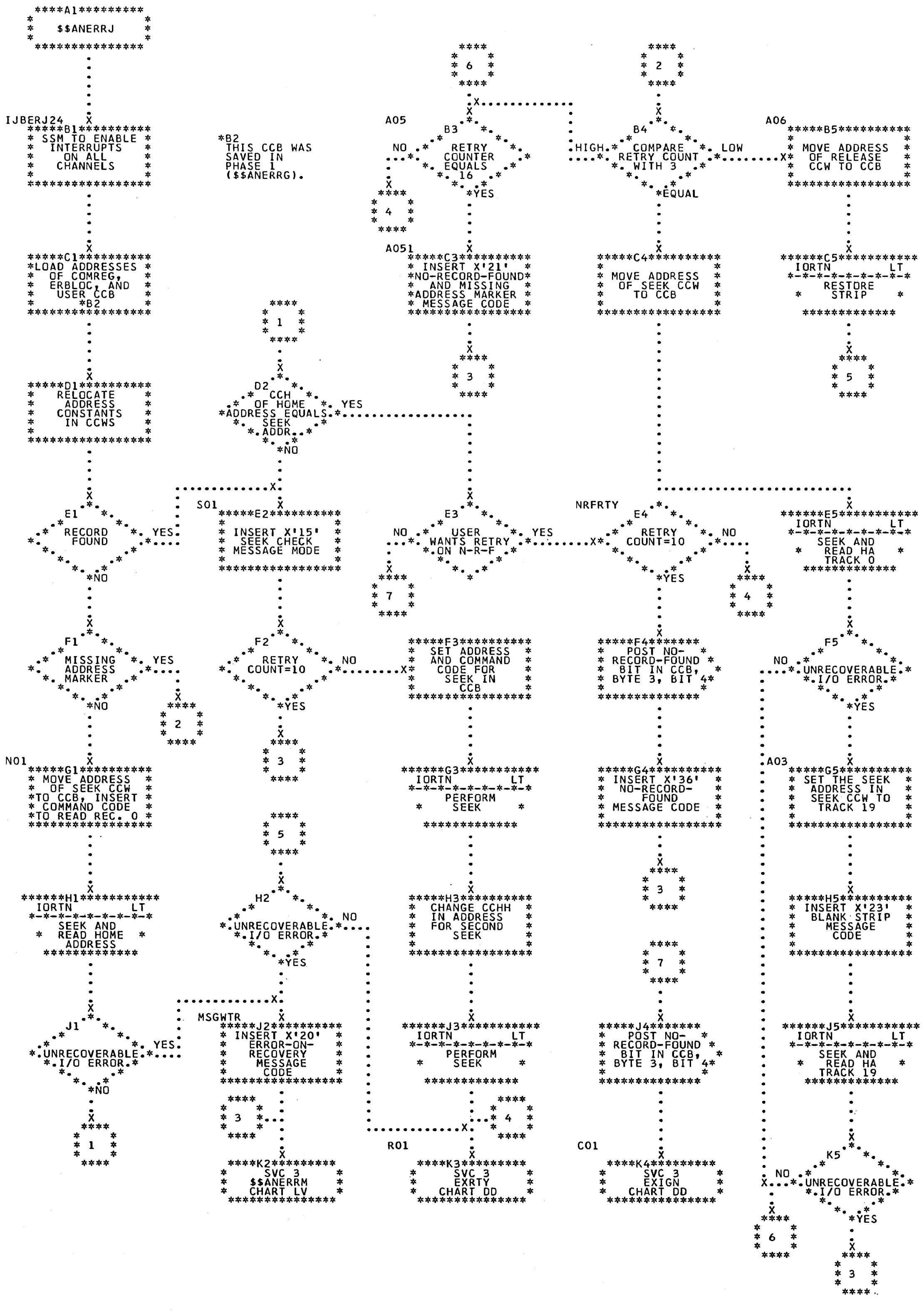

A1
$$ANERRJ
IJBERJ24
B1
SSM TO ENABLE INTERRUPTS ON ALL CHANNELS
C1
LOAD ADDRESSES OF COMREG, ERBLOC, AND USER CCB
*B2
D1
RELOCATE ADDRESS CONSTANTS IN CCWS
E1
RECORD FOUND
YES
NO
F1
MISSING ADDRESS MARKER
YES
2
NO
N01
G1
MOVE ADDRESS OF SEEK CCW TO CCB, INSERT COMMAND CODE TO READ REC. 0
H1
IORTN LT
SEEK AND READ HOME ADDRESS
J1
UNRECOVERABLE I/O ERROR
YES
NO
1
*B2
THIS CCB WAS SAVED IN PHASE 1 ($$ANERRG).
1
D2
CCH OF HOME ADDRESS EQUALS SEEK ADDR.
YES
NO
S01
E2
INSERT X'15' SEEK CHECK MESSAGE MODE
F2
RETRY COUNT=10
NO
YES
3
5
H2
UNRECOVERABLE I/O ERROR
NO
YES
MSGWTR
J2
INSERT X'20' ERROR-ON-RECOVERY MESSAGE CODE
3
K2
SVC 3
$$ANERRM
CHART LV
6
A05
B3
RETRY COUNTER EQUALS 16
NO
4
YES
A051
C3
INSERT X'21' NO-RECORD-FOUND AND MISSING ADDRESS MARKER MESSAGE CODE
3
E3
USER WANTS RETRY ON N-R-F
NO
7
YES
F3
SET ADDRESS AND COMMAND CODE FOR SEEK IN CCB
G3
IORTN LT
PERFORM SEEK
H3
CHANGE CCHH IN ADDRESS FOR SECOND SEEK
J3
IORTN LT
PERFORM SEEK
4
R01
K3
SVC 3
EXRTY
CHART DD
2
B4
COMPARE RETRY COUNT WITH 3
HIGH
LOW
EQUAL
C4
MOVE ADDRESS OF SEEK CCW TO CCB
NRFRTY
E4
RETRY COUNT=10
NO
4
YES
F4
POST NO-RECORD-FOUND BIT IN CCB, BYTE 3, BIT 4
G4
INSERT X'36' NO-RECORD-FOUND MESSAGE CODE
3
7
J4
POST NO-RECORD-FOUND BIT IN CCB, BYTE 3, BIT 4
C01
K4
SVC 3
EXIGN
CHART DD
A06
B5
MOVE ADDRESS OF RELEASE CCW TO CCB
C5
IORTN LT
RESTORE STRIP
5
E5
IORTN LT
SEEK AND READ HA TRACK 0
F5
UNRECOVERABLE I/O ERROR
NO
YES
A03
G5
SET THE SEEK ADDRESS IN SEEK CCW TO TRACK 19
H5
INSERT X'23' BLANK STRIP MESSAGE CODE
J5
IORTN LT
SEEK AND READ HA TRACK 19
K5
UNRECOVERABLE I/O ERROR
NO
6
YES
3

Chart LT. $$ANERRJ - Phase 4 of Data Cell ERP (Part 2 of 2)
Refer to Chart 07.

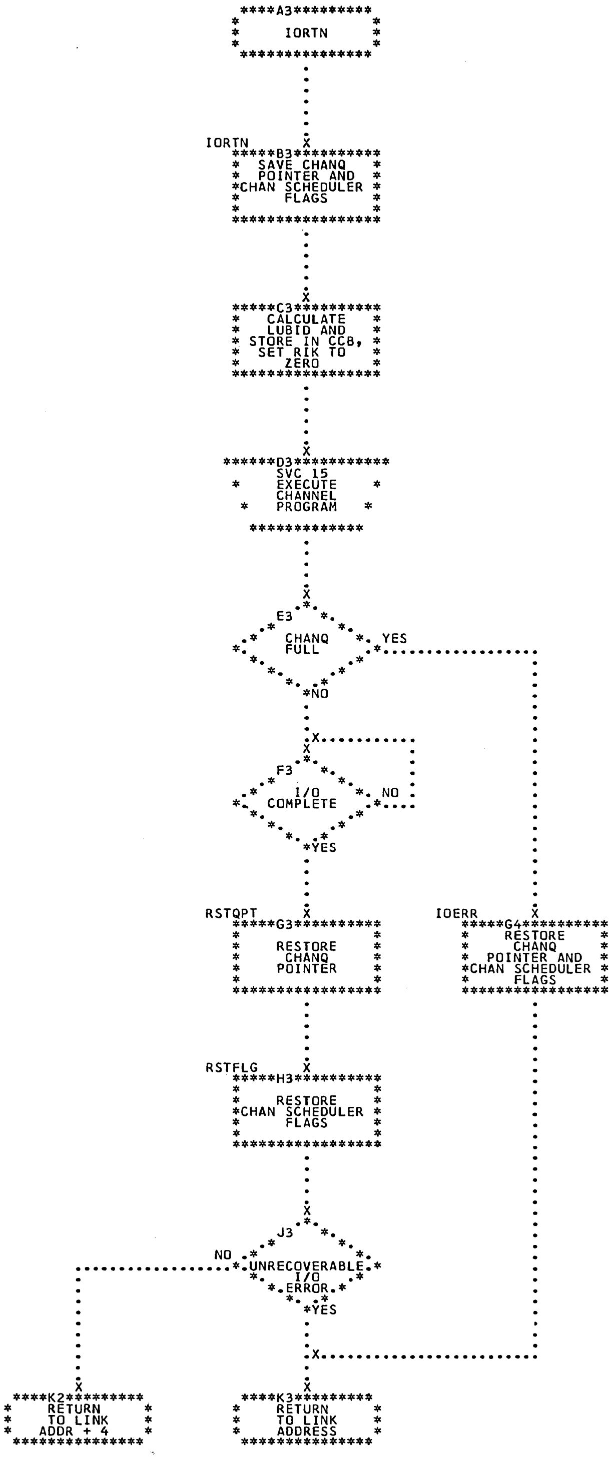

A3
IORTN
IORTN
B3
SAVE CHANQ POINTER AND CHAN SCHEDULER FLAGS
C3
CALCULATE LUBID AND STORE IN CCB, SET RIK TO ZERO
D3
SVC 15 EXECUTE CHANNEL PROGRAM
E3
CHANQ FULL
YES
NO
F3
I/O COMPLETE
NO
YES
RSTQPT
G3
RESTORE CHANQ POINTER
IOERR
G4
RESTORE CHANQ POINTER AND CHAN SCHEDULER FLAGS
RSTFLG
H3
RESTORE CHAN SCHEDULER FLAGS
J3
UNRECOVERABLE I/O ERROR
NO
YES
K2
RETURN TO LINK ADDR + 4
K3
RETURN TO LINK ADDRESS

Chart LU. $$ANERRK - Phase 5 of Data Cell ERP
Refer to Chart 07.

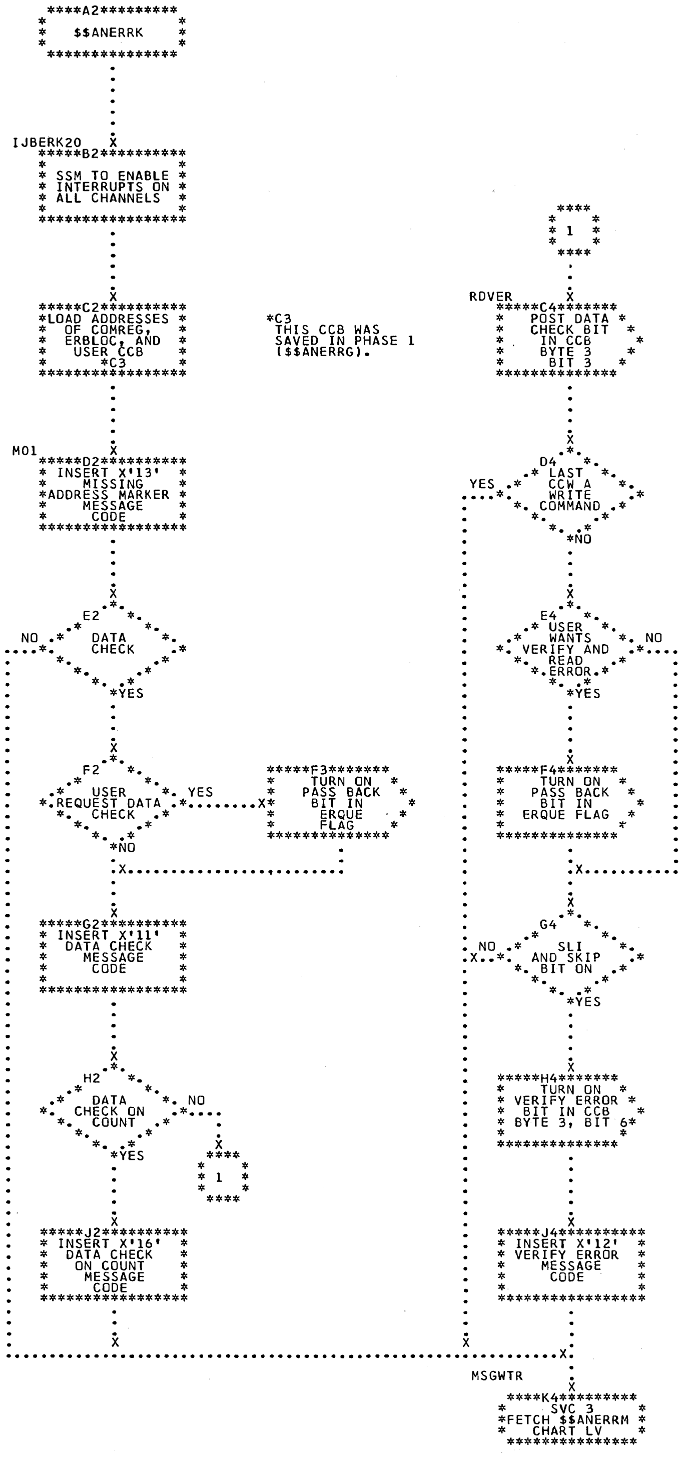

A2
$$ANERRK
IJBERK20
B2
SSM TO ENABLE INTERRUPTS ON ALL CHANNELS
C2
LOAD ADDRESSES OF COMREG, ERBLOC, AND USER CCB *C3
*C3
THIS CCB WAS SAVED IN PHASE 1 ($$ANERRG).
M01
D2
INSERT X'13' MISSING ADDRESS MARKER MESSAGE CODE
E2
DATA CHECK
NO
YES
F2
USER REQUEST DATA CHECK
YES
NO
F3
TURN ON PASS BACK BIT IN ERQUE FLAG
G2
INSERT X'11' DATA CHECK MESSAGE CODE
H2
DATA CHECK ON COUNT
NO
YES
1
J2
INSERT X'16' DATA CHECK ON COUNT MESSAGE CODE
1
RDVER
C4
POST DATA CHECK BIT IN CCB BYTE 3 BIT 3
D4
LAST CCW A WRITE COMMAND
YES
NO
E4
USER WANTS VERIFY AND READ ERROR
NO
YES
F4
TURN ON PASS BACK BIT IN ERQUE FLAG
G4
SLI AND SKIP BIT ON
NO
YES
H4
TURN ON VERIFY ERROR BIT IN CCB BYTE 3, BIT 6
J4
INSERT X'12' VERIFY ERROR MESSAGE CODE
MSGWTR
K4
SVC 3 FETCH $$ANERRM CHART LV

Chart LV. $$ANERRM - Phase 1 of ERP Message Writer
Refer to Chart 08.

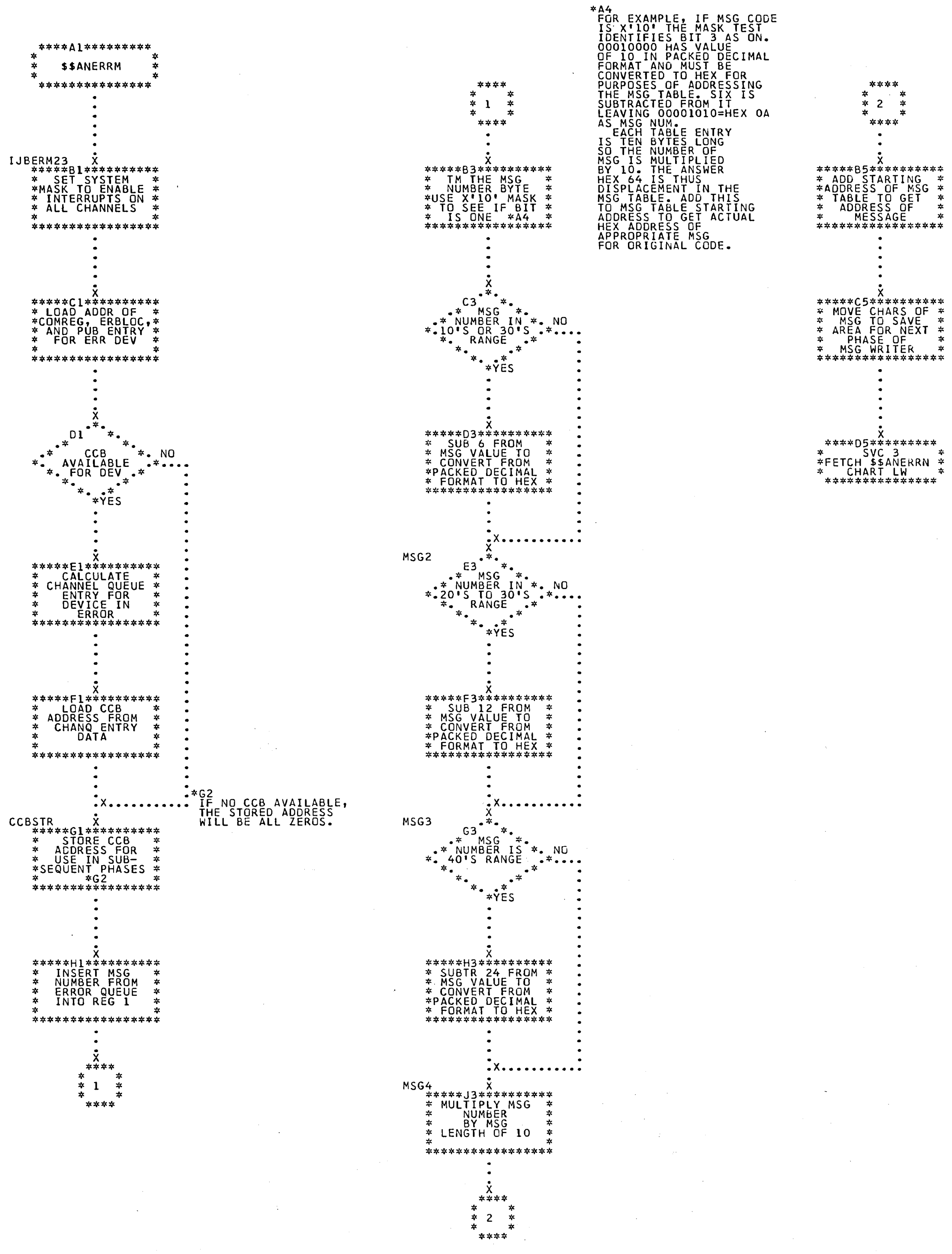
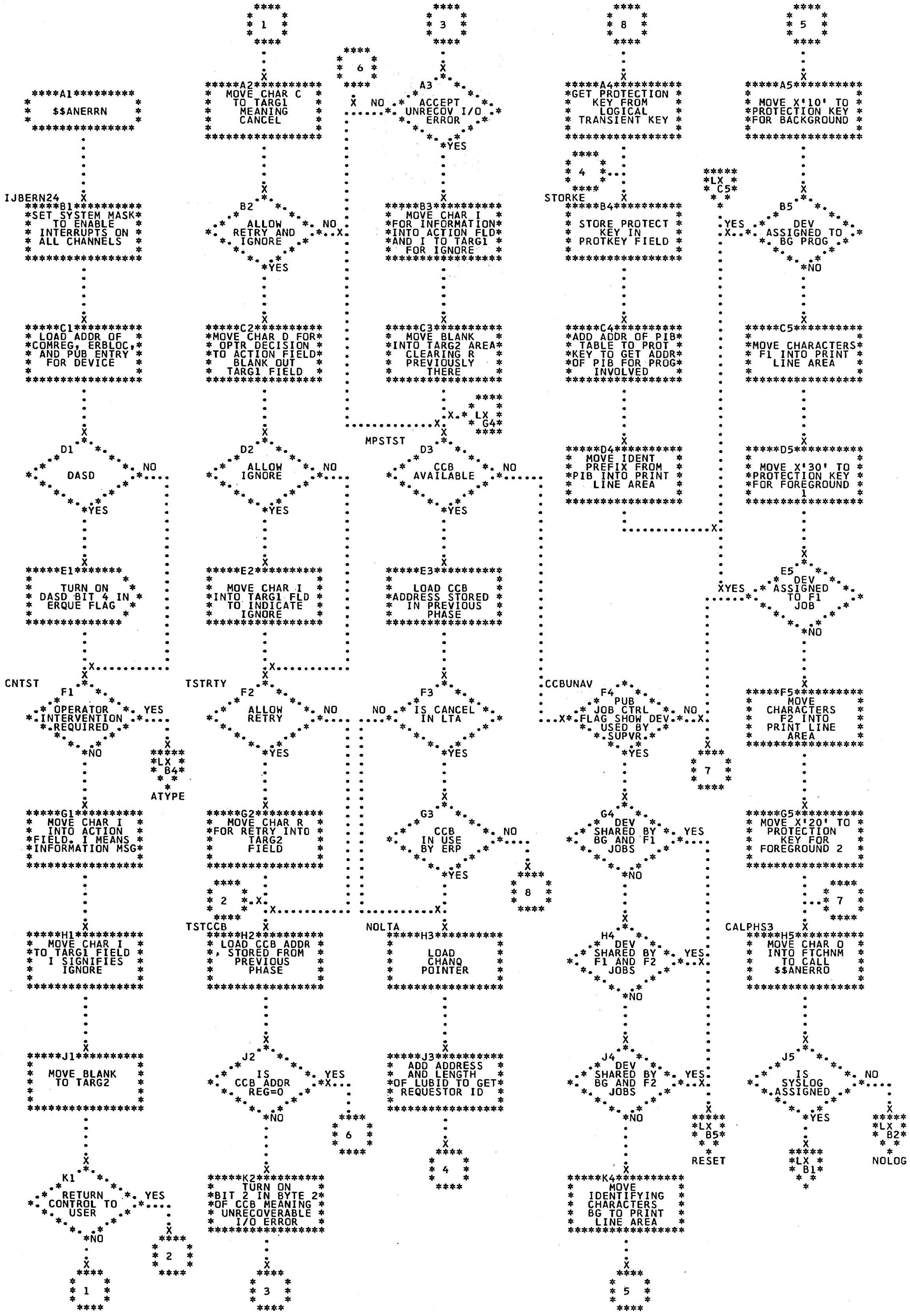

A1
$$ANERRM
IJBERM23
B1
SET SYSTEM MASK TO ENABLE INTERRUPTS ON ALL CHANNELS
C1
LOAD ADDR OF COMREG, ERBLOC, AND PUB ENTRY FOR ERR DEV
D1
CCB AVAILABLE FOR DEV
NO
YES
E1
CALCULATE CHANNEL QUEUE ENTRY FOR DEVICE IN ERROR
F1
LOAD CCB ADDRESS FROM CHANQ ENTRY DATA
*G2
IF NO CCB AVAILABLE, THE STORED ADDRESS WILL BE ALL ZEROS.
CCBSTR
G1
STORE CCB ADDRESS FOR USE IN SUB-SEQUENT PHASES *G2
H1
INSERT MSG NUMBER FROM ERROR QUEUE INTO REG 1
1
1
B3
TM THE MSG NUMBER BYTE USE X'10' MASK TO SEE IF BIT IS ONE *A4
C3
MSG NUMBER IN 10'S OR 30'S RANGE
NO
YES
D3
SUB 6 FROM MSG VALUE TO CONVERT FROM PACKED DECIMAL FORMAT TO HEX
MSG2
E3
MSG NUMBER IN 20'S TO 30'S RANGE
NO
YES
F3
SUB 12 FROM MSG VALUE TO CONVERT FROM PACKED DECIMAL FORMAT TO HEX
MSG3
G3
MSG NUMBER IS 40'S RANGE
NO
YES
H3
SUBTR 24 FROM MSG VALUE TO CONVERT FROM PACKED DECIMAL FORMAT TO HEX
MSG4
J3
MULTIPLY MSG NUMBER BY MSG LENGTH OF 10
2
*A4
FOR EXAMPLE, IF MSG CODE IS X'10' THE MASK TEST IDENTIFIES BIT 3 AS ON. 00010000 HAS VALUE OF 10 IN PACKED DECIMAL FORMAT AND MUST BE CONVERTED TO HEX FOR PURPOSES OF ADDRESSING THE MSG TABLE. SIX IS SUBTRACTED FROM IT LEAVING 00001010=HEX OA AS MSG NUM.
EACH TABLE ENTRY IS TEN BYTES LONG SO THE NUMBER OF MSG IS MULTIPLIED BY 10. THE ANSWER HEX 64 IS THUS DISPLACEMENT IN THE MSG TABLE. ADD THIS TO MSG TABLE STARTING ADDRESS TO GET ACTUAL HEX ADDRESS OF APPROPRIATE MSG FOR ORIGINAL CODE.
2
B5
ADD STARTING ADDRESS OF MSG TABLE TO GET ADDRESS OF MESSAGE
C5
MOVE CHARS OF MSG TO SAVE AREA FOR NEXT PHASE OF MSG WRITER
D5
SVC 3
FETCH $$ANERRN
CHART LW

Chart LW. $$ANERRN - Phase 2 of ERP Message Writer (Part 1 of 2)
Refer to Chart 08.



A1 $$ANERRN
IJBERN24
B1 SET SYSTEM MASK TO ENABLE INTERRUPTS ON ALL CHANNELS
C1 LOAD ADDR OF COMREG, ERBLOC, AND PUB ENTRY FOR DEVICE
D1 DASD
YES
NO
E1 TURN ON DASD BIT 4 IN ERQUE FLAG
CNTST
F1 OPERATOR INTERVENTION REQUIRED
YES
NO
LX B4
ATYPE
G1 MOVE CHAR I INTO ACTION FIELD. I MEANS INFORMATION MSG
H1 MOVE CHAR I TO TARG1 FIELD I SIGNIFIES IGNORE
J1 MOVE BLANK TO TARG2
K1 RETURN CONTROL TO USER
YES
NO
2
1
1
A2 MOVE CHAR C TO TARG1 MEANING CANCEL
B2 ALLOW RETRY AND IGNORE
NO
YES
C2 MOVE CHAR D FOR OPTR DECISION TO ACTION FIELD BLANK OUT TARG1 FIELD
D2 ALLOW IGNORE
NO
YES
E2 MOVE CHAR I INTO TARG1 FLD TO INDICATE IGNORE
TSTRTY
F2 ALLOW RETRY
NO
YES
G2 MOVE CHAR R FOR RETRY INTO TARG2 FIELD
2
TSTCCB
H2 LOAD CCB ADDR STORED FROM PREVIOUS PHASE
J2 IS CCB ADDR REG=0
YES
NO
6
K2 TURN ON BIT 2 IN BYTE 2 OF CCB MEANING UNRECOVERABLE I/O ERROR
3
6
3
A3 ACCEPT UNRECOV I/O ERROR
NO
YES
B3 MOVE CHAR I FOR INFORMATION INTO ACTION FLD AND I TO TARG1 FOR IGNORE
C3 MOVE BLANK INTO TARG2 AREA CLEARING R PREVIOUSLY THERE
LX G4
MPSTST
D3 CCB AVAILABLE
NO
YES
E3 LOAD CCB ADDRESS STORED IN PREVIOUS PHASE
F3 IS CANCEL IN LTA
NO
YES
G3 CCB IN USE BY ERP
NO
YES
8
NOLTA
H3 LOAD CHANQ POINTER
J3 ADD ADDRESS AND LENGTH OF LUBID TO GET REQUESTOR ID
4
8
A4 GET PROTECTION KEY FROM LOGICAL TRANSIENT KEY
4
STORKE
B4 STORE PROTECT KEY IN PROTKEY FIELD
C4 ADD ADDR OF PIB TABLE TO PROT KEY TO GET ADDR OF PIB FOR PROG INVOLVED
D4 MOVE IDENT PREFIX FROM PIB INTO PRINT LINE AREA
CCBUNAV
F4 PUB JOB CTRL FLAG SHOW DEV. USED BY SUPVR.
NO
YES
7
G4 DEV SHARED BY BG AND F1 JOBS
YES
NO
H4 DEV SHARED BY F1 AND F2 JOBS
YES
NO
J4 DEV SHARED BY BG AND F2 JOBS
YES
NO
LX B5
RESET
K4 MOVE IDENTIFYING CHARACTERS BG TO PRINT LINE AREA
5
5
A5 MOVE X'10' TO PROTECTION KEY FOR BACKGROUND
LX C5
B5 DEV ASSIGNED TO BG PROG
YES
NO
C5 MOVE CHARACTERS F1 INTO PRINT LINE AREA
D5 MOVE X'30' TO PROTECTION KEY FOR FOREGROUND 1
E5 DEV ASSIGNED TO F1 JOB
YES
NO
F5 MOVE CHARACTERS F2 INTO PRINT LINE AREA
G5 MOVE X'20' TO PROTECTION KEY FOR FOREGROUND 2
7
CALPHS3
H5 MOVE CHAR O INTO FTCHNM TO CALL $$ANERRO
J5 IS SYSLOG ASSIGNED
NO
YES
LX B1
LX B2
NOLOG

Chart LX. $$ANERRN - Phase 2 of ERP Message Writer (Part 2 of 2)
Refer to Chart 08.

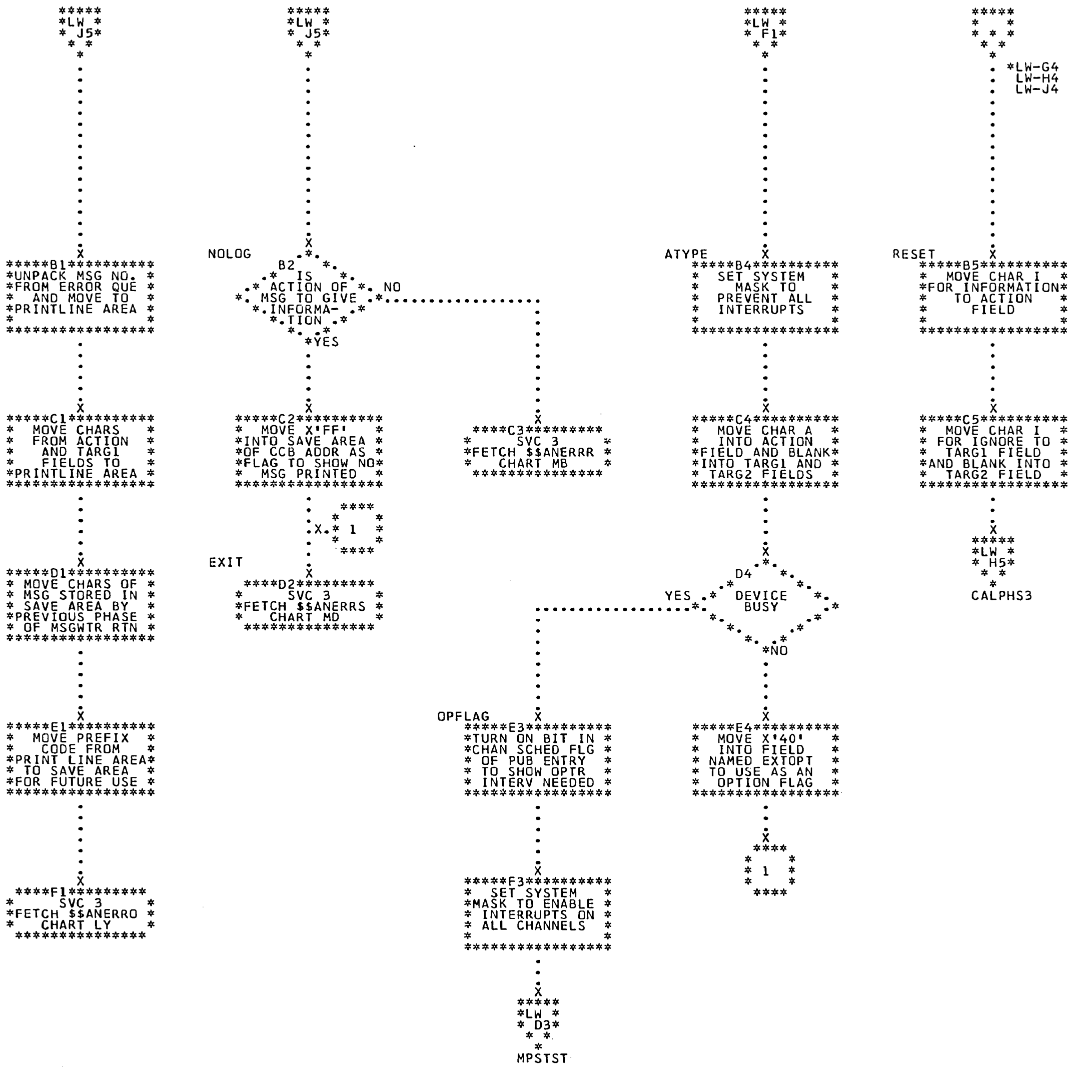

LW J5
LW J5
LW F1
LW-G4
LW-H4
LW-J4
B1
UNPACK MSG NO. FROM ERROR QUE AND MOVE TO PRINTLINE AREA
NOLOG
B2
IS ACTION OF MSG TO GIVE INFORMA-TION
NO
YES
ATYPE
B4
SET SYSTEM MASK TO PREVENT ALL INTERRUPTS
RESET
B5
MOVE CHAR I FOR INFORMATION TO ACTION FIELD
C1
MOVE CHARS FROM ACTION AND TARG1 FIELDS TO PRINTLINE AREA
C2
MOVE X'FF' INTO SAVE AREA OF CCB ADDR AS FLAG TO SHOW NO MSG PRINTED
C3
SVC 3 FETCH $$ANERRR CHART MB
C4
MOVE CHAR A INTO ACTION FIELD AND BLANK INTO TARG1 AND TARG2 FIELDS
C5
MOVE CHAR I FOR IGNORE TO TARG1 FIELD AND BLANK INTO TARG2 FIELD
1
EXIT
LW H5
CALPHS3
D1
MOVE CHARS OF MSG STORED IN SAVE AREA BY PREVIOUS PHASE OF MSGWTR RTN
D2
SVC 3 FETCH $$ANERRS CHART MD
D4
DEVICE BUSY
YES
NO
E1
MOVE PREFIX CODE FROM PRINT LINE AREA TO SAVE AREA FOR FUTURE USE
OPFLAG
E3
TURN ON BIT IN CHAN SCHED FLG OF PUB ENTRY TO SHOW OPTR INTERV NEEDED
E4
MOVE X'40' INTO FIELD NAMED EXTOPT TO USE AS AN OPTION FLAG
1
F1
SVC 3 FETCH $$ANERRO CHART LY
F3
SET SYSTEM MASK TO ENABLE INTERRUPTS ON ALL CHANNELS
LW D3
MPSTST

Chart LY. $$ANERRO - Phase 3 of ERP Message Writer
Refer to Chart 08.

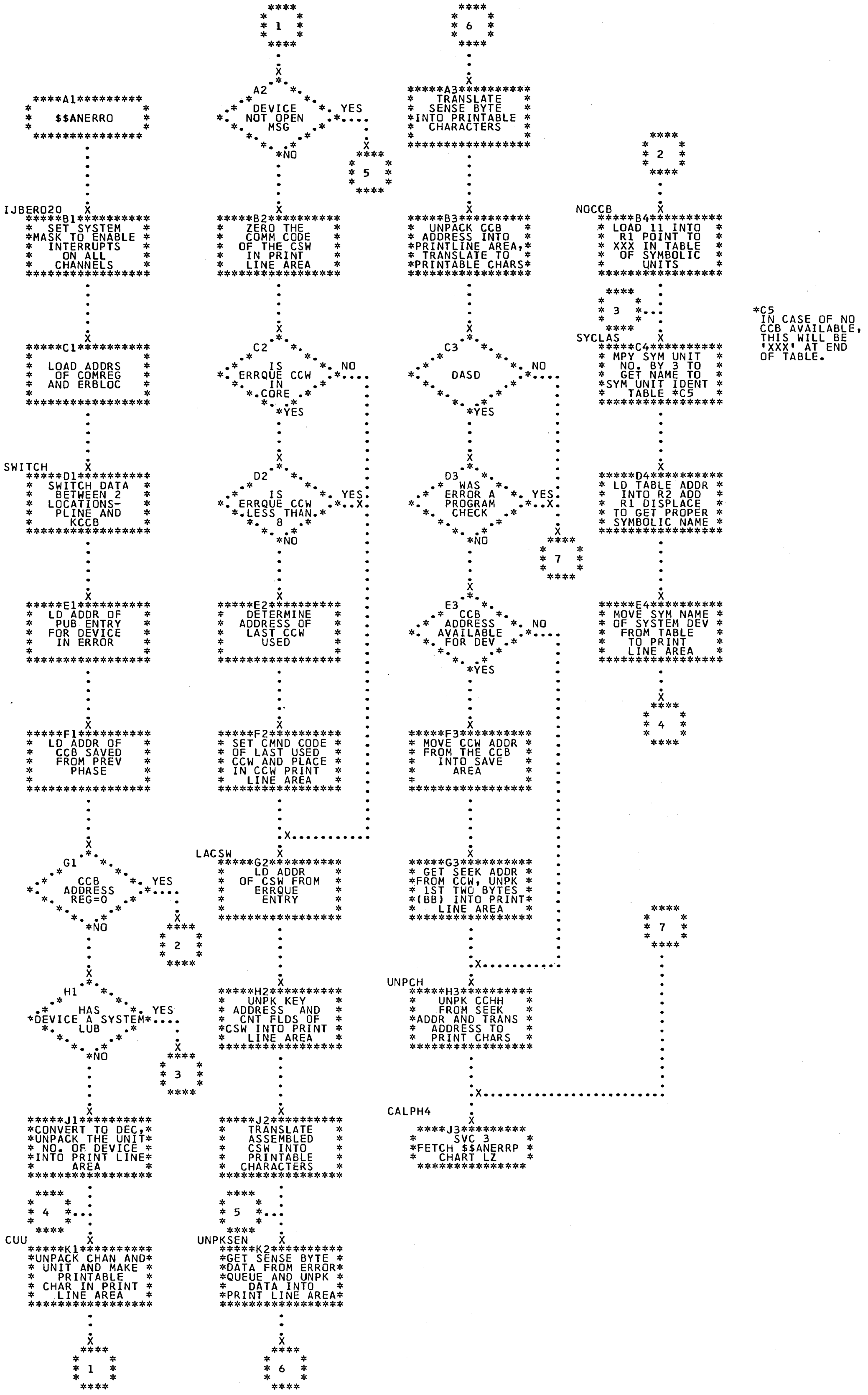

A1 $$ANERRO
IJBERO20
B1 SET SYSTEM MASK TO ENABLE INTERRUPTS ON ALL CHANNELS
C1 LOAD ADDRS OF COMREG AND ERBLOC
SWITCH
D1 SWITCH DATA BETWEEN 2 LOCATIONS- PLINE AND KCCB
E1 LD ADDR OF PUB ENTRY FOR DEVICE IN ERROR
F1 LD ADDR OF CCB SAVED FROM PREV PHASE
G1 CCB ADDRESS REG=0
YES
NO
2
H1 HAS DEVICE A SYSTEM LUB
YES
NO
3
J1 CONVERT TO DEC, UNPACK THE UNIT NO. OF DEVICE INTO PRINT LINE AREA
4
CUU
K1 UNPACK CHAN AND UNIT AND MAKE PRINTABLE CHAR IN PRINT LINE AREA
1
1
A2 DEVICE NOT OPEN MSG
YES
NO
5
B2 ZERO THE COMM CODE OF THE CSW IN PRINT LINE AREA
C2 IS ERRQUE CCW IN CORE
NO
YES
D2 IS ERRQUE CCW LESS THAN 8
YES
NO
E2 DETERMINE ADDRESS OF LAST CCW USED
F2 SET CMND CODE OF LAST USED CCW AND PLACE IN CCW PRINT LINE AREA
LACSW
G2 LD ADDR OF CSW FROM ERRQUE ENTRY
H2 UNPK KEY ADDRESS AND CNT FLDS OF CSW INTO PRINT LINE AREA
J2 TRANSLATE ASSEMBLED CSW INTO PRINTABLE CHARACTERS
5
UNPKSEN
K2 GET SENSE BYTE DATA FROM ERROR QUEUE AND UNPK DATA INTO PRINT LINE AREA
6
6
A3 TRANSLATE SENSE BYTE INTO PRINTABLE CHARACTERS
B3 UNPACK CCB ADDRESS INTO PRINTLINE AREA, TRANSLATE TO PRINTABLE CHARS
C3 DASD
NO
YES
D3 WAS ERROR A PROGRAM CHECK
YES
NO
7
E3 CCB ADDRESS AVAILABLE FOR DEV
NO
YES
F3 MOVE CCW ADDR FROM THE CCB INTO SAVE AREA
G3 GET SEEK ADDR FROM CCW, UNPK 1ST TWO BYTES (BB) INTO PRINT LINE AREA
UNPCH
H3 UNPK CCHH FROM SEEK ADDR AND TRANS ADDRESS TO PRINT CHARS
CALPH4
J3 SVC 3 FETCH $$ANERRP CHART LZ
2
NOCCB
B4 LOAD 11 INTO R1 POINT TO XXX IN TABLE OF SYMBOLIC UNITS
3
SYCLAS
C4 MPY SYM UNIT NO. BY 3 TO GET NAME TO SYM UNIT IDENT TABLE *C5
D4 LD TABLE ADDR INTO R2 ADD R1 DISPLACE TO GET PROPER SYMBOLIC NAME
E4 MOVE SYM NAME OF SYSTEM DEV FROM TABLE TO PRINT LINE AREA
4
7
*C5 IN CASE OF NO CCB AVAILABLE, THIS WILL BE 'XXX' AT END OF TABLE.

Chart LZ. $$ANERRP - Phase 4 of ERP Message Writer
Refer to Chart 08.

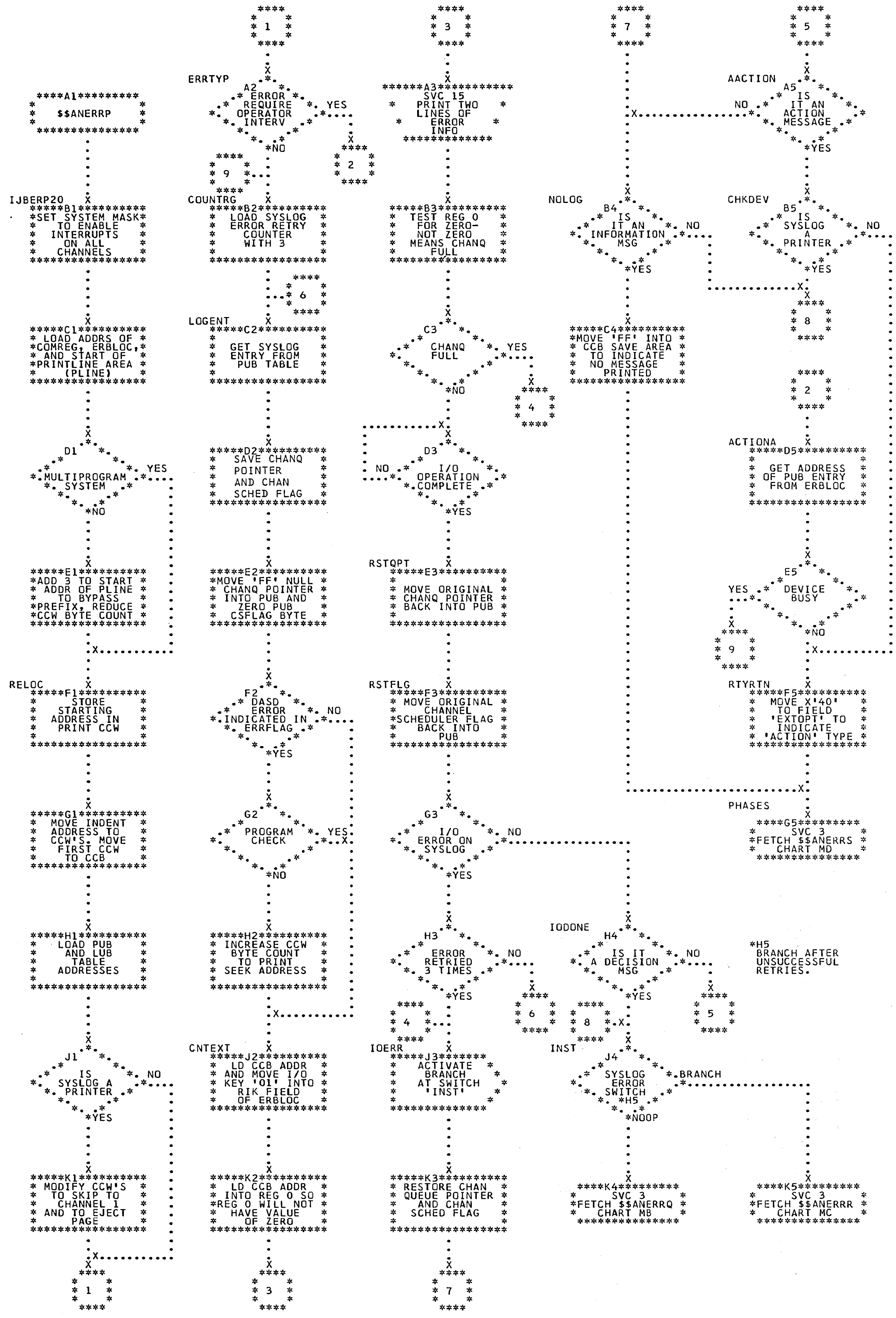
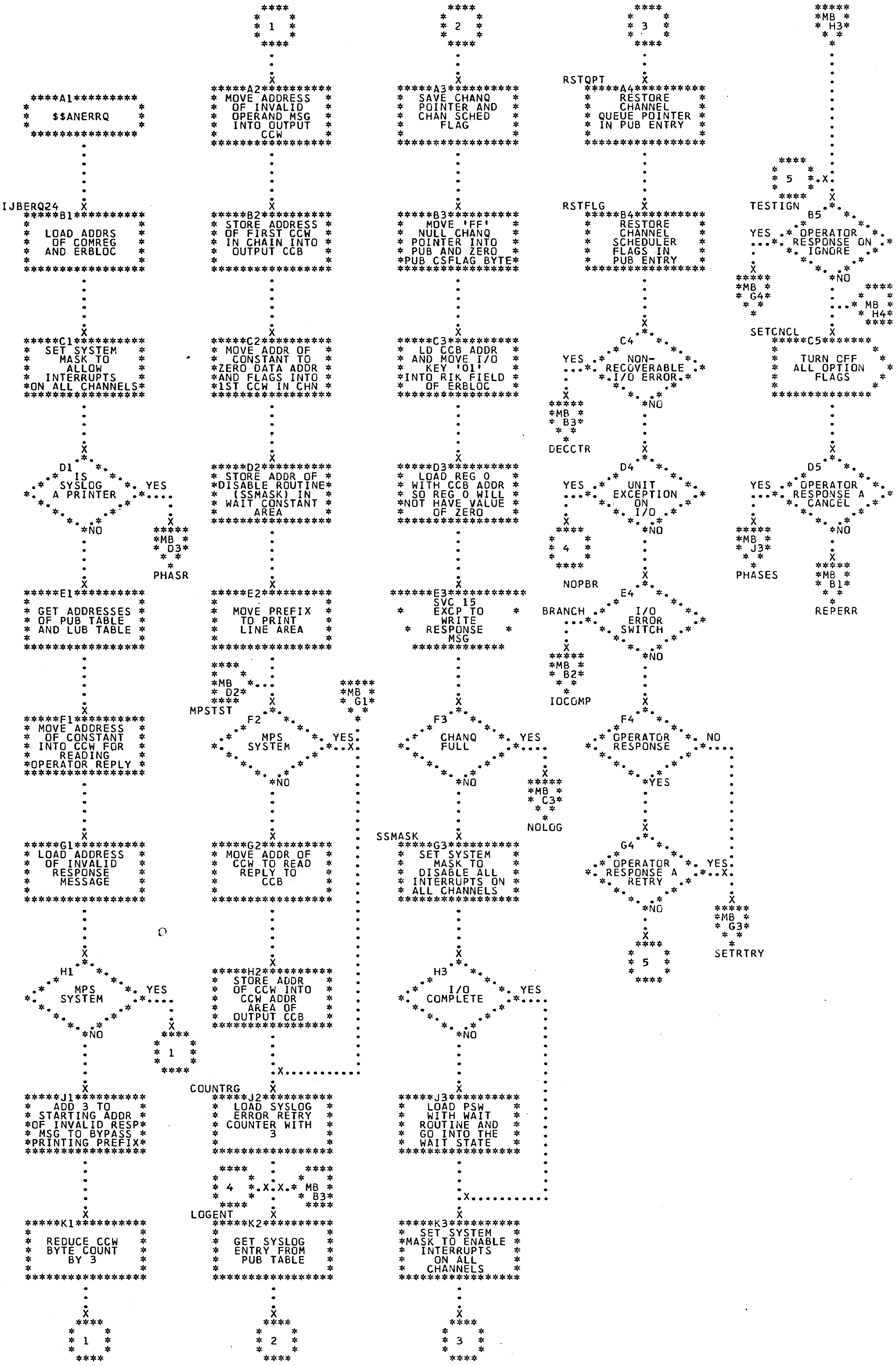

1
3
7
5
A1 $$ANERRP
ERRTYP A2 ERROR REQUIRE OPERATOR INTERV
YES
NO
2
9
A3 SVC 15 PRINT TWO LINES OF ERROR INFO
AACTION A5 IS IT AN ACTION MESSAGE
NO
YES
IJBERP20 B1 SET SYSTEM MASK TO ENABLE INTERRUPTS ON ALL CHANNELS
COUNTRG B2 LOAD SYSLOG ERROR RETRY COUNTER WITH 3
B3 TEST REG 0 FOR ZERO- NOT ZERO MEANS CHANQ FULL
NOLOG B4 IS IT AN INFORMATION MSG
NO
YES
CHKDEV B5 IS SYSLOG A PRINTER
NO
YES
6
8
C1 LOAD ADDRS OF COMREG, ERBLOC, AND START OF PRINTLINE AREA (PLINE)
LOGENT C2 GET SYSLOG ENTRY FROM PUB TABLE
C3 CHANQ FULL
YES
NO
4
C4 MOVE 'FF' INTO CCB SAVE AREA TO INDICATE NO MESSAGE PRINTED
2
D1 MULTIPROGRAM SYSTEM
YES
NO
D2 SAVE CHANQ POINTER AND CHAN SCHED FLAG
D3 I/O OPERATION COMPLETE
NO
YES
ACTIONA D5 GET ADDRESS OF PUB ENTRY FROM ERBLOC
E1 ADD 3 TO START ADDR OF PLINE TO BYPASS PREFIX, REDUCE CCW BYTE COUNT
E2 MOVE 'FF' NULL CHANQ POINTER INTO PUB AND ZERO PUB CSFLAG BYTE
RSTQPT E3 MOVE ORIGINAL CHANQ POINTER BACK INTO PUB
E5 DEVICE BUSY
YES
NO
9
RELOC F1 STORE STARTING ADDRESS IN PRINT CCW
F2 DASD ERROR INDICATED IN ERRFLAG
NO
YES
RSTFLG F3 MOVE ORIGINAL CHANNEL SCHEDULER FLAG BACK INTO PUB
RTYRTN F5 MOVE X'40' TO FIELD 'EXTOPT' TO INDICATE 'ACTION' TYPE
PHASES
G1 MOVE INDENT ADDRESS TO CCW'S. MOVE FIRST CCW TO CCB
G2 PROGRAM CHECK
YES
NO
G3 I/O ERROR ON SYSLOG
NO
YES
G5 SVC 3 FETCH $$ANERRS CHART MD
H1 LOAD PUB AND LUB TABLE ADDRESSES
H2 INCREASE CCW BYTE COUNT TO PRINT SEEK ADDRESS
H3 ERROR RETRIED 3 TIMES
NO
YES
IODONE H4 IS IT A DECISION MSG
NO
YES
*H5 BRANCH AFTER UNSUCCESSFUL RETRIES.
4
6
8
5
J1 IS SYSLOG A PRINTER
NO
YES
CNTEXT J2 LD CCB ADDR AND MOVE I/O KEY '01' INTO RIK FIELD OF ERBLOC
IOERR J3 ACTIVATE BRANCH AT SWITCH 'INST'
INST J4 SYSLOG ERROR SWITCH *H5
BRANCH
NOOP
K1 MODIFY CCW'S TO SKIP TO CHANNEL 1 AND TO EJECT PAGE
K2 LD CCB ADDR INTO REG 0 SO REG 0 WILL NOT HAVE VALUE OF ZERO
K3 RESTORE CHAN QUEUE POINTER AND CHAN SCHED FLAG
K4 SVC 3 FETCH $$ANERRQ CHART MB
K5 SVC 3 FETCH $$ANERRR CHART MC
1
3
7

Chart MA. $$ANERRQ - Phase 5 of ERP Message Writer (Part 1 of 2)
Refer to Chart 08.



1
2
3
MB H3
A1
$$ANERRQ
A2
MOVE ADDRESS OF INVALID OPERAND MSG INTO OUTPUT CCW
A3
SAVE CHANQ POINTER AND CHAN SCHED FLAG
RSTQPT
A4
RESTORE CHANNEL QUEUE POINTER IN PUB ENTRY
IJBERQ24
B1
LOAD ADDRS OF COMREG AND ERBLOC
B2
STORE ADDRESS OF FIRST CCW IN CHAIN INTO OUTPUT CCB
B3
MOVE 'FF' NULL CHANQ POINTER INTO PUB AND ZERO PUB CSFLAG BYTE
RSTFLG
B4
RESTORE CHANNEL SCHEDULER FLAGS IN PUB ENTRY
5
TESTIGN
B5
OPERATOR RESPONSE ON IGNORE
YES
MB G4
NO
MB H4
C1
SET SYSTEM MASK TO ALLOW INTERRUPTS ON ALL CHANNELS
C2
MOVE ADDR OF CONSTANT TO ZERO DATA ADDR AND FLAGS INTO 1ST CCW IN CHN
C3
LD CCB ADDR AND MOVE I/O KEY '01' INTO RIK FIELD OF ERBLOC
C4
NON-RECOVERABLE I/O ERROR
YES
MB B3
DECCTR
NO
SETCNCL
C5
TURN CFF ALL OPTION FLAGS
D1
IS SYSLOG A PRINTER
YES
MB D3
PHASR
NO
D2
STORE ADDR OF DISABLE ROUTINE (SSMASK) IN WAIT CONSTANT AREA
D3
LOAD REG 0 WITH CCB ADDR SO REG 0 WILL NOT HAVE VALUE OF ZERO
D4
UNIT EXCEPTION ON I/O
YES
4
NO
D5
OPERATOR RESPONSE A CANCEL
YES
MB J3
PHASES
NO
MB B1
REPERR
E1
GET ADDRESSES OF PUB TABLE AND LUB TABLE
E2
MOVE PREFIX TO PRINT LINE AREA
E3
SVC 15 EXCP TO WRITE RESPONSE MSG
NOPBR
E4
I/O ERROR SWITCH
BRANCH
MB B2
IOCOMP
NO
MB D2
MB G1
F1
MOVE ADDRESS OF CONSTANT INTO CCW FOR READING OPERATOR REPLY
MPSTST
F2
MPS SYSTEM
YES
NO
F3
CHANQ FULL
YES
MB C3
NOLOG
NO
F4
OPERATOR RESPONSE
NO
YES
G1
LOAD ADDRESS OF INVALID RESPONSE MESSAGE
G2
MOVE ADDR OF CCW TO READ REPLY TO CCB
SSMASK
G3
SET SYSTEM MASK TO DISABLE ALL INTERRUPTS ON ALL CHANNELS
G4
OPERATOR RESPONSE A RETRY
YES
MB G3
SETRTRY
NO
5
H1
MPS SYSTEM
YES
1
NO
H2
STORE ADDR OF CCW INTO CCW ADDR AREA OF OUTPUT CCB
H3
I/O COMPLETE
YES
NO
J1
ADD 3 TO STARTING ADDR OF INVALID RESP MSG TO BYPASS PRINTING PREFIX
COUNTRG
J2
LOAD SYSLOG ERROR RETRY COUNTER WITH 3
J3
LOAD PSW WITH WAIT ROUTINE AND GO INTO THE WAIT STATE
4
MB B3
LOGENT
K1
REDUCE CCW BYTE COUNT BY 3
K2
GET SYSLOG ENTRY FROM PUB TABLE
K3
SET SYSTEM MASK TO ENABLE INTERRUPTS ON ALL CHANNELS
1
2
3

Chart MB. $$ANERRQ - Phase 5 of ERP Message Writer (Part 2 of 2)
Refer to Chart 08.

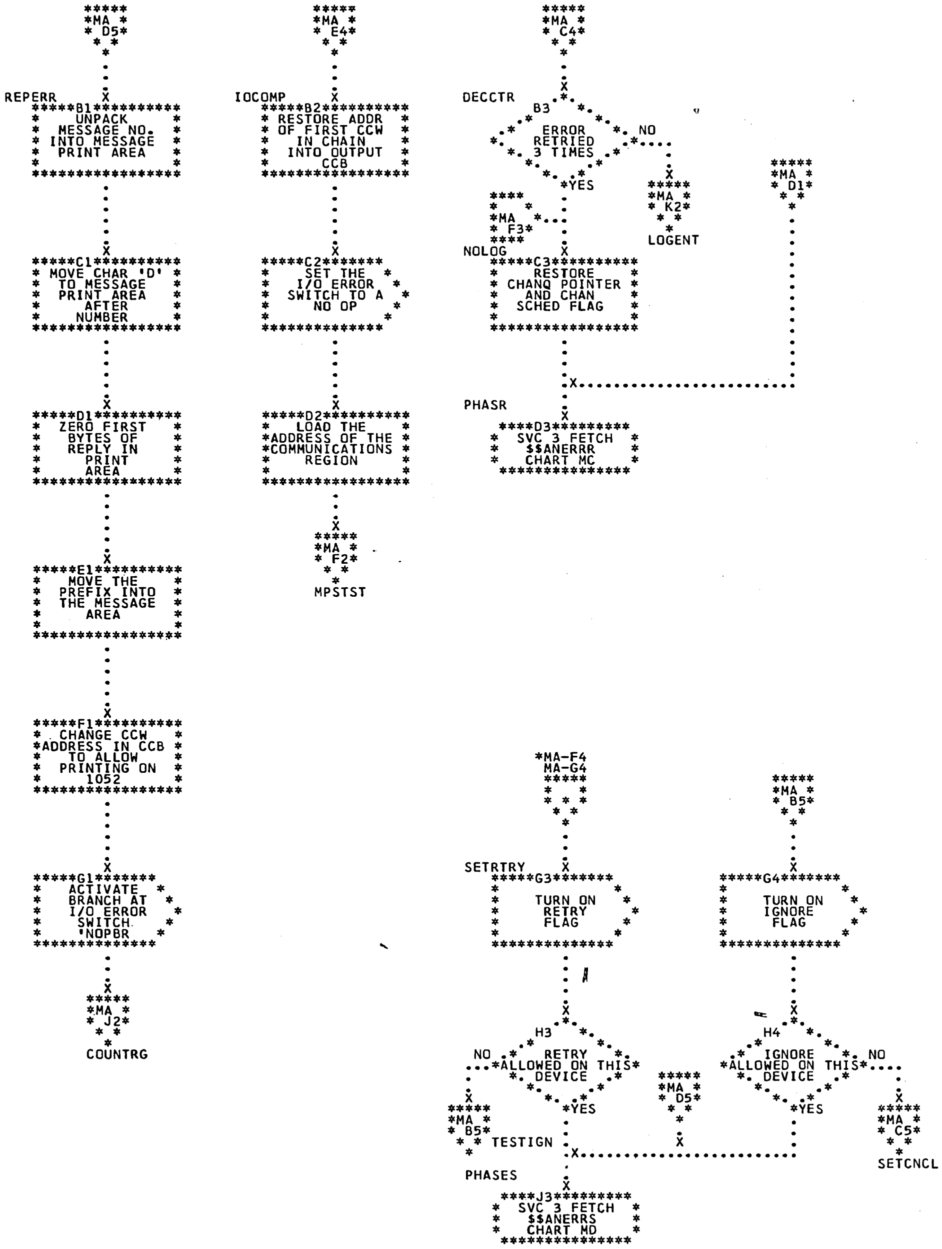

MA D5
REPERR
B1
UNPACK MESSAGE NO. INTO MESSAGE PRINT AREA
C1
MOVE CHAR 'D' TO MESSAGE PRINT AREA AFTER NUMBER
D1
ZERO FIRST BYTES OF REPLY IN PRINT AREA
E1
MOVE THE PREFIX INTO THE MESSAGE AREA
F1
CHANGE CCW ADDRESS IN CCB TO ALLOW PRINTING ON 1052
G1
ACTIVATE BRANCH AT I/O ERROR SWITCH 'NOPBR
MA J2
COUNTRG
MA E4
IOCOMP
B2
RESTORE ADDR OF FIRST CCW IN CHAIN INTO OUTPUT CCB
C2
SET THE I/O ERROR SWITCH TO A NO OP
D2
LOAD THE ADDRESS OF THE COMMUNICATIONS REGION
MA F2
MPSTST
MA C4
DECCTR
B3
ERROR RETRIED 3 TIMES
NO
YES
MA K2
LOGENT
MA F3
NOLOG
C3
RESTORE CHANQ POINTER AND CHAN SCHED FLAG
MA D1
PHASR
D3
SVC 3 FETCH $$ANERRR CHART MC
*MA-F4
MA-G4
SETRTRY
G3
TURN ON RETRY FLAG
H3
RETRY ALLOWED ON THIS DEVICE
NO
YES
MA B5
TESTIGN
MA D5
PHASES
J3
SVC 3 FETCH $$ANERRS CHART MD
MA B5
G4
TURN ON IGNORE FLAG
H4
IGNORE ALLOWED ON THIS DEVICE
NO
YES
MA C5
SETCNCL

Chart MC. $$ANERRR - Phase 6 of ERP Message Writer
Refer to Chart 08.

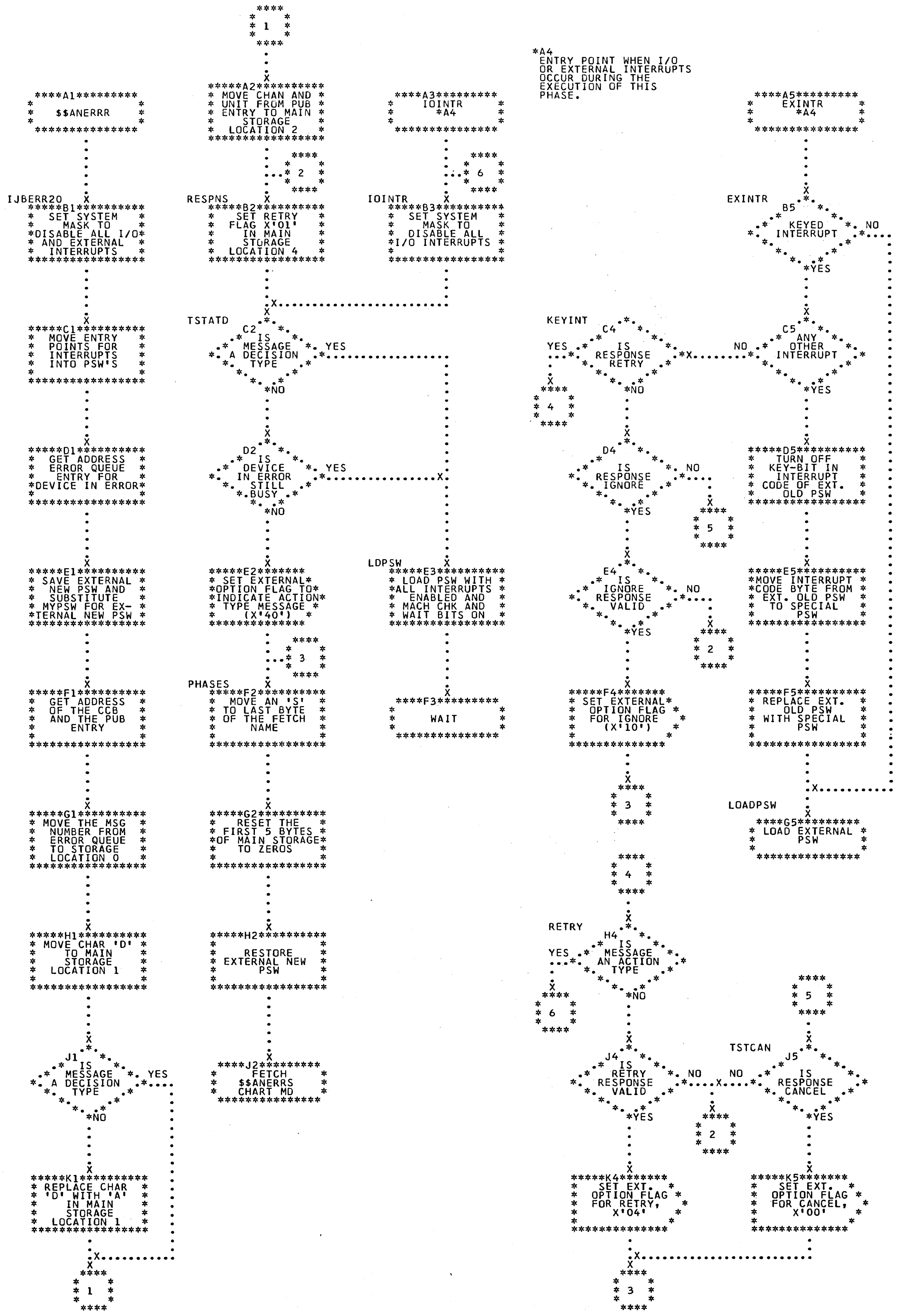
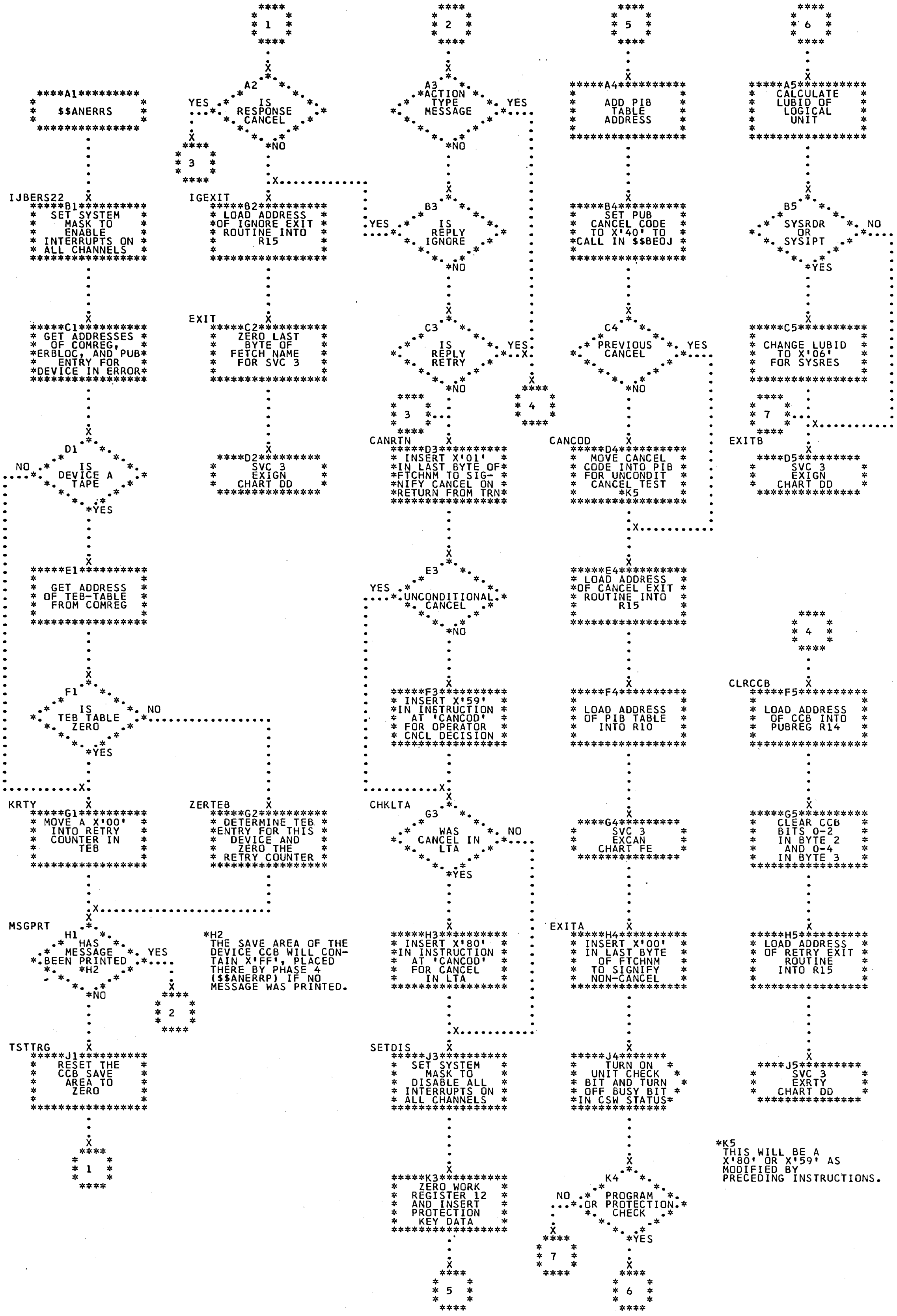

1
A1 $$ANERRR
IJBERR20
B1 SET SYSTEM MASK TO DISABLE ALL I/O AND EXTERNAL INTERRUPTS
C1 MOVE ENTRY POINTS FOR INTERRUPTS INTO PSW'S
D1 GET ADDRESS ERROR QUEUE ENTRY FOR DEVICE IN ERROR
E1 SAVE EXTERNAL NEW PSW AND SUBSTITUTE MYPSW FOR EX- TERNAL NEW PSW
F1 GET ADDRESS OF THE CCB AND THE PUB ENTRY
G1 MOVE THE MSG NUMBER FROM ERROR QUEUE TO STORAGE LOCATION 0
H1 MOVE CHAR 'D' TO MAIN STORAGE LOCATION 1
J1 IS MESSAGE A DECISION TYPE
YES
NO
K1 REPLACE CHAR 'D' WITH 'A' IN MAIN STORAGE LOCATION 1
1
1
A2 MOVE CHAN AND UNIT FROM PUB ENTRY TO MAIN STORAGE LOCATION 2
2
RESPNS
B2 SET RETRY FLAG X'01' IN MAIN STORAGE LOCATION 4
TSTATD
C2 IS MESSAGE A DECISION TYPE
YES
NO
D2 IS DEVICE IN ERROR STILL BUSY
YES
NO
E2 SET EXTERNAL OPTION FLAG TO INDICATE ACTION TYPE MESSAGE (X'40')
3
PHASES
F2 MOVE AN 'S' TO LAST BYTE OF THE FETCH NAME
G2 RESET THE FIRST 5 BYTES OF MAIN STORAGE TO ZEROS
H2 RESTORE EXTERNAL NEW PSW
J2 FETCH $$ANERRS CHART MD
A3 IOINTR *A4
6
IOINTR
B3 SET SYSTEM MASK TO DISABLE ALL I/O INTERRUPTS
LDPSW
E3 LOAD PSW WITH ALL INTERRUPTS ENABLED AND MACH CHK AND WAIT BITS ON
F3 WAIT
*A4
ENTRY POINT WHEN I/O
OR EXTERNAL INTERRUPTS
OCCUR DURING THE
EXECUTION OF THIS
PHASE.
KEYINT
C4 IS RESPONSE RETRY
YES
NO
4
D4 IS RESPONSE IGNORE
NO
YES
5
E4 IS IGNORE RESPONSE VALID
NO
YES
2
F4 SET EXTERNAL OPTION FLAG FOR IGNORE (X'10')
3
4
RETRY
H4 IS MESSAGE AN ACTION TYPE
YES
NO
6
J4 IS RETRY RESPONSE VALID
NO
YES
2
K4 SET EXT. OPTION FLAG FOR RETRY, X'04'
3
A5 EXINTR *A4
EXINTR
B5 KEYED INTERRUPT
NO
YES
C5 ANY OTHER INTERRUPT
NO
YES
D5 TURN OFF KEY-BIT IN INTERRUPT CODE OF EXT. OLD PSW
E5 MOVE INTERRUPT CODE BYTE FROM EXT. OLD PSW TO SPECIAL PSW
F5 REPLACE EXT. OLD PSW WITH SPECIAL PSW
LOADPSW
G5 LOAD EXTERNAL PSW
5
TSTCAN
J5 IS RESPONSE CANCEL
NO
YES
K5 SET EXT. OPTION FLAG FOR CANCEL, X'00'

Chart MD. $$ANERRS - Phase 7 of ERP Message Writer
Refer to Chart 08.



A1 $$ANERRS
IJBERS22
B1 SET SYSTEM MASK TO ENABLE INTERRUPTS ON ALL CHANNELS
C1 GET ADDRESSES OF COMREG, ERBLOC, AND PUB ENTRY FOR DEVICE IN ERROR
D1 IS DEVICE A TAPE
NO
YES
E1 GET ADDRESS OF TEB-TABLE FROM COMREG
F1 IS TEB TABLE ZERO
NO
YES
KRTY
G1 MOVE A X'00' INTO RETRY COUNTER IN TEB
MSGPRT
H1 HAS MESSAGE BEEN PRINTED *H2
YES
NO
TSTTRG
J1 RESET THE CCB SAVE AREA TO ZERO
1
1
A2 IS RESPONSE CANCEL
YES
NO
3
IGEXIT
B2 LOAD ADDRESS OF IGNORE EXIT ROUTINE INTO R15
EXIT
C2 ZERO LAST BYTE OF FETCH NAME FOR SVC 3
D2 SVC 3 EXIGN CHART DD
ZERTEB
G2 DETERMINE TEB ENTRY FOR THIS DEVICE AND ZERO THE RETRY COUNTER
*H2
THE SAVE AREA OF THE DEVICE CCB WILL CONTAIN X'FF', PLACED THERE BY PHASE 4 ($$ANERRP) IF NO MESSAGE WAS PRINTED.
2
2
A3 ACTION TYPE MESSAGE
YES
NO
B3 IS REPLY IGNORE
YES
NO
C3 IS REPLY RETRY
YES
NO
3
4
CANRTN
D3 INSERT X'01' IN LAST BYTE OF FTCHNM TO SIGNIFY CANCEL ON RETURN FROM TRN
E3 UNCONDITIONAL CANCEL
YES
NO
F3 INSERT X'59' IN INSTRUCTION AT 'CANCOD' FOR OPERATOR CNCL DECISION
CHKLTA
G3 WAS CANCEL IN LTA
NO
YES
H3 INSERT X'80' IN INSTRUCTION AT 'CANCOD' FOR CANCEL IN LTA
SETDIS
J3 SET SYSTEM MASK TO DISABLE ALL INTERRUPTS ON ALL CHANNELS
K3 ZERO WORK REGISTER 12 AND INSERT PROTECTION KEY DATA
5
5
A4 ADD PIB TABLE ADDRESS
B4 SET PUB CANCEL CODE TO X'40' TO CALL IN $$BEOJ
C4 PREVIOUS CANCEL
YES
NO
CANCOD
D4 MOVE CANCEL CODE INTO PIB FOR UNCONDIT CANCEL TEST *K5
E4 LOAD ADDRESS OF CANCEL EXIT ROUTINE INTO R15
F4 LOAD ADDRESS OF PIB TABLE INTO R10
G4 SVC 3 EXCAN CHART FE
EXITA
H4 INSERT X'00' IN LAST BYTE OF FTCHNM TO SIGNIFY NON-CANCEL
J4 TURN ON UNIT CHECK BIT AND TURN OFF BUSY BIT IN CSW STATUS
K4 PROGRAM OR PROTECTION CHECK
NO
YES
7
6
6
A5 CALCULATE LUBID OF LOGICAL UNIT
B5 SYSRDR OR SYSIPT
NO
YES
C5 CHANGE LUBID TO X'06' FOR SYSRES
7
EXITB
D5 SVC 3 EXIGN CHART DD
4
CLRCCB
F5 LOAD ADDRESS OF CCB INTO PUBREG R14
G5 CLEAR CCB BITS 0-2 IN BYTE 2 AND 0-4 IN BYTE 3
H5 LOAD ADDRESS OF RETRY EXIT ROUTINE INTO R15
J5 SVC 3 EXRTY CHART DD
*K5
THIS WILL BE A X'80' OR X'59' AS MODIFIED BY PRECEDING INSTRUCTIONS.

Chart ME. $$ANERRT - MICR (1412/1419) ERP
Refer to Chart 08.

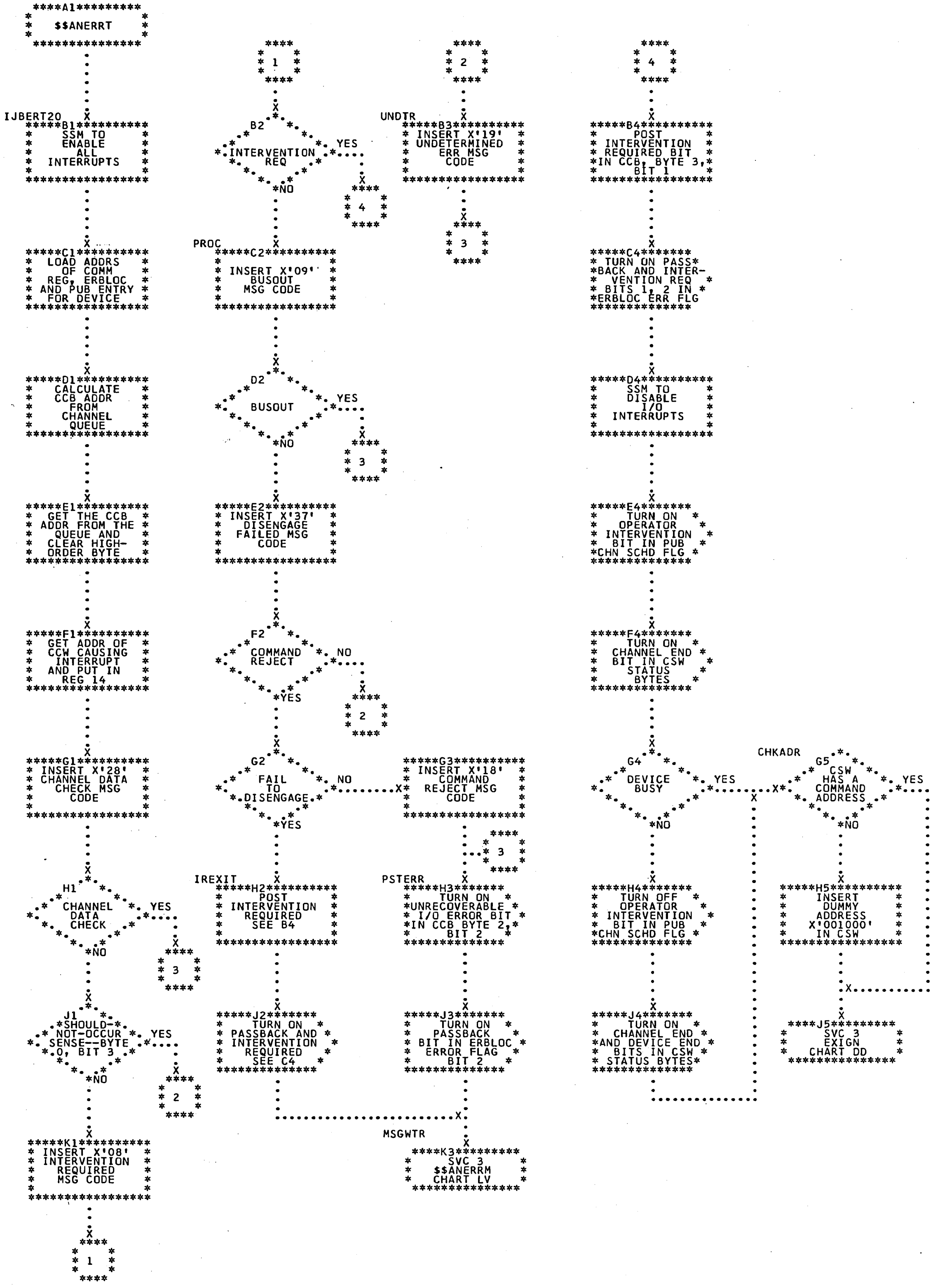
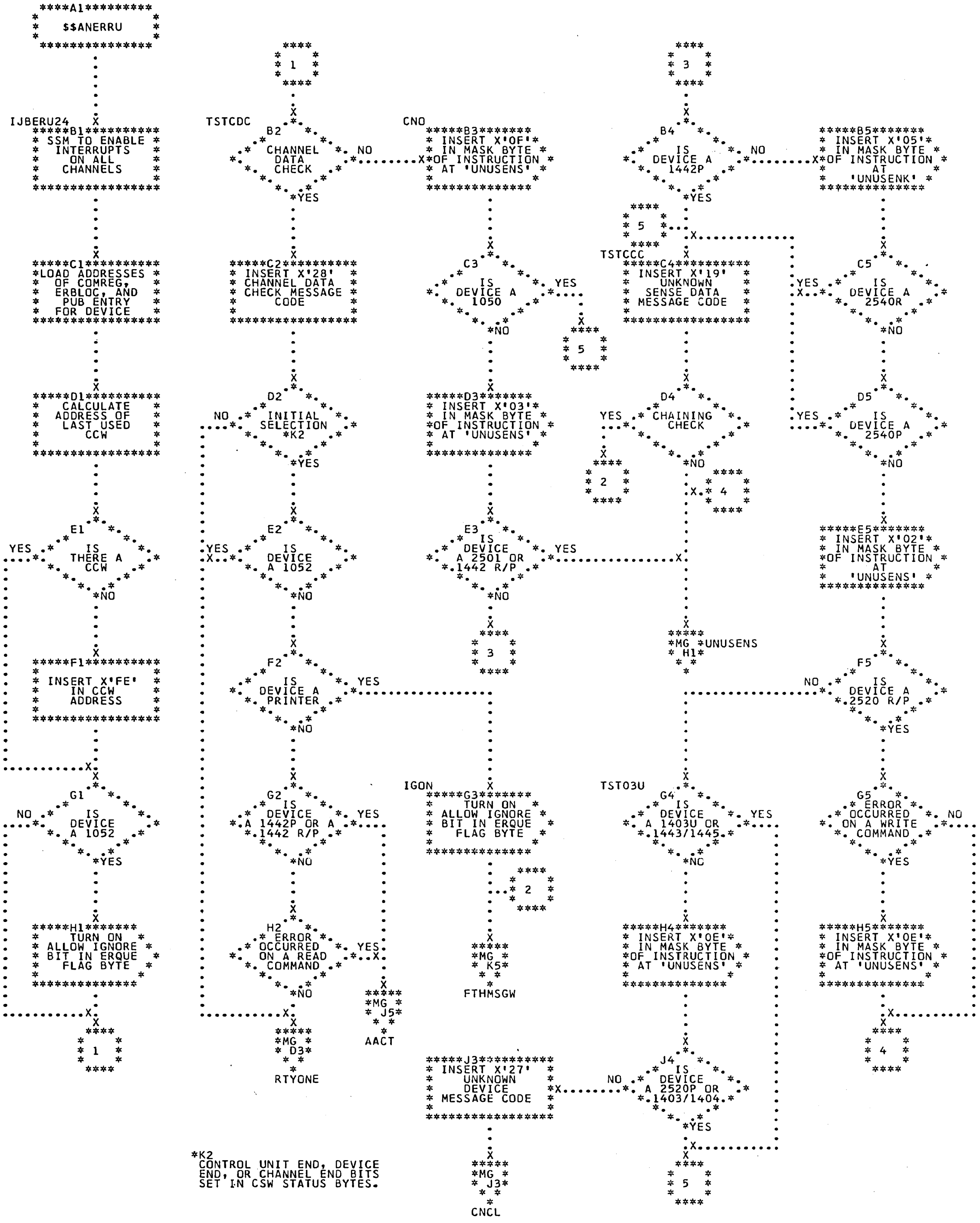

A1
$$ANERRT
IJBERT20
B1
SSM TO ENABLE ALL INTERRUPTS
C1
LOAD ADDRS OF COMM REG, ERBLOC AND PUB ENTRY FOR DEVICE
D1
CALCULATE CCB ADDR FROM CHANNEL QUEUE
E1
GET THE CCB ADDR FROM THE QUEUE AND CLEAR HIGH-ORDER BYTE
F1
GET ADDR OF CCW CAUSING INTERRUPT AND PUT IN REG 14
G1
INSERT X'28' CHANNEL DATA CHECK MSG CODE
H1
CHANNEL DATA CHECK
YES
3
NO
J1
SHOULD-NOT-OCCUR SENSE--BYTE 0, BIT 3
YES
2
NO
K1
INSERT X'08' INTERVENTION REQUIRED MSG CODE
1
1
B2
INTERVENTION REQ
YES
4
NO
PROC
C2
INSERT X'09' BUSOUT MSG CODE
D2
BUSOUT
YES
3
NO
E2
INSERT X'37' DISENGAGE FAILED MSG CODE
F2
COMMAND REJECT
NO
2
YES
G2
FAIL TO DISENGAGE
NO
YES
IREXIT
H2
POST INTERVENTION REQUIRED SEE B4
J2
TURN ON PASSBACK AND INTERVENTION REQUIRED SEE C4
2
UNDTR
B3
INSERT X'19' UNDETERMINED ERR MSG CODE
3
G3
INSERT X'18' COMMAND REJECT MSG CODE
3
PSTERR
H3
TURN ON UNRECOVERABLE I/O ERROR BIT IN CCB BYTE 2, BIT 2
J3
TURN ON PASSBACK BIT IN ERBLOC ERROR FLAG BIT 2
MSGWTR
K3
SVC 3 $$ANERRM CHART LV
4
B4
POST INTERVENTION REQUIRED BIT IN CCB, BYTE 3, BIT 1
C4
TURN ON PASS-BACK AND INTER-VENTION REQ BITS 1, 2 IN ERBLOC ERR FLG
D4
SSM TO DISABLE I/O INTERRUPTS
E4
TURN ON OPERATOR INTERVENTION BIT IN PUB CHN SCHD FLG
F4
TURN ON CHANNEL END BIT IN CSW STATUS BYTES
G4
DEVICE BUSY
YES
NO
H4
TURN OFF OPERATOR INTERVENTION BIT IN PUB CHN SCHD FLG
J4
TURN ON CHANNEL END AND DEVICE END BITS IN CSW STATUS BYTES
CHKADR
G5
CSW HAS A COMMAND ADDRESS
YES
NO
H5
INSERT DUMMY ADDRESS X'001000' IN CSW
J5
SVC 3 EXIGN CHART DD

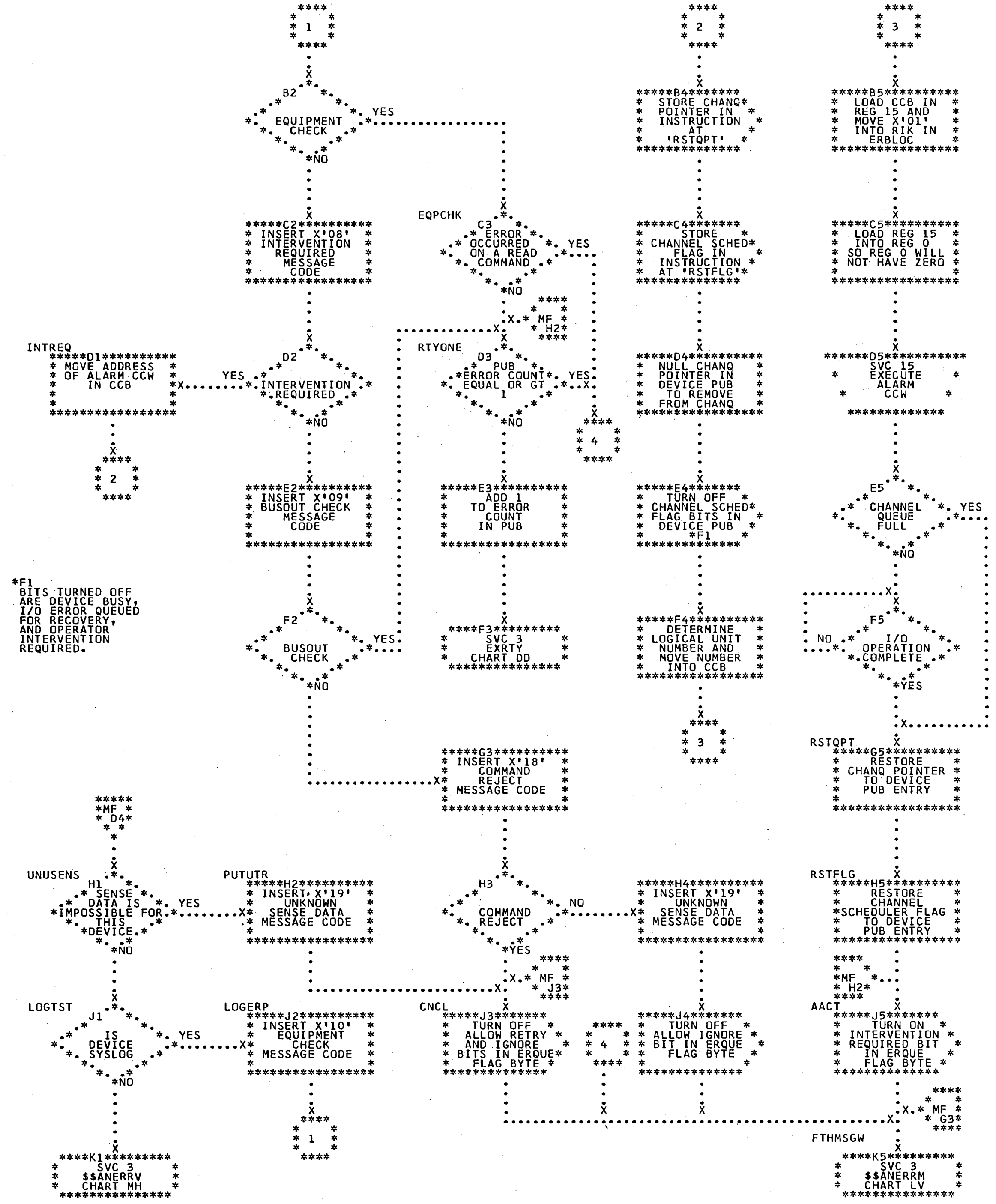
Chart MF. $$ANERRU - Phase 1 of Unit Record ERP (Part 1 of 2)
Refer to Chart 07.



A1 $$ANERRU
1
3
IJBERU24
B1 SSM TO ENABLE INTERRUPTS ON ALL CHANNELS
TSTCDC
B2 CHANNEL DATA CHECK
NO
YES
CNO
B3 INSERT X'0F' IN MASK BYTE OF INSTRUCTION AT 'UNUSENS'
B4 IS DEVICE A 1442P
NO
YES
B5 INSERT X'05' IN MASK BYTE OF INSTRUCTION AT 'UNUSENK'
5
C1 LOAD ADDRESSES OF COMREG, ERBLOC, AND PUB ENTRY FOR DEVICE
C2 INSERT X'28' CHANNEL DATA CHECK MESSAGE CODE
C3 IS DEVICE A 1050
YES
NO
5
TSTCCC
C4 INSERT X'19' UNKNOWN SENSE DATA MESSAGE CODE
C5 IS DEVICE A 2540R
YES
NO
D1 CALCULATE ADDRESS OF LAST USED CCW
D2 INITIAL SELECTION *K2
NO
YES
D3 INSERT X'03' IN MASK BYTE OF INSTRUCTION AT 'UNUSENS'
D4 CHAINING CHECK
YES
NO
2
4
D5 IS DEVICE A 2540P
YES
NO
E1 IS THERE A CCW
YES
NO
E2 IS DEVICE A 1052
YES
NO
E3 IS DEVICE A 2501 OR 1442 R/P
YES
NO
3
E5 INSERT X'02' IN MASK BYTE OF INSTRUCTION AT 'UNUSENS'
MG H1 UNUSENS
F1 INSERT X'FE' IN CCW ADDRESS
F2 IS DEVICE A PRINTER
YES
NO
F5 IS DEVICE A 2520 R/P
NO
YES
G1 IS DEVICE A 1052
NO
YES
G2 IS DEVICE A 1442P OR A 1442 R/P
YES
NO
IGON
G3 TURN ON ALLOW IGNORE BIT IN ERQUE FLAG BYTE
TST03U
G4 IS DEVICE A 1403U OR 1443/1445
YES
NO
G5 ERROR OCCURRED ON A WRITE COMMAND
NO
YES
2
H1 TURN ON ALLOW IGNORE BIT IN ERQUE FLAG BYTE
H2 ERROR OCCURRED ON A READ COMMAND
YES
NO
MG K5 FTHMSGW
H4 INSERT X'0E' IN MASK BYTE OF INSTRUCTION AT 'UNUSENS'
H5 INSERT X'0E' IN MASK BYTE OF INSTRUCTION AT 'UNUSENS'
MG J5 AACT
1
MG D3 RTYONE
J3 INSERT X'27' UNKNOWN DEVICE MESSAGE CODE
J4 IS DEVICE A 2520P OR 1403/1404
NO
YES
4
MG J3 CNCL
5

*K2
CONTROL UNIT END, DEVICE END, OR CHANNEL END BITS SET IN CSW STATUS BYTES.

Chart MG. $$ANERRU - Phase 1 of Unit Record ERP (Part 2 of 2)
Refer to Chart 07.


1
B2 EQUIPMENT CHECK
YES
NO
C2 INSERT X'08' INTERVENTION REQUIRED MESSAGE CODE
EQPCHK
C3 ERROR OCCURRED ON A READ COMMAND
YES
NO
MF H2
INTREQ
D1 MOVE ADDRESS OF ALARM CCW IN CCB
D2 INTERVENTION REQUIRED
YES
NO
2
RTYONE
D3 PUB ERROR COUNT EQUAL OR GT 1
YES
NO
4
E2 INSERT X'09' BUSOUT CHECK MESSAGE CODE
E3 ADD 1 TO ERROR COUNT IN PUB
F2 BUSOUT CHECK
YES
NO
F3 SVC 3 EXRTY CHART DD
G3 INSERT X'18' COMMAND REJECT MESSAGE CODE
MF D4
UNUSENS
H1 SENSE DATA IS IMPOSSIBLE FOR THIS DEVICE
YES
NO
PUTUTR
H2 INSERT X'19' UNKNOWN SENSE DATA MESSAGE CODE
H3 COMMAND REJECT
NO
YES
MF J3
LOGTST
J1 IS DEVICE SYSLOG
YES
NO
LOGERP
J2 INSERT X'10' EQUIPMENT CHECK MESSAGE CODE
1
K1 SVC 3 $$ANERRV CHART MH
CNCL
J3 TURN OFF ALLOW RETRY AND IGNORE BITS IN ERQUE FLAG BYTE
4
2
B4 STORE CHANQ POINTER IN INSTRUCTION AT 'RSTQPT'
C4 STORE CHANNEL SCHED FLAG IN INSTRUCTION AT 'RSTFLG'
D4 NULL CHANQ POINTER IN DEVICE PUB TO REMOVE FROM CHANQ
E4 TURN OFF CHANNEL SCHED FLAG BITS IN DEVICE PUB *F1
F4 DETERMINE LOGICAL UNIT NUMBER AND MOVE NUMBER INTO CCB
3
H4 INSERT X'19' UNKNOWN SENSE DATA MESSAGE CODE
J4 TURN OFF ALLOW IGNORE BIT IN ERQUE FLAG BYTE
3
B5 LOAD CCB IN REG 15 AND MOVE X'01' INTO RIK IN ERBLOC
C5 LOAD REG 15 INTO REG 0 SO REG 0 WILL NOT HAVE ZERO
D5 SVC 15 EXECUTE ALARM CCW
E5 CHANNEL QUEUE FULL
YES
NO
F5 I/O OPERATION COMPLETE
NO
YES
RSTQPT
G5 RESTORE CHANQ POINTER TO DEVICE PUB ENTRY
RSTFLG
H5 RESTORE CHANNEL SCHEDULER FLAG TO DEVICE PUB ENTRY
MF H2
AACT
J5 TURN ON INTERVENTION REQUIRED BIT IN ERQUE FLAG BYTE
MF G3
FTHMSGW
K5 SVC 3 $$ANERRM CHART LV

*F1
BITS TURNED OFF
ARE DEVICE BUSY,
I/O ERROR QUEUED
FOR RECOVERY,
AND OPERATOR
INTERVENTION
REQUIRED.

Chart MH. $$ANERRV - Phase 2 of Unit Record ERP (Part 1 of 2)
Refer to Chart 07.

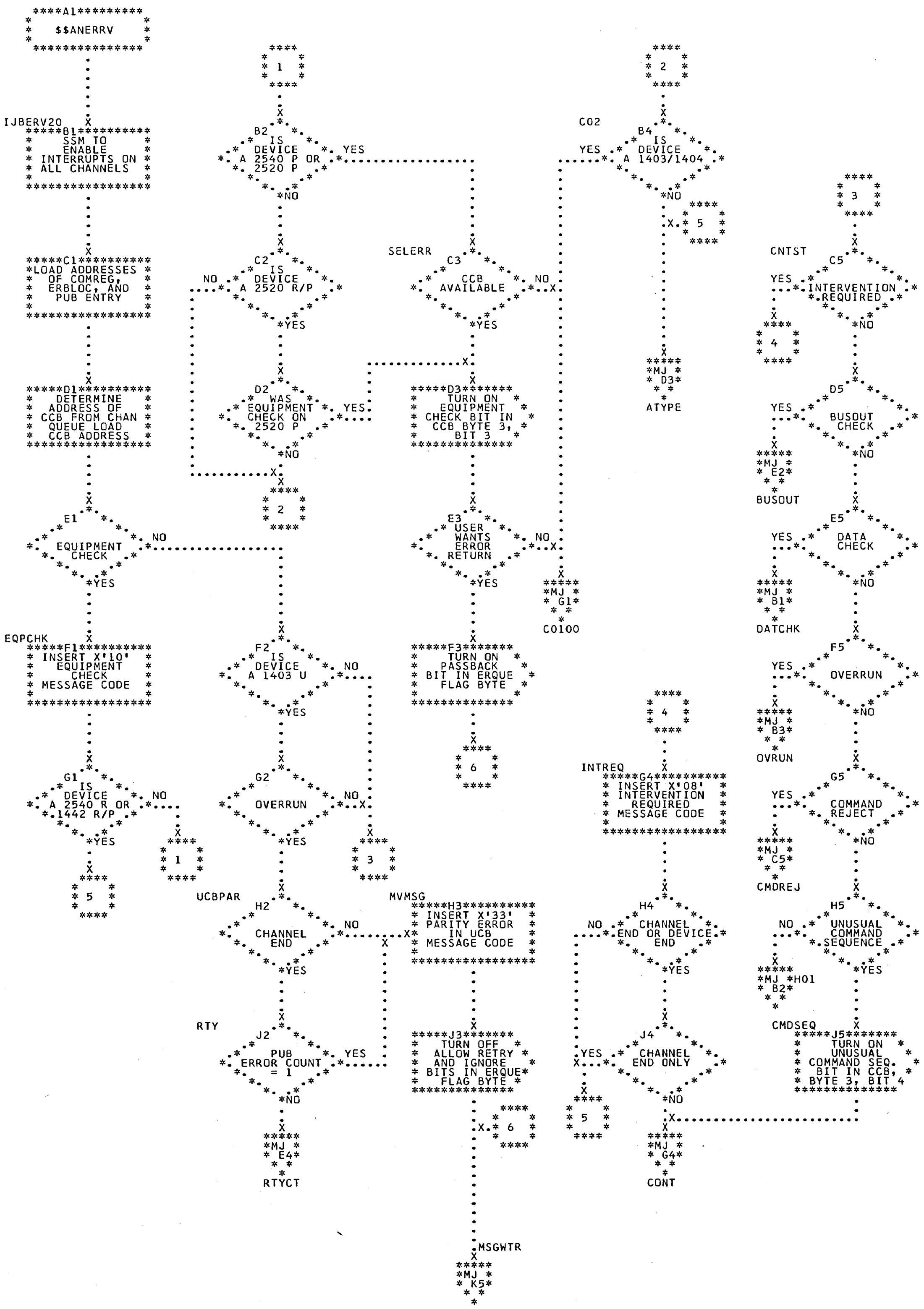

A1
$$ANERRV
IJBERV20
B1
SSM TO ENABLE INTERRUPTS ON ALL CHANNELS
C1
LOAD ADDRESSES OF COMREG, ERBLOC, AND PUB ENTRY
D1
DETERMINE ADDRESS OF CCB FROM CHAN QUEUE LOAD CCB ADDRESS
E1
EQUIPMENT CHECK
NO
YES
EQPCHK
F1
INSERT X'10' EQUIPMENT CHECK MESSAGE CODE
G1
IS DEVICE A 2540 R OR 1442 R/P
NO
YES
1
5
1
B2
IS DEVICE A 2540 P OR 2520 P
YES
NO
C2
IS DEVICE A 2520 R/P
NO
YES
D2
WAS EQUIPMENT CHECK ON 2520 P
YES
NO
2
F2
IS DEVICE A 1403 U
NO
YES
G2
OVERRUN
NO
YES
3
UCBPAR
H2
CHANNEL END
NO
YES
RTY
J2
PUB ERROR COUNT = 1
YES
NO
MJ
E4
RTYCT
SELERR
C3
CCB AVAILABLE
NO
YES
D3
TURN ON EQUIPMENT CHECK BIT IN CCB BYTE 3, BIT 3
E3
USER WANTS ERROR RETURN
NO
YES
MJ
G1
C0100
F3
TURN ON PASSBACK BIT IN ERQUE FLAG BYTE
6
MVMSG
H3
INSERT X'33' PARITY ERROR IN UCB MESSAGE CODE
J3
TURN OFF ALLOW RETRY AND IGNORE BITS IN ERQUE FLAG BYTE
6
MSGWTR
MJ
K5
2
C02
B4
IS DEVICE A 1403/1404
YES
NO
5
MJ
D3
ATYPE
4
INTREQ
G4
INSERT X'08' INTERVENTION REQUIRED MESSAGE CODE
H4
CHANNEL END OR DEVICE END
NO
YES
J4
CHANNEL END ONLY
YES
NO
5
MJ
G4
CONT
3
CNTST
C5
INTERVENTION REQUIRED
YES
NO
4
D5
BUSOUT CHECK
YES
NO
MJ
E2
BUSOUT
E5
DATA CHECK
YES
NO
MJ
B1
DATCHK
F5
OVERRUN
YES
NO
MJ
B3
OVRUN
G5
COMMAND REJECT
YES
NO
MJ
C5
CMDREJ
H5
UNUSUAL COMMAND SEQUENCE
NO
YES
MJ
H01
B2
CMDSEQ
J5
TURN ON UNUSUAL COMMAND SEQ. BIT IN CCB, BYTE 3, BIT 4

Chart MJ. $$ANERRV - Phase 2 of Unit Record ERP (Part 2 of 2)
Refer to Chart 07.

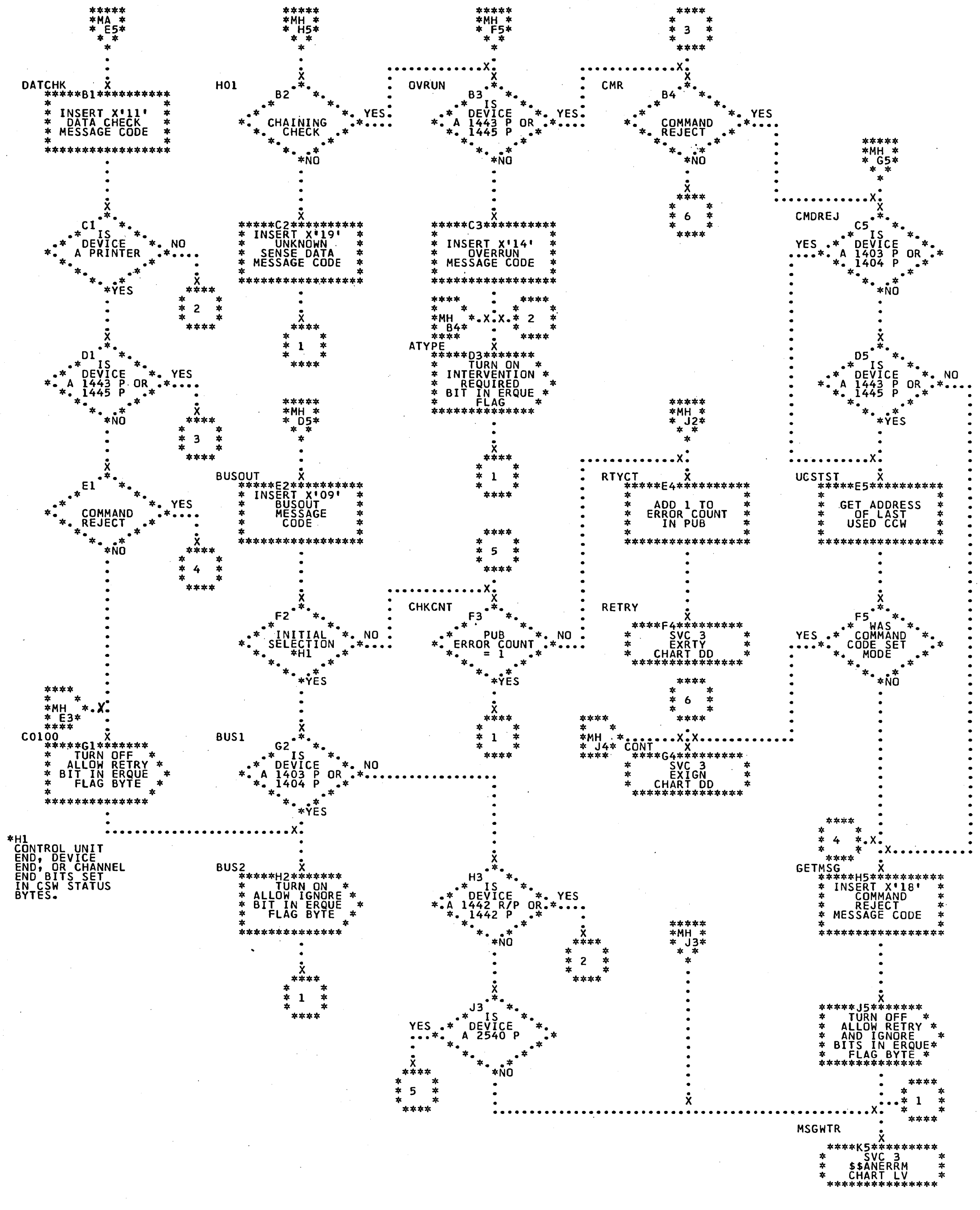
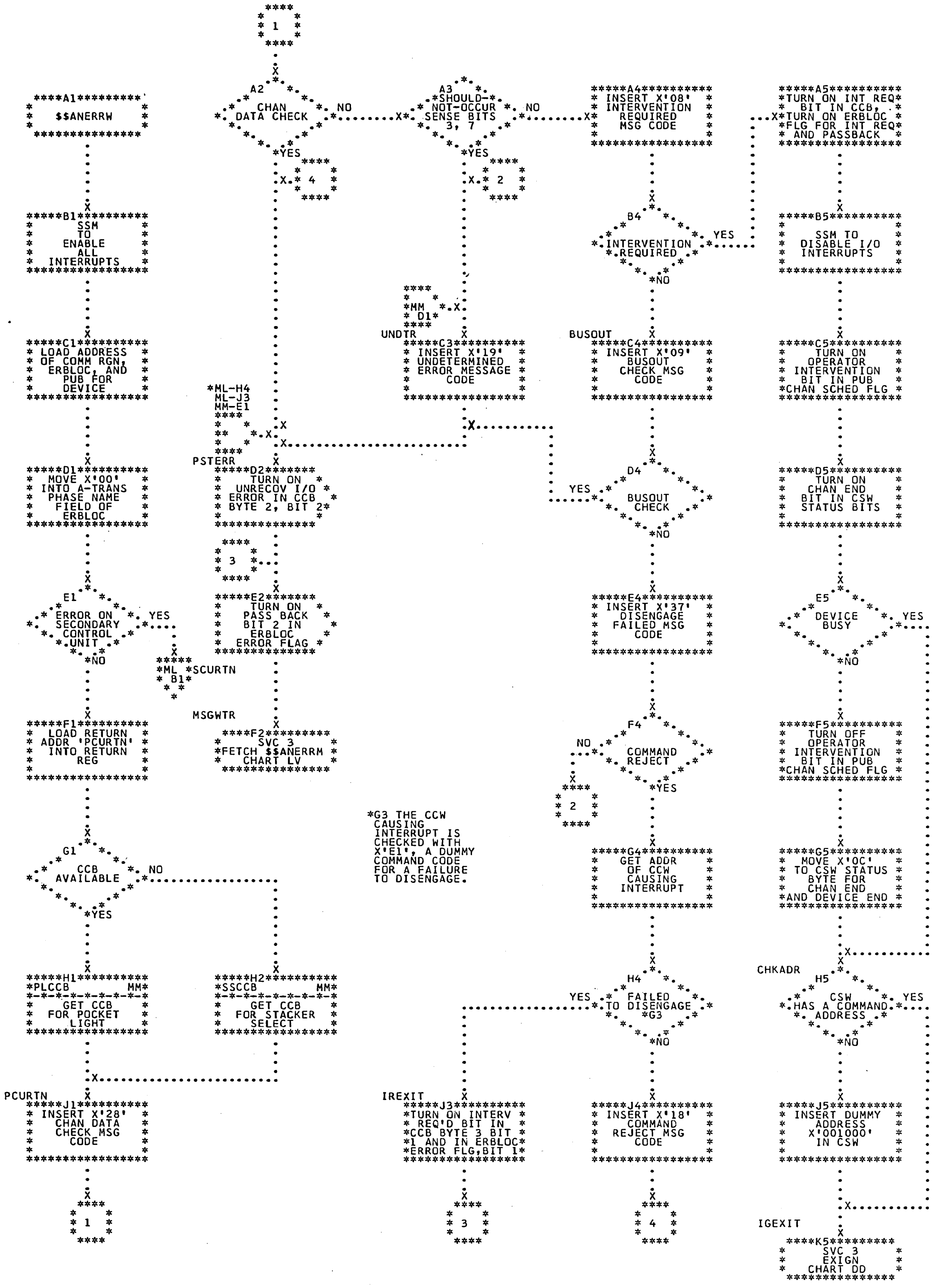

MA E5
DATCHK
B1
INSERT X'11' DATA CHECK MESSAGE CODE
C1
IS DEVICE A PRINTER
NO
2
YES
D1
IS DEVICE A 1443 P OR 1445 P
YES
3
NO
E1
COMMAND REJECT
YES
4
NO
MH E3
C0100
G1
TURN OFF ALLOW RETRY BIT IN ERQUE FLAG BYTE
*H1
CONTROL UNIT END, DEVICE END, OR CHANNEL END BITS SET IN CSW STATUS BYTES.
MH H5
H01
B2
CHAINING CHECK
YES
NO
C2
INSERT X'19' UNKNOWN SENSE DATA MESSAGE CODE
1
MH D5
BUSOUT
E2
INSERT X'09' BUSOUT MESSAGE CODE
F2
INITIAL SELECTION *H1
NO
YES
BUS1
G2
IS DEVICE A 1403 P OR 1404 P
NO
YES
BUS2
H2
TURN ON ALLOW IGNORE BIT IN ERQUE FLAG BYTE
1
MH F5
OVRUN
B3
IS DEVICE A 1443 P OR 1445 P
YES
NO
C3
INSERT X'14' OVERRUN MESSAGE CODE
MH B4
2
ATYPE
D3
TURN ON INTERVENTION REQUIRED BIT IN ERQUE FLAG
1
5
CHKCNT
F3
PUB ERROR COUNT = 1
NO
YES
1
H3
IS DEVICE A 1442 R/P OR 1442 P
YES
2
NO
J3
IS DEVICE A 2540 P
YES
5
NO
3
CMR
B4
COMMAND REJECT
YES
NO
6
MH J2
RTYCT
E4
ADD 1 TO ERROR COUNT IN PUB
RETRY
F4
SVC 3 EXRTY CHART DD
6
MH J4
CONT
G4
SVC 3 EXIGN CHART DD
MH J3
MH G5
CMDREJ
C5
IS DEVICE A 1403 P OR 1404 P
YES
NO
D5
IS DEVICE A 1443 P OR 1445 P
NO
YES
UCSTST
E5
GET ADDRESS OF LAST USED CCW
F5
WAS COMMAND CODE SET MODE
YES
NO
4
GETMSG
H5
INSERT X'18' COMMAND REJECT MESSAGE CODE
J5
TURN OFF ALLOW RETRY AND IGNORE BITS IN ERQUE FLAG BYTE
1
MSGWTR
K5
SVC 3 $$ANERRM CHART LV

Chart MK. $$ANERRW - MICR (1419D) ERP (Part 1 of 3)
Refer to Chart 08.


1
A1
$$ANERRW
B1
SSM TO ENABLE ALL INTERRUPTS
C1
LOAD ADDRESS OF COMM RGN, ERBLOC, AND PUB FOR DEVICE
D1
MOVE X'00' INTO A-TRANS PHASE NAME FIELD OF ERBLOC
E1
ERROR ON SECONDARY CONTROL UNIT
YES
NO
ML B1
SCURTN
F1
LOAD RETURN ADDR 'PCURTN' INTO RETURN REG
G1
CCB AVAILABLE
NO
YES
H1
PLCCB MM
GET CCB FOR POCKET LIGHT
PCURTN
J1
INSERT X'28' CHAN DATA CHECK MSG CODE
1
A2
CHAN DATA CHECK
NO
YES
4
ML-H4
ML-J3
MM-E1
PSTERR
D2
TURN ON UNRECOV I/O ERROR IN CCB BYTE 2, BIT 2
3
E2
TURN ON PASS BACK BIT 2 IN ERBLOC ERROR FLAG
MSGWTR
F2
SVC 3 FETCH $$ANERRM CHART LV
H2
SSCCB MM
GET CCB FOR STACKER SELECT
A3
SHOULD-NOT-OCCUR SENSE BITS 3, 7
NO
YES
2
MM D1
UNDTR
C3
INSERT X'19' UNDETERMINED ERROR MESSAGE CODE
IREXIT
J3
TURN ON INTERV REQ'D BIT IN CCB BYTE 3 BIT 1 AND IN ERBLOC ERROR FLG,BIT 1
3
A4
INSERT X'08' INTERVENTION REQUIRED MSG CODE
B4
INTERVENTION REQUIRED
YES
NO
BUSOUT
C4
INSERT X'09' BUSOUT CHECK MSG CODE
D4
BUSOUT CHECK
YES
NO
E4
INSERT X'37' DISENGAGE FAILED MSG CODE
F4
COMMAND REJECT
NO
YES
2
G4
GET ADDR OF CCW CAUSING INTERRUPT
H4
FAILED TO DISENGAGE *G3
YES
NO
J4
INSERT X'18' COMMAND REJECT MSG CODE
4
A5
TURN ON INT REQ BIT IN CCB, TURN ON ERBLOC FLG FOR INT REQ AND PASSBACK
B5
SSM TO DISABLE I/O INTERRUPTS
C5
TURN ON OPERATOR INTERVENTION BIT IN PUB CHAN SCHED FLG
D5
TURN ON CHAN END BIT IN CSW STATUS BITS
E5
DEVICE BUSY
YES
NO
F5
TURN OFF OPERATOR INTERVENTION BIT IN PUB CHAN SCHED FLG
G5
MOVE X'0C' TO CSW STATUS BYTE FOR CHAN END AND DEVICE END
CHKADR
H5
CSW HAS A COMMAND ADDRESS
YES
NO
J5
INSERT DUMMY ADDRESS X'001000' IN CSW
IGEXIT
K5
SVC 3 EXIGN CHART DD

*G3 THE CCW CAUSING INTERRUPT IS CHECKED WITH X'E1', A DUMMY COMMAND CODE FOR A FAILURE TO DISENGAGE.

Chart ML. $$ANERRW - MICR (1419D) ERP (Part 2 of 3)
Refer to Chart 08.

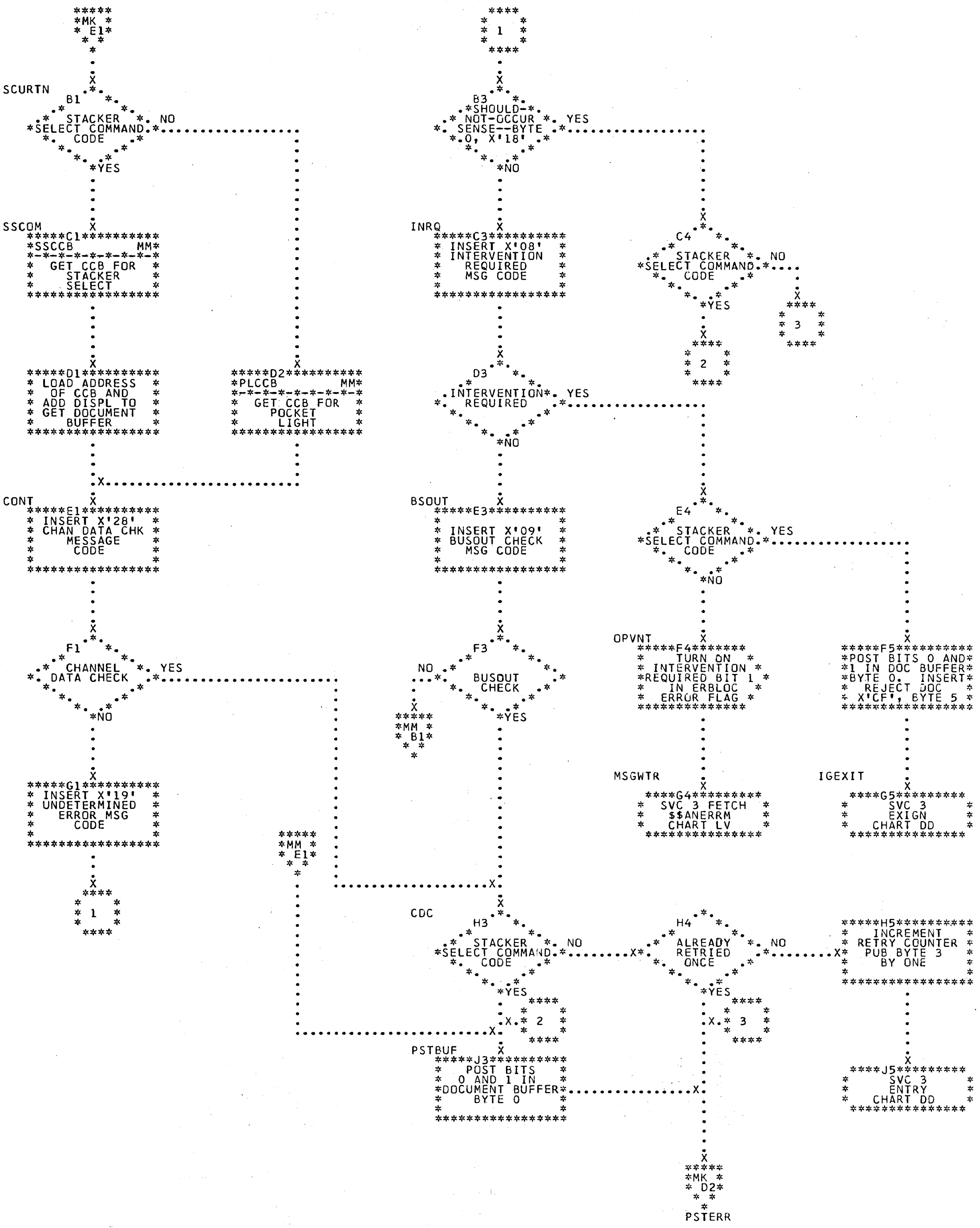

MK E1
SCURTN
B1
STACKER SELECT COMMAND CODE
NO
YES
SSCOM
C1
SSCCB MM
GET CCB FOR STACKER SELECT
D1
LOAD ADDRESS OF CCB AND ADD DISPL TO GET DOCUMENT BUFFER
D2
PLCCB MM
GET CCB FOR POCKET LIGHT
CONT
E1
INSERT X'28' CHAN DATA CHK MESSAGE CODE
F1
CHANNEL DATA CHECK
YES
NO
G1
INSERT X'19' UNDETERMINED ERROR MSG CODE
1
MM E1
1
B3
SHOULD-NOT-OCCUR SENSE--BYTE 0, X'18'
YES
NO
INRQ
C3
INSERT X'08' INTERVENTION REQUIRED MSG CODE
D3
INTERVENTION REQUIRED
YES
NO
BSOUT
E3
INSERT X'09' BUSOUT CHECK MSG CODE
F3
BUSOUT CHECK
NO
YES
MM B1
CDC
H3
STACKER SELECT COMMAND CODE
NO
YES
2
PSTBUF
J3
POST BITS 0 AND 1 IN DOCUMENT BUFFER BYTE 0
C4
STACKER SELECT COMMAND CODE
NO
YES
3
2
E4
STACKER SELECT COMMAND CODE
YES
NO
OPVNT
F4
TURN ON INTERVENTION REQUIRED BIT 1 IN ERBLOC ERROR FLAG
MSGWTR
G4
SVC 3 FETCH $$ANERRM CHART LV
H4
ALREADY RETRIED ONCE
NO
YES
3
MK D2
PSTERR
F5
POST BITS 0 AND 1 IN DOC BUFFER BYTE 0. INSERT REJECT DOC X'CF', BYTE 5
IGEXIT
G5
SVC 3 EXIGN CHART DD
H5
INCREMENT RETRY COUNTER PUB BYTE 3 BY ONE
J5
SVC 3 ENTRY CHART DD

Chart MM. $$ANERRW - MICR (1419D) ERP (Part 3 of 3)
Refer to Chart 08.

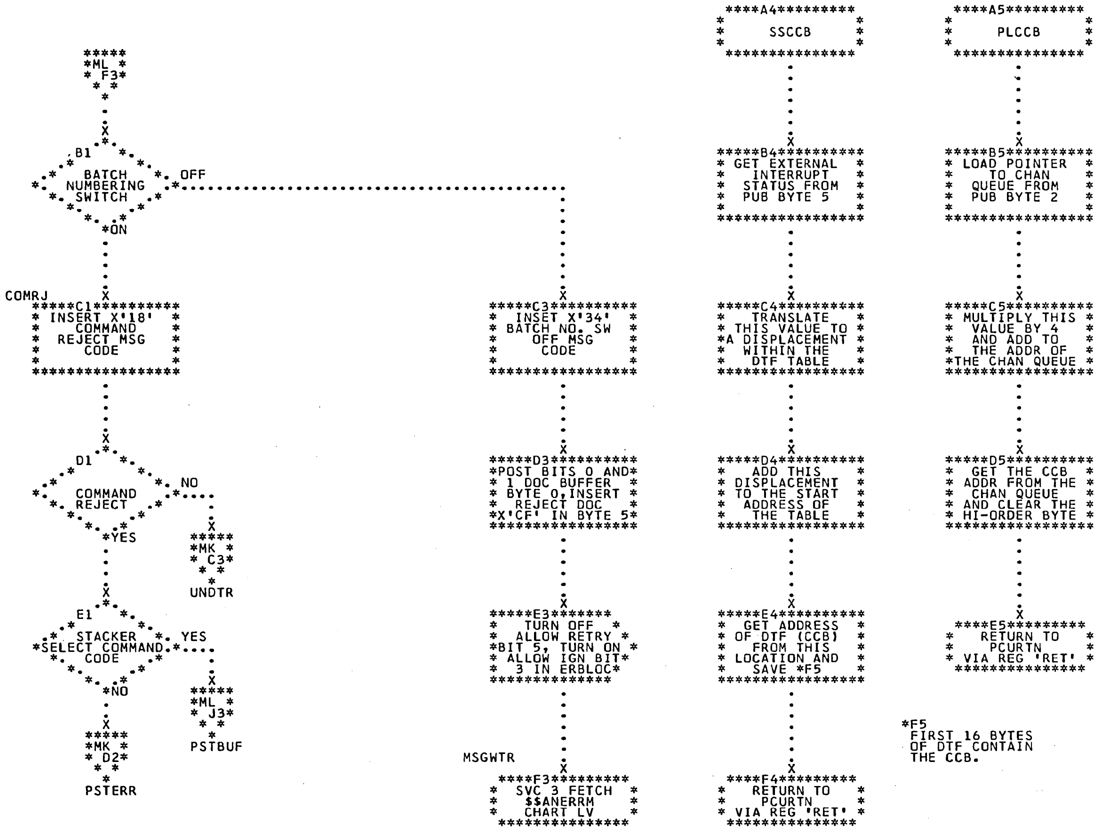

ML F3
B1 BATCH NUMBERING SWITCH
OFF
ON
COMRJ
C1 INSERT X'18' COMMAND REJECT MSG CODE
D1 COMMAND REJECT
NO
YES
MK C3
UNDTR
E1 STACKER SELECT COMMAND CODE
YES
NO
ML J3
PSTBUF
MK D2
PSTERR
C3 INSET X'34' BATCH NO. SW OFF MSG CODE
D3 POST BITS 0 AND 1 DOC BUFFER BYTE 0,INSERT REJECT DOC X'CF' IN BYTE 5
E3 TURN OFF ALLOW RETRY BIT 5, TURN ON ALLOW IGN BIT 3 IN ERBLOC
MSGWTR
F3 SVC 3 FETCH $$ANERRM CHART LV
A4 SSCCB
B4 GET EXTERNAL INTERRUPT STATUS FROM PUB BYTE 5
C4 TRANSLATE THIS VALUE TO A DISPLACEMENT WITHIN THE DTF TABLE
D4 ADD THIS DISPLACEMENT TO THE START ADDRESS OF THE TABLE
E4 GET ADDRESS OF DTF (CCB) FROM THIS LOCATION AND SAVE *F5
F4 RETURN TO PCURTN VIA REG 'RET'
A5 PLCCB
B5 LOAD POINTER TO CHAN QUEUE FROM PUB BYTE 2
C5 MULTIPLY THIS VALUE BY 4 AND ADD TO THE ADDR OF THE CHAN QUEUE
D5 GET THE CCB ADDR FROM THE CHAN QUEUE AND CLEAR THE HI-ORDER BYTE
E5 RETURN TO PCURTN VIA REG 'RET'
*F5
FIRST 16 BYTES OF DTF CONTAIN THE CCB.

Chart MN. $$ANERRX - Paper Tape ERP (Part 1 of 2)
Refer to Chart 07.

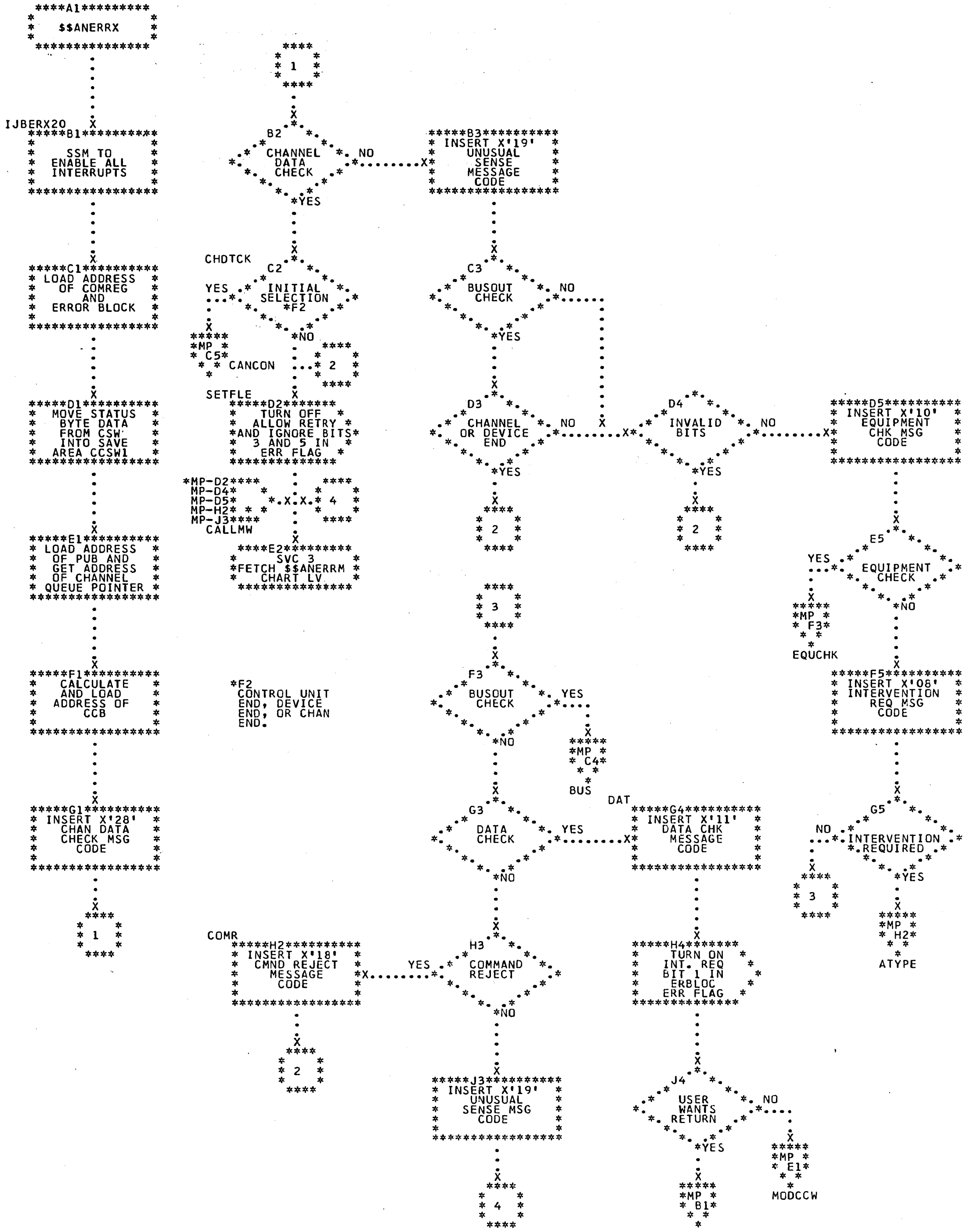

A1
$$ANERRX
IJBERX20
B1
SSM TO ENABLE ALL INTERRUPTS
C1
LOAD ADDRESS OF COMREG AND ERROR BLOCK
D1
MOVE STATUS BYTE DATA FROM CSW INTO SAVE AREA CCSW1
E1
LOAD ADDRESS OF PUB AND GET ADDRESS OF CHANNEL QUEUE POINTER
F1
CALCULATE AND LOAD ADDRESS OF CCB
G1
INSERT X'28' CHAN DATA CHECK MSG CODE
1
1
B2
CHANNEL DATA CHECK
NO
YES
CHDTCK
C2
INITIAL SELECTION *F2
YES
NO
MP C5
CANCON
2
SETFLE
D2
TURN OFF ALLOW RETRY AND IGNORE BITS 3 AND 5 IN ERR FLAG
MP-D2
MP-D4
MP-D5
MP-H2
MP-J3
CALLMW
4
E2
SVC 3 FETCH $$ANERRM CHART LV
*F2
CONTROL UNIT END, DEVICE END, OR CHAN END.
COMR
H2
INSERT X'18' CMND REJECT MESSAGE CODE
2
B3
INSERT X'19' UNUSUAL SENSE MESSAGE CODE
C3
BUSOUT CHECK
NO
YES
D3
CHANNEL OR DEVICE END
NO
YES
2
3
F3
BUSOUT CHECK
YES
NO
MP C4
BUS
G3
DATA CHECK
YES
NO
H3
COMMAND REJECT
YES
NO
J3
INSERT X'19' UNUSUAL SENSE MSG CODE
4
D4
INVALID BITS
NO
YES
2
DAT
G4
INSERT X'11' DATA CHK MESSAGE CODE
H4
TURN ON INT. REQ BIT 1 IN ERBLOC ERR FLAG
J4
USER WANTS RETURN
NO
YES
MP B1
MP E1
MODCCW
D5
INSERT X'10' EQUIPMENT CHK MSG CODE
E5
EQUIPMENT CHECK
YES
NO
MP F3
EQUCHK
F5
INSERT X'08' INTERVENTION REQ MSG CODE
G5
INTERVENTION REQUIRED
NO
YES
3
MP H2
ATYPE

Chart MP. $$ANERRX - Paper Tape ERP (Part 2 of 2)
Refer to Chart 07.

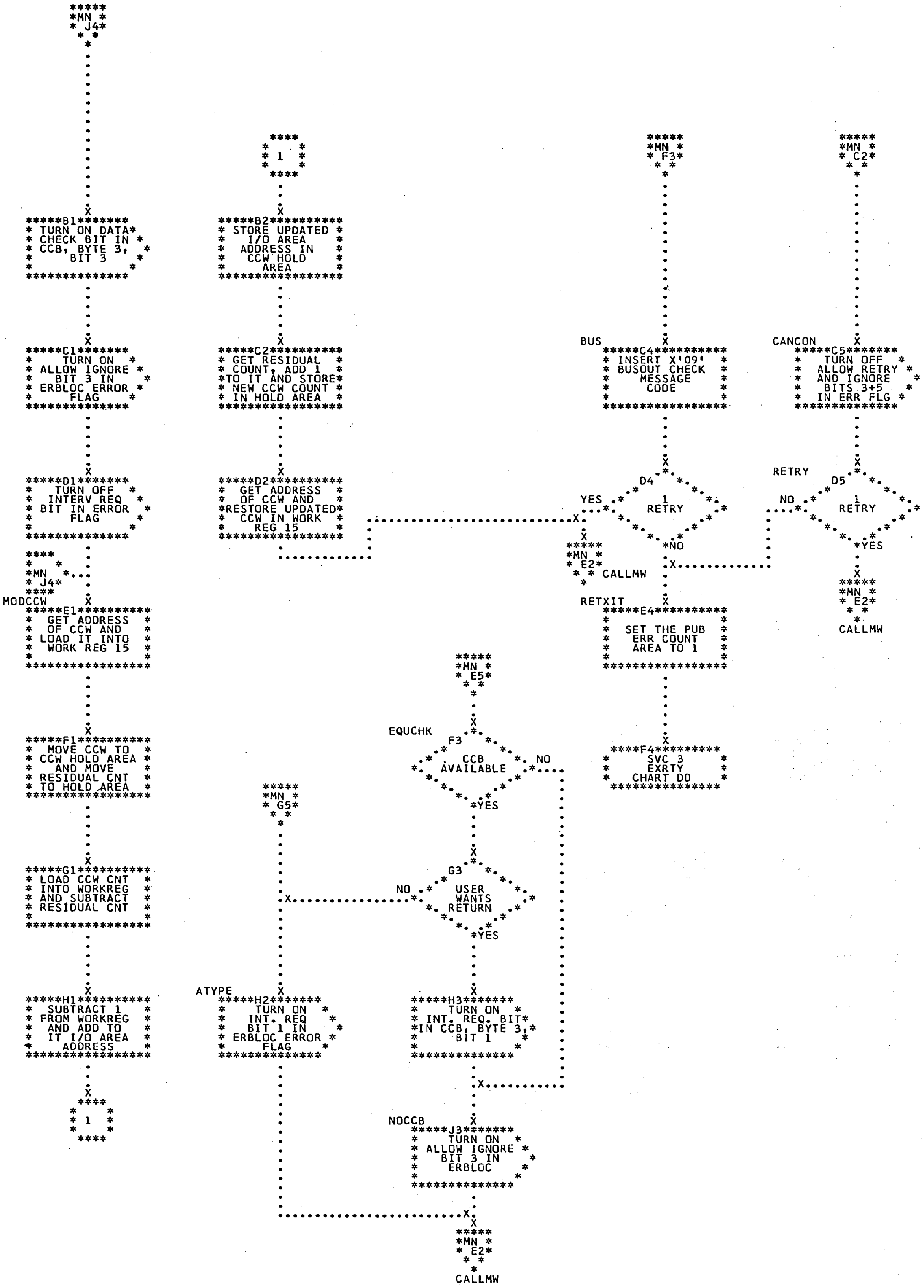
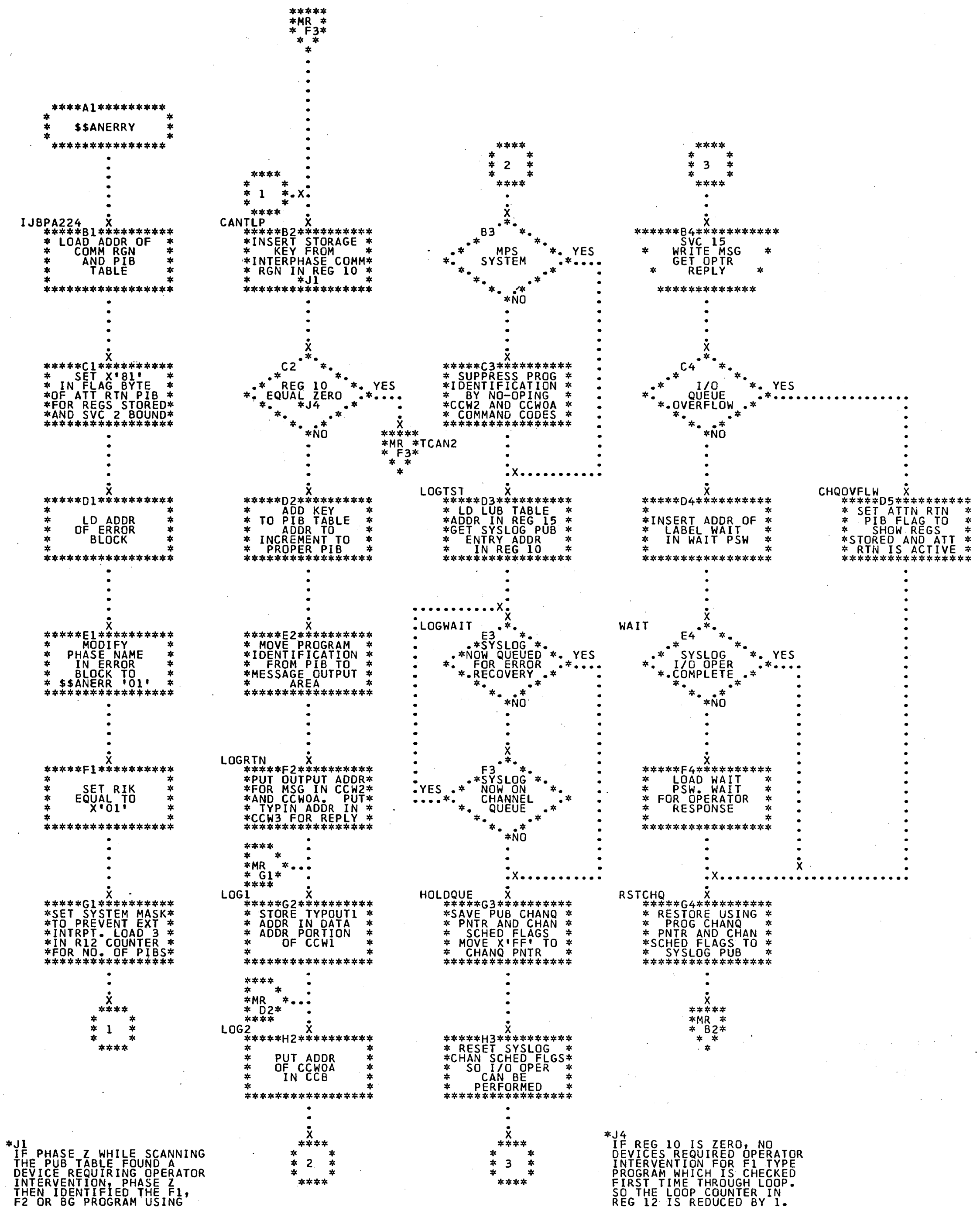

MN J4
B1 TURN ON DATA CHECK BIT IN CCB, BYTE 3, BIT 3
C1 TURN ON ALLOW IGNORE BIT 3 IN ERBLOC ERROR FLAG
D1 TURN OFF INTERV REQ BIT IN ERROR FLAG
MN J4
MODCCW
E1 GET ADDRESS OF CCW AND LOAD IT INTO WORK REG 15
F1 MOVE CCW TO CCW HOLD AREA AND MOVE RESIDUAL CNT TO HOLD AREA
G1 LOAD CCW CNT INTO WORKREG AND SUBTRACT RESIDUAL CNT
H1 SUBTRACT 1 FROM WORKREG AND ADD TO IT I/O AREA ADDRESS
1
1
B2 STORE UPDATED I/O AREA ADDRESS IN CCW HOLD AREA
C2 GET RESIDUAL COUNT, ADD 1 TO IT AND STORE NEW CCW COUNT IN HOLD AREA
D2 GET ADDRESS OF CCW AND RESTORE UPDATED CCW IN WORK REG 15
MN G5
ATYPE
H2 TURN ON INT. REQ BIT 1 IN ERBLOC ERROR FLAG
MN E5
EQUCHK
F3 CCB AVAILABLE
NO
YES
G3 USER WANTS RETURN
NO
YES
H3 TURN ON INT. REQ. BIT IN CCB, BYTE 3, BIT 1
NOCCB
J3 TURN ON ALLOW IGNORE BIT 3 IN ERBLOC
MN E2
CALLMW
MN F3
BUS
C4 INSERT X'09' BUSOUT CHECK MESSAGE CODE
D4 1 RETRY
YES
NO
MN E2
CALLMW
RETXIT
E4 SET THE PUB ERR COUNT AREA TO 1
F4 SVC 3 EXRTY CHART DD
MN C2
CANCON
C5 TURN OFF ALLOW RETRY AND IGNORE BITS 3+5 IN ERR FLG
RETRY
D5 1 RETRY
NO
YES
MN E2
CALLMW

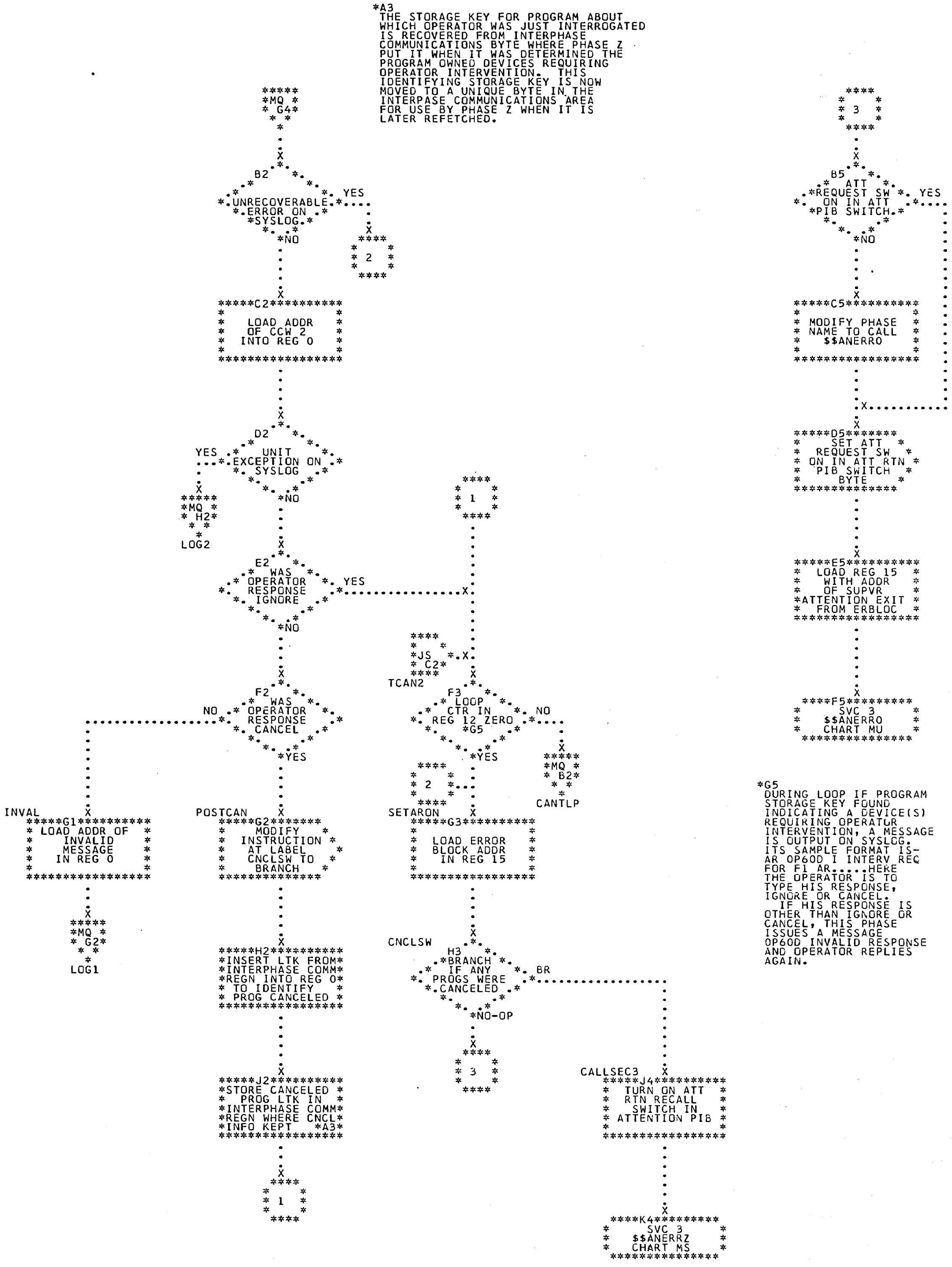
Chart MQ. $$ANERRY - Physical Attention Routine: Send Message
Refer to Chart 09.


MR F3
A1 $$ANERRY
IJBPA224
B1 LOAD ADDR OF COMM RGN AND PIB TABLE
C1 SET X'81' IN FLAG BYTE OF ATT RTN PIB FOR REGS STORED AND SVC 2 BOUND
D1 LD ADDR OF ERROR BLOCK
E1 MODIFY PHASE NAME IN ERROR BLOCK TO $$ANERR '01'
F1 SET RIK EQUAL TO X'01'
G1 SET SYSTEM MASK TO PREVENT EXT INTRPT. LOAD 3 IN R12 COUNTER FOR NO. OF PIBS
1
CANTLP
B2 INSERT STORAGE KEY FROM INTERPHASE COMM RGN IN REG 10 *J1
C2 REG 10 EQUAL ZERO *J4
YES
NO
MR F3 TCAN2
D2 ADD KEY TO PIB TABLE ADDR TO INCREMENT TO PROPER PIB
E2 MOVE PROGRAM IDENTIFICATION FROM PIB TO MESSAGE OUTPUT AREA
LOGRTN
F2 PUT OUTPUT ADDR FOR MSG IN CCW2 AND CCWOA. PUT TYPIN ADDR IN CCW3 FOR REPLY
MR G1
LOG1
G2 STORE TYPOUT1 ADDR IN DATA ADDR PORTION OF CCW1
MR D2
LOG2
H2 PUT ADDR OF CCWOA IN CCB
2
2
B3 MPS SYSTEM
YES
NO
C3 SUPPRESS PROG IDENTIFICATION BY NO-OPING CCW2 AND CCWOA COMMAND CODES
LOGTST
D3 LD LUB TABLE ADDR IN REG 15 GET SYSLOG PUB ENTRY ADDR IN REG 10
LOGWAIT
E3 SYSLOG NOW QUEUED FOR ERROR RECOVERY
YES
NO
F3 SYSLOG NOW ON CHANNEL QUEUE
YES
NO
HOLDQUE
G3 SAVE PUB CHANQ PNTR AND CHAN SCHED FLAGS MOVE X'FF' TO CHANQ PNTR
H3 RESET SYSLOG CHAN SCHED FLGS SO I/O OPER CAN BE PERFORMED
3
3
B4 SVC 15 WRITE MSG GET OPTR REPLY
C4 I/O QUEUE OVERFLOW
YES
NO
D4 INSERT ADDR OF LABEL WAIT IN WAIT PSW
WAIT
E4 SYSLOG I/O OPER COMPLETE
YES
NO
F4 LOAD WAIT PSW. WAIT FOR OPERATOR RESPONSE
RSTCHQ
G4 RESTORE USING PROG CHANQ PNTR AND CHAN SCHED FLAGS TO SYSLOG PUB
MR B2
CHQOVFLW
D5 SET ATTN RTN PIB FLAG TO SHOW REGS STORED AND ATT RTN IS ACTIVE

*J1
IF PHASE Z WHILE SCANNING THE PUB TABLE FOUND A DEVICE REQUIRING OPERATOR INTERVENTION, PHASE Z THEN IDENTIFIED THE F1, F2 OR BG PROGRAM USING THE DEVICE, STORED ITS STORAGE KEY IN THIS INTERPHASE COMMUNICATIONS AREA AND CALLED THIS PHASE.

*J4
IF REG 10 IS ZERO, NO DEVICES REQUIRED OPERATOR INTERVENTION FOR F1 TYPE PROGRAM WHICH IS CHECKED FIRST TIME THROUGH LOOP. SO THE LOOP COUNTER IN REG 12 IS REDUCED BY 1. REG 12 IS ALSO BEING USED TO INDEX THE INTERPHASE COMMUNICATIONS AREA. IT NOW POINTS AT THE NEXT LOWER STORAGE BYTE WHERE F2 CAN BE CHECKED AND FINALLY BG.

Chart MR. $$ANERRY - Physical Attention Routine: Read Operator Reply
Refer to Chart 09.


MQ G4
B2 UNRECOVERABLE ERROR ON SYSLOG
YES
2
NO
C2 LOAD ADDR OF CCW 2 INTO REG 0
D2 UNIT EXCEPTION ON SYSLOG
YES
MQ H2
LOG2
NO
1
E2 WAS OPERATOR RESPONSE IGNORE
YES
NO
JS C2
TCAN2
F2 WAS OPERATOR RESPONSE CANCEL
NO
YES
F3 LOOP CTR IN REG 12 ZERO *G5
NO
YES
MQ B2
CANTLP
2
INVAL
G1 LOAD ADDR OF INVALID MESSAGE IN REG 0
MQ G2
LOG1
POSTCAN
G2 MODIFY INSTRUCTION AT LABEL CNCLSW TO BRANCH
SETARON
G3 LOAD ERROR BLOCK ADDR IN REG 15
H2 INSERT LTK FROM INTERPHASE COMM REGN INTO REG 0 TO IDENTIFY PROG CANCELED
CNCLSW
H3 BRANCH IF ANY PROGS WERE CANCELED
BR
NO-OP
3
J2 STORE CANCELED PROG LTK IN INTERPHASE COMM REGN WHERE CNCL INFO KEPT *A3
1
CALLSEC3
J4 TURN ON ATT RTN RECALL SWITCH IN ATTENTION PIB
K4 SVC 3 $$ANERRZ CHART MS
3
B5 ATT REQUEST SW ON IN ATT PIB SWITCH
YES
NO
C5 MODIFY PHASE NAME TO CALL $$ANERRO
D5 SET ATT REQUEST SW ON IN ATT RTN PIB SWITCH BYTE
E5 LOAD REG 15 WITH ADDR OF SUPVR ATTENTION EXIT FROM ERBLOC
F5 SVC 3 $$ANERRO CHART MU

*A3
THE STORAGE KEY FOR PROGRAM ABOUT WHICH OPERATOR WAS JUST INTERROGATED IS RECOVERED FROM INTERPHASE COMMUNICATIONS BYTE WHERE PHASE Z PUT IT WHEN IT WAS DETERMINED THE PROGRAM OWNED DEVICES REQUIRING OPERATOR INTERVENTION. THIS IDENTIFYING STORAGE KEY IS NOW MOVED TO A UNIQUE BYTE IN THE INTERPASE COMMUNICATIONS AREA FOR USE BY PHASE Z WHEN IT IS LATER REFETCHED.

*G5
DURING LOOP IF PROGRAM STORAGE KEY FOUND INDICATING A DEVICE(S) REQUIRING OPERATOR INTERVENTION, A MESSAGE IS OUTPUT ON SYSLOG. ITS SAMPLE FORMAT IS- AR OP60D I INTERV REQ FOR F1 AR.....HERE THE OPERATOR IS TO TYPE HIS RESPONSE, IGNORE OR CANCEL.

IF HIS RESPONSE IS OTHER THAN IGNORE OR CANCEL, THIS PHASE ISSUES A MESSAGE OP60D INVALID RESPONSE AND OPERATOR REPLIES AGAIN.

Chart MS. $$ANERRZ - Physical Attention Routine: Initial PUB Scan
Refer to Chart 09.

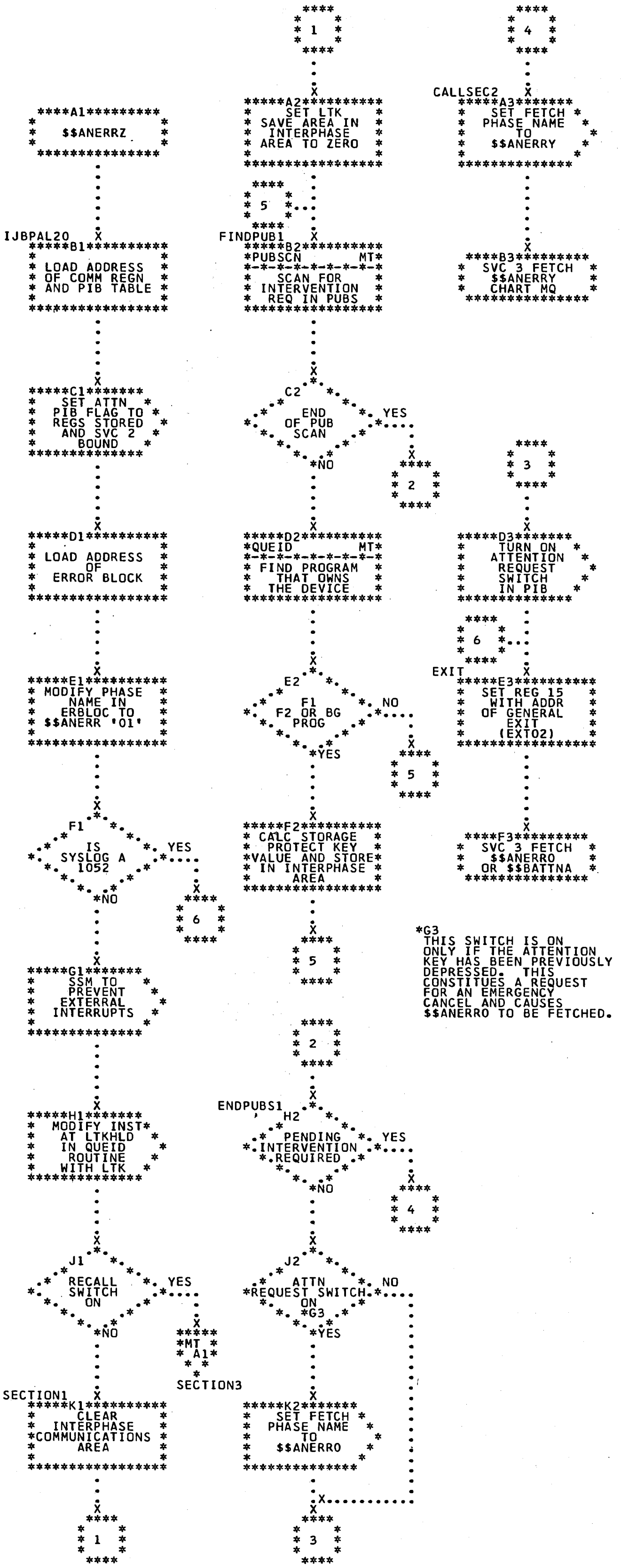
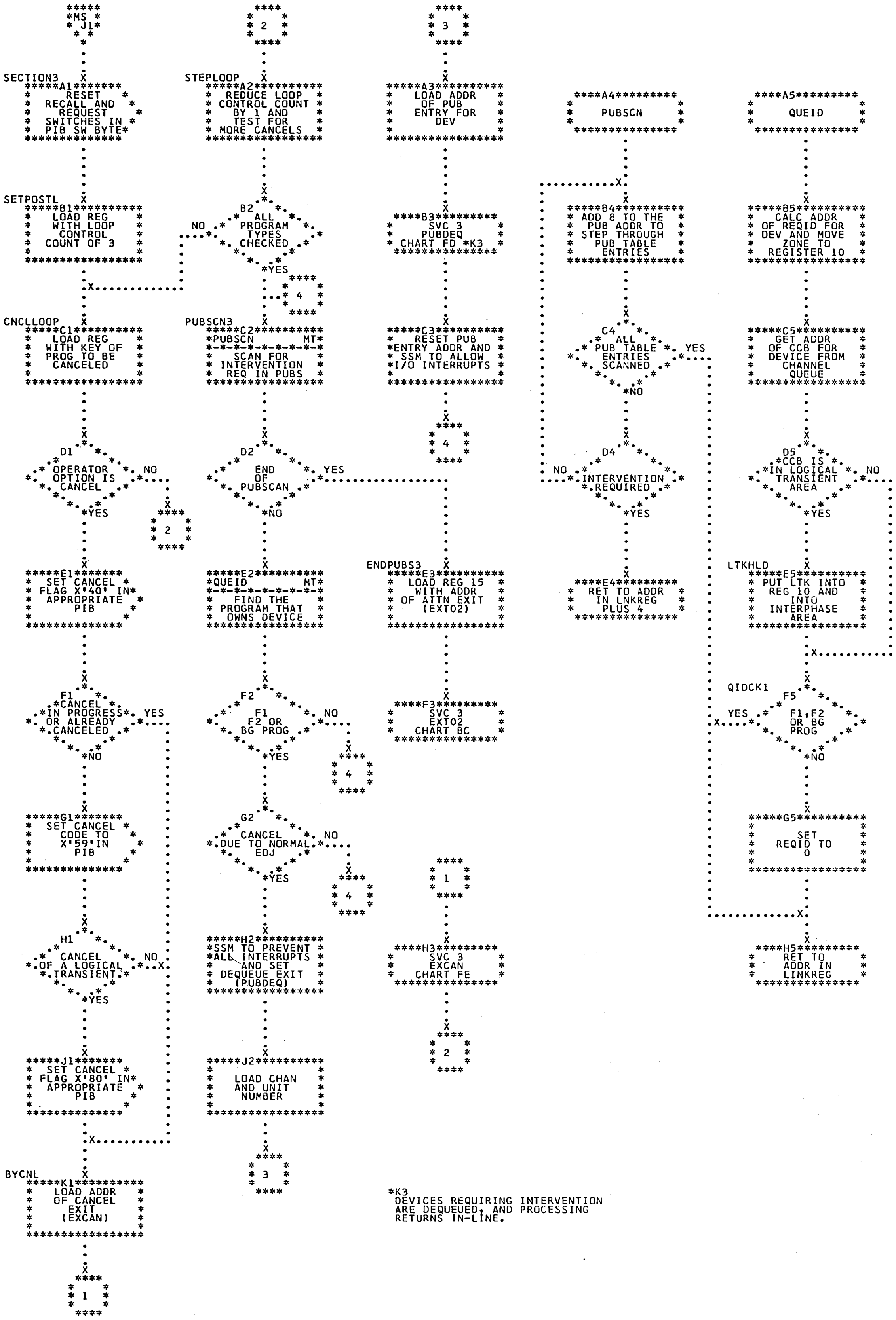
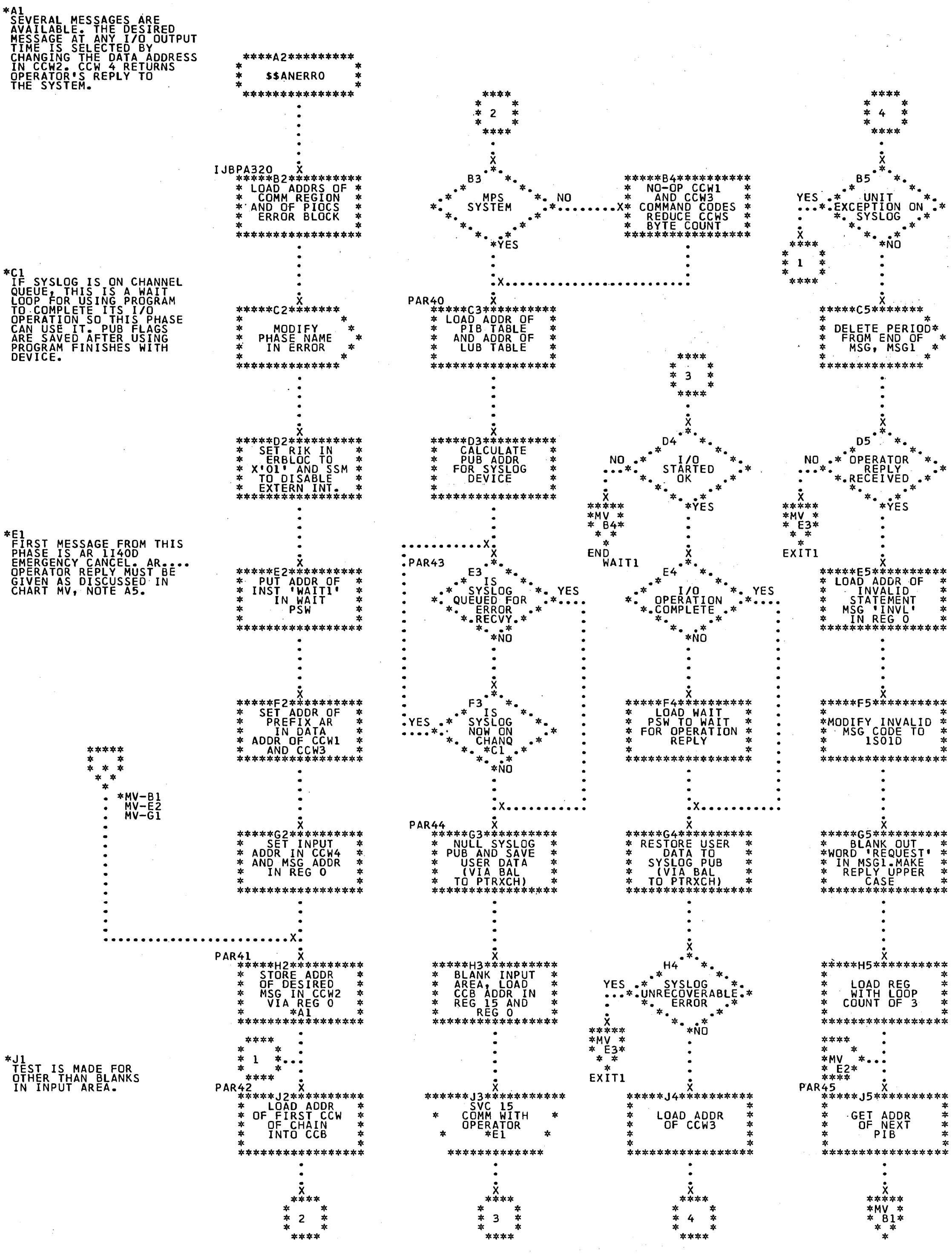

1
A1
$$ANERRZ
IJBPAL20
B1
LOAD ADDRESS OF COMM REGN AND PIB TABLE
C1
SET ATTN PIB FLAG TO REGS STORED AND SVC 2 BOUND
D1
LOAD ADDRESS OF ERROR BLOCK
E1
MODIFY PHASE NAME IN ERBLOC TO $$ANERR '01'
F1
IS SYSLOG A 1052
YES
NO
6
G1
SSM TO PREVENT EXTERRAL INTERRUPTS
H1
MODIFY INST AT LTKHLD IN QUEID ROUTINE WITH LTK
J1
RECALL SWITCH ON
YES
NO
MT
A1
SECTION3
SECTION1
K1
CLEAR INTERPHASE COMMUNICATIONS AREA
1
A2
SET LTK SAVE AREA IN INTERPHASE AREA TO ZERO
5
FINDPUB1
B2
PUBSCN MT
SCAN FOR INTERVENTION REQ IN PUBS
C2
END OF PUB SCAN
YES
NO
2
D2
QUEID MT
FIND PROGRAM THAT OWNS THE DEVICE
E2
F1 F2 OR BG PROG
NO
YES
5
F2
CALC STORAGE PROTECT KEY VALUE AND STORE IN INTERPHASE AREA
5
2
ENDPUBS1
H2
PENDING INTERVENTION REQUIRED
YES
NO
4
J2
ATTN REQUEST SWITCH ON
*G3
NO
YES
K2
SET FETCH PHASE NAME TO $$ANERRO
3
4
CALLSEC2
A3
SET FETCH PHASE NAME TO $$ANERRY
B3
SVC 3 FETCH $$ANERRY CHART MQ
3
D3
TURN ON ATTENTION REQUEST SWITCH IN PIB
6
EXIT
E3
SET REG 15 WITH ADDR OF GENERAL EXIT (EXTO2)
F3
SVC 3 FETCH $$ANERRO OR $$BATTNA
*G3
THIS SWITCH IS ON ONLY IF THE ATTENTION KEY HAS BEEN PREVIOUSLY DEPRESSED. THIS CONSTITUES A REQUEST FOR AN EMERGENCY CANCEL AND CAUSES $$ANERRO TO BE FETCHED.

Chart MT. $$ANERRZ - Physical Attention Routine: Cancel
Refer to Chart 09.


MS J1
2
3
SECTION3 A1 RESET RECALL AND REQUEST SWITCHES IN PIB SW BYTE
STEPLOOP A2 REDUCE LOOP CONTROL COUNT BY 1 AND TEST FOR MORE CANCELS
A3 LOAD ADDR OF PUB ENTRY FOR DEV
A4 PUBSCN
A5 QUEID
SETPOSTL B1 LOAD REG WITH LOOP CONTROL COUNT OF 3
B2 ALL PROGRAM TYPES CHECKED
NO
YES
4
B3 SVC 3 PUBDEQ CHART FD *K3
B4 ADD 8 TO THE PUB ADDR TO STEP THROUGH PUB TABLE ENTRIES
B5 CALC ADDR OF REQID FOR DEV AND MOVE ZONE TO REGISTER 10
CNCLLOOP C1 LOAD REG WITH KEY OF PROG TO BE CANCELED
PUBSCN3 C2 PUBSCN MT SCAN FOR INTERVENTION REQ IN PUBS
C3 RESET PUB ENTRY ADDR AND SSM TO ALLOW I/O INTERRUPTS
C4 ALL PUB TABLE ENTRIES SCANNED
YES
NO
C5 GET ADDR OF CCB FOR DEVICE FROM CHANNEL QUEUE
4
D1 OPERATOR OPTION IS CANCEL
NO
YES
2
D2 END OF PUBSCAN
YES
NO
D4 INTERVENTION REQUIRED
NO
YES
D5 CCB IS IN LOGICAL TRANSIENT AREA
NO
YES
E1 SET CANCEL FLAG X'40' IN APPROPRIATE PIB
E2 QUEID MT FIND THE PROGRAM THAT OWNS DEVICE
ENDPUBS3 E3 LOAD REG 15 WITH ADDR OF ATTN EXIT (EXT02)
E4 RET TO ADDR IN LNKREG PLUS 4
LTKHLD E5 PUT LTK INTO REG 10 AND INTO INTERPHASE AREA
F1 CANCEL IN PROGRESS OR ALREADY CANCELED
YES
NO
F2 F1 F2 OR BG PROG
NO
YES
4
F3 SVC 3 EXT02 CHART BC
QIDCK1 F5 F1,F2 OR BG PROG
YES
NO
G1 SET CANCEL CODE TO X'59'IN PIB
G2 CANCEL DUE TO NORMAL EOJ
NO
YES
4
1
G5 SET REQID TO 0
H1 CANCEL OF A LOGICAL TRANSIENT
NO
YES
H2 SSM TO PREVENT ALL INTERRUPTS AND SET DEQUEUE EXIT (PUBDEQ)
H3 SVC 3 EXCAN CHART FE
H5 RET TO ADDR IN LINKREG
2
J1 SET CANCEL FLAG X'80' IN APPROPRIATE PIB
J2 LOAD CHAN AND UNIT NUMBER
3
BYCNL K1 LOAD ADDR OF CANCEL EXIT (EXCAN)
1

*K3
DEVICES REQUIRING INTERVENTION
ARE DEQUEUED, AND PROCESSING
RETURNS IN-LINE.

Chart MU. $$ANERR0 - Physical Attention Routine: Emergency Cancel (Part 1 of 2)
Refer to Chart 09.


A2
$$ANERRO
IJBPA320
B2
LOAD ADDRS OF COMM REGION AND OF PIOCS ERROR BLOCK
C2
MODIFY PHASE NAME IN ERROR
D2
SET RIK IN ERBLOC TO X'01' AND SSM TO DISABLE EXTERN INT.
E2
PUT ADDR OF INST 'WAIT1' IN WAIT PSW
F2
SET ADDR OF PREFIX AR IN DATA ADDR OF CCW1 AND CCW3
*MV-B1
MV-E2
MV-G1
G2
SET INPUT ADDR IN CCW4 AND MSG ADDR IN REG 0
PAR41
H2
STORE ADDR OF DESIRED MSG IN CCW2 VIA REG 0
*A1
1
PAR42
J2
LOAD ADDR OF FIRST CCW OF CHAIN INTO CCB
2
2
B3
MPS SYSTEM
NO
YES
PAR40
C3
LOAD ADDR OF PIB TABLE AND ADDR OF LUB TABLE
D3
CALCULATE PUB ADDR FOR SYSLOG DEVICE
PAR43
E3
IS SYSLOG QUEUED FOR ERROR RECVY.
YES
NO
F3
IS SYSLOG NOW ON CHANQ
*C1
YES
NO
PAR44
G3
NULL SYSLOG PUB AND SAVE USER DATA (VIA BAL TO PTRXCH)
H3
BLANK INPUT AREA, LOAD CCB ADDR IN REG 15 AND REG 0
J3
SVC 15
COMM WITH OPERATOR
*E1
3
B4
NO-OP CCW1 AND CCW3 COMMAND CODES REDUCE CCWS BYTE COUNT
3
D4
I/O STARTED OK
NO
YES
MV
B4
END
WAIT1
E4
I/O OPERATION COMPLETE
YES
NO
F4
LOAD WAIT PSW TO WAIT FOR OPERATION REPLY
G4
RESTORE USER DATA TO SYSLOG PUB (VIA BAL TO PTRXCH)
H4
SYSLOG UNRECOVERABLE ERROR
YES
NO
MV
E3
EXIT1
J4
LOAD ADDR OF CCW3
4
4
B5
UNIT EXCEPTION ON SYSLOG
YES
NO
1
C5
DELETE PERIOD FROM END OF MSG, MSG1
D5
OPERATOR REPLY RECEIVED
NO
YES
MV
E3
EXIT1
E5
LOAD ADDR OF INVALID STATEMENT MSG 'INVL' IN REG 0
F5
MODIFY INVALID MSG CODE TO 1S01D
G5
BLANK OUT WORD 'REQUEST' IN MSG1.MAKE REPLY UPPER CASE
H5
LOAD REG WITH LOOP COUNT OF 3
MV
E2
PAR45
J5
GET ADDR OF NEXT PIB
MV
B1

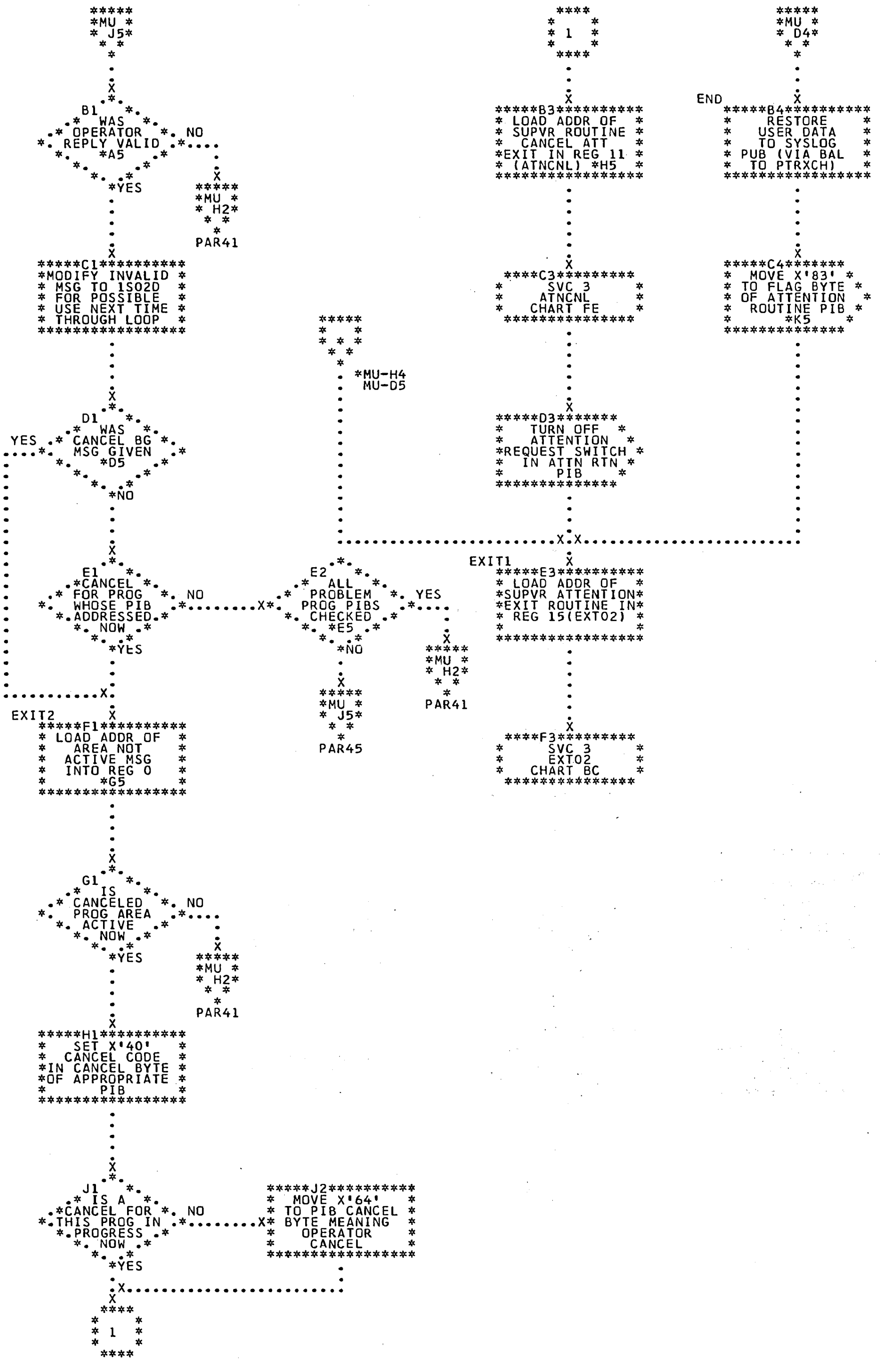
*A1
SEVERAL MESSAGES ARE AVAILABLE. THE DESIRED MESSAGE AT ANY I/O OUTPUT TIME IS SELECTED BY CHANGING THE DATA ADDRESS IN CCW2. CCW 4 RETURNS OPERATOR'S REPLY TO THE SYSTEM.

*C1
IF SYSLOG IS ON CHANNEL QUEUE, THIS IS A WAIT LOOP FOR USING PROGRAM TO COMPLETE ITS I/O OPERATION SO THIS PHASE CAN USE IT. PUB FLAGS ARE SAVED AFTER USING PROGRAM FINISHES WITH DEVICE.

*E1
FIRST MESSAGE FROM THIS PHASE IS AR 1I40D EMERGENCY CANCEL. AR.... OPERATOR REPLY MUST BE GIVEN AS DISCUSSED IN CHART MV, NOTE A5.

*J1
TEST IS MADE FOR OTHER THAN BLANKS IN INPUT AREA.

Chart MV. $$ANERR0 - Physical Attention Routine: Emergency Cancel (Part 2 of 2)
Refer to Chart 09.


MU J5
B1 WAS OPERATOR REPLY VALID *A5
NO
YES
MU H2
PAR41
C1 MODIFY INVALID MSG TO 1S02D FOR POSSIBLE USE NEXT TIME THROUGH LOOP
D1 WAS CANCEL BG MSG GIVEN *D5
YES
NO
E1 CANCEL FOR PROG WHOSE PIB ADDRESSED NOW
NO
YES
EXIT2
F1 LOAD ADDR OF AREA NOT ACTIVE MSG INTO REG 0 *G5
G1 IS CANCELED PROG AREA ACTIVE NOW
NO
YES
MU H2
PAR41
H1 SET X'40' CANCEL CODE IN CANCEL BYTE OF APPROPRIATE PIB
J1 IS A CANCEL FOR THIS PROG IN PROGRESS NOW
NO
YES
J2 MOVE X'64' TO PIB CANCEL BYTE MEANING OPERATOR CANCEL
1
*MU-H4
MU-D5
E2 ALL PROBLEM PROG PIBS CHECKED *E5
YES
NO
MU H2
PAR41
MU J5
PAR45
1
B3 LOAD ADDR OF SUPVR ROUTINE CANCEL ATT EXIT IN REG 11 (ATNCNL) *H5
C3 SVC 3 ATNCNL CHART FE
D3 TURN OFF ATTENTION REQUEST SWITCH IN ATTN RTN PIB
EXIT1
E3 LOAD ADDR OF SUPVR ATTENTION EXIT ROUTINE IN REG 15(EXT02)
F3 SVC 3 EXT02 CHART BC
MU D4
END
B4 RESTORE USER DATA TO SYSLOG PUB (VIA BAL TO PTRXCH)
C4 MOVE X'83' TO FLAG BYTE OF ATTENTION ROUTINE PIB *K5

*A5
OPERATOR RESPONSES CAN BE-
1. CANCEL F1
2. CANCEL F2
3. CANCEL BG
4. CANCEL
REPLY 4 MEANS CANCEL BG PROGRAMS. BEFORE TEST FOR VALID REPLY REG 0 WAS LOADED WITH ADDRESS OF INVALID STATEMENT MESSAGE. IF REPLY PROVES INVALID, THIS ADDRESS WILL BE PUT IN CCW2 DATA ADDRESS FOR SYSLOG OUTPUT TO OPERATOR SO HE CAN TRY AGAIN TO GIVE A VALID REPLY.

*D5
BLANKS IN PROGRAM IDENTIFICATION PART OF CANCEL MESSAGE MEANS CANCEL ALL PROGRAMS.

*E5
COUNTER IN REGISTER 14 SEES IF CANCEL MESSAGE WAS CHECKED FOR ALL 3 PROGRAM TYPES (F1, F2 AND BG). LOOP IS REPEATED UNTIL ALL TYPES ARE CHECKED.

*G5
IF PROGRAM AREA IS NOT NOW ACTIVE, SYSLOG IS USED TO OUTPUT '1I32D AREA NOT ACTIVE' MESSAGE.

*H5
A RETURN TO SUPERVISOR IS TAKEN TO PROCESS THE PROGRAM CANCEL AFTER WHICH CONTROL RETURNS TO NEXT SEQUENTIAL INSTRUCTION OF THIS A-TRANSIENT PHASE.

*K5
X'83' IN FLAG BYTE MEANS REGISTERS ARE STORED, ATTENTION ROUTINE IS ACTIVE AND NOT SVC 7 BOUND.ATTENTION ROUTINE WILL BE CALLED AGAIN.

Chart MW. $$ANERR1 - Move Data to Communications Region

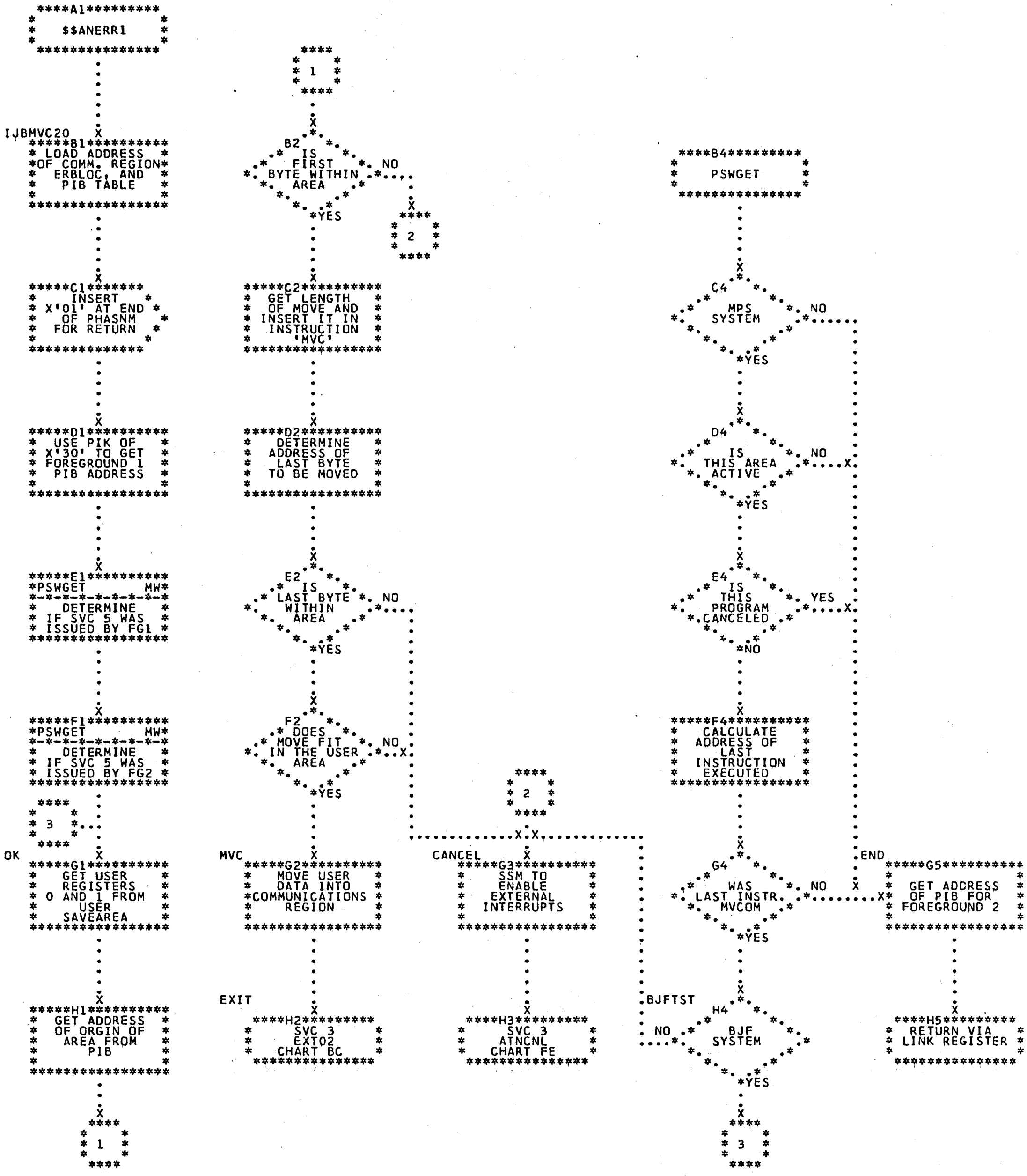

A1
$$ANERR1
IJBMVC20
B1
LOAD ADDRESS OF COMM. REGION, ERBLOC, AND PIB TABLE
C1
INSERT X'01' AT END OF PHASNM FOR RETURN
D1
USE PIK OF X'30' TO GET FOREGROUND 1 PIB ADDRESS
E1
PSWGET MW
DETERMINE IF SVC 5 WAS ISSUED BY FG1
F1
PSWGET MW
DETERMINE IF SVC 5 WAS ISSUED BY FG2
3
OK
G1
GET USER REGISTERS 0 AND 1 FROM USER SAVEAREA
H1
GET ADDRESS OF ORGIN OF AREA FROM PIB
1
1
B2
IS FIRST BYTE WITHIN AREA
NO
YES
2
C2
GET LENGTH OF MOVE AND INSERT IT IN INSTRUCTION 'MVC'
D2
DETERMINE ADDRESS OF LAST BYTE TO BE MOVED
E2
IS LAST BYTE WITHIN AREA
NO
YES
F2
DOES MOVE FIT IN THE USER AREA
NO
YES
MVC
G2
MOVE USER DATA INTO COMMUNICATIONS REGION
EXIT
H2
SVC 3 EXTO2 CHART BC
2
CANCEL
G3
SSM TO ENABLE EXTERNAL INTERRUPTS
H3
SVC 3 ATNCNL CHART FE
B4
PSWGET
C4
MPS SYSTEM
NO
YES
D4
IS THIS AREA ACTIVE
NO
YES
E4
IS THIS PROGRAM CANCELED
YES
NO
F4
CALCULATE ADDRESS OF LAST INSTRUCTION EXECUTED
G4
WAS LAST INSTR. MVCOM
NO
YES
BJFTST
H4
BJF SYSTEM
NO
YES
3
END
G5
GET ADDRESS OF PIB FOR FOREGROUND 2
H5
RETURN VIA LINK REGISTER

Chart MX. $$ANERR9 - Optical Reader ERP (Part 1 of 2)
Refer to Chart 07.

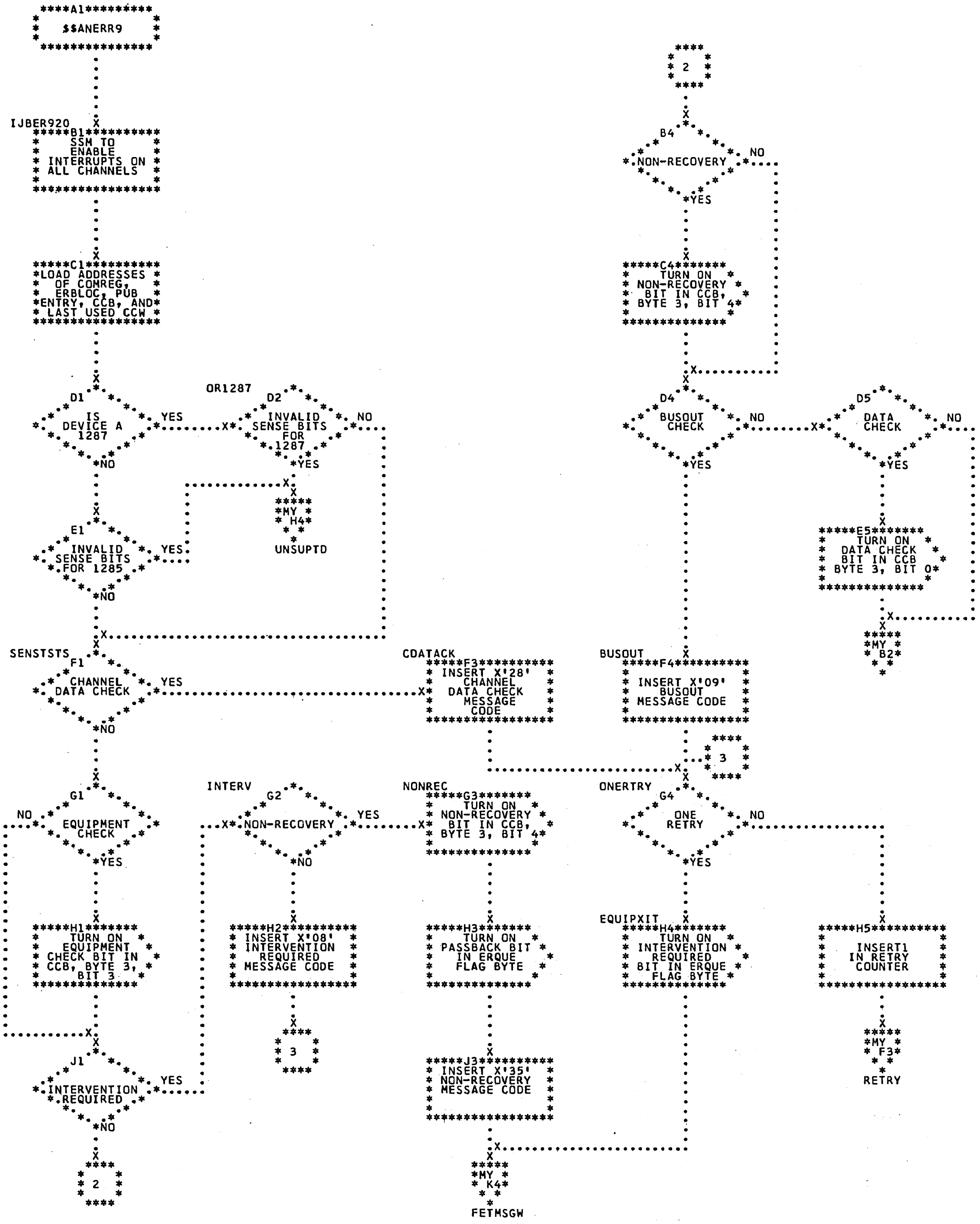

A1
$$ANERR9
IJBER920
B1
SSM TO ENABLE INTERRUPTS ON ALL CHANNELS
C1
LOAD ADDRESSES OF COMREG, ERBLOC, PUB ENTRY, CCB, AND LAST USED CCW
D1
IS DEVICE A 1287
YES
NO
OR1287
D2
INVALID SENSE BITS FOR 1287
NO
YES
MY H4
UNSUPTD
E1
INVALID SENSE BITS FOR 1285
YES
NO
SENSTSTS
F1
CHANNEL DATA CHECK
YES
NO
CDATACK
F3
INSERT X'28' CHANNEL DATA CHECK MESSAGE CODE
G1
EQUIPMENT CHECK
NO
YES
H1
TURN ON EQUIPMENT CHECK BIT IN CCB, BYTE 3, BIT 3
J1
INTERVENTION REQUIRED
YES
NO
2
INTERV
G2
NON-RECOVERY
YES
NO
H2
INSERT X'08' INTERVENTION REQUIRED MESSAGE CODE
3
NONREC
G3
TURN ON NON-RECOVERY BIT IN CCB, BYTE 3, BIT 4
H3
TURN ON PASSBACK BIT IN ERQUE FLAG BYTE
J3
INSERT X'35' NON-RECOVERY MESSAGE CODE
MY K4
FETMSGW
2
B4
NON-RECOVERY
NO
YES
C4
TURN ON NON-RECOVERY BIT IN CCB, BYTE 3, BIT 4
D4
BUSOUT CHECK
NO
YES
BUSOUT
F4
INSERT X'09' BUSOUT MESSAGE CODE
3
ONERTRY
G4
ONE RETRY
NO
YES
EQUIPXIT
H4
TURN ON INTERVENTION REQUIRED BIT IN ERQUE FLAG BYTE
D5
DATA CHECK
NO
YES
E5
TURN ON DATA CHECK BIT IN CCB BYTE 3, BIT 0
MY B2
H5
INSERT1 IN RETRY COUNTER
MY F3
RETRY

Chart MY. $$ANERR9 - Optical Reader ERP (Part 2 of 2)
Refer to Chart 07.

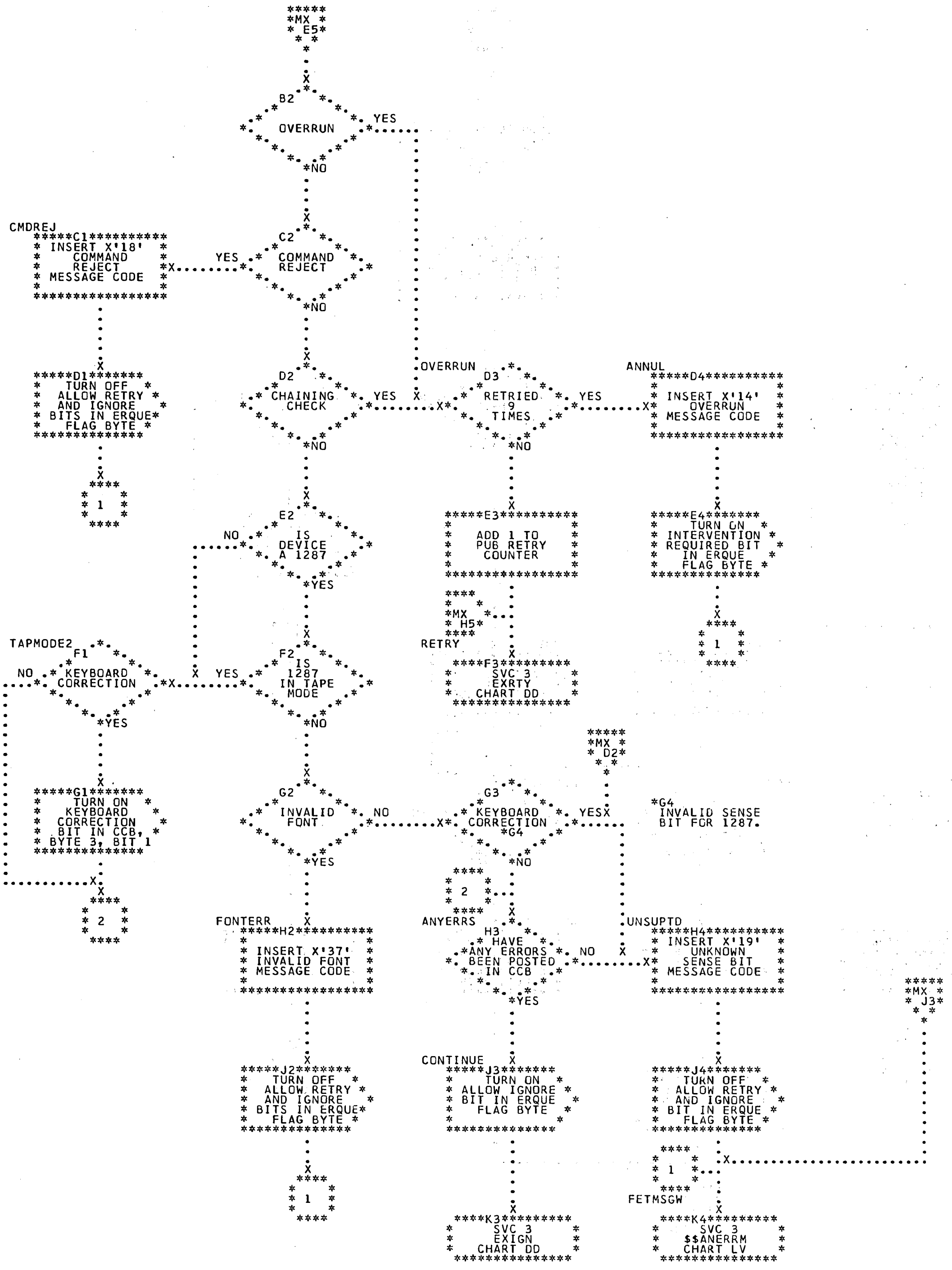
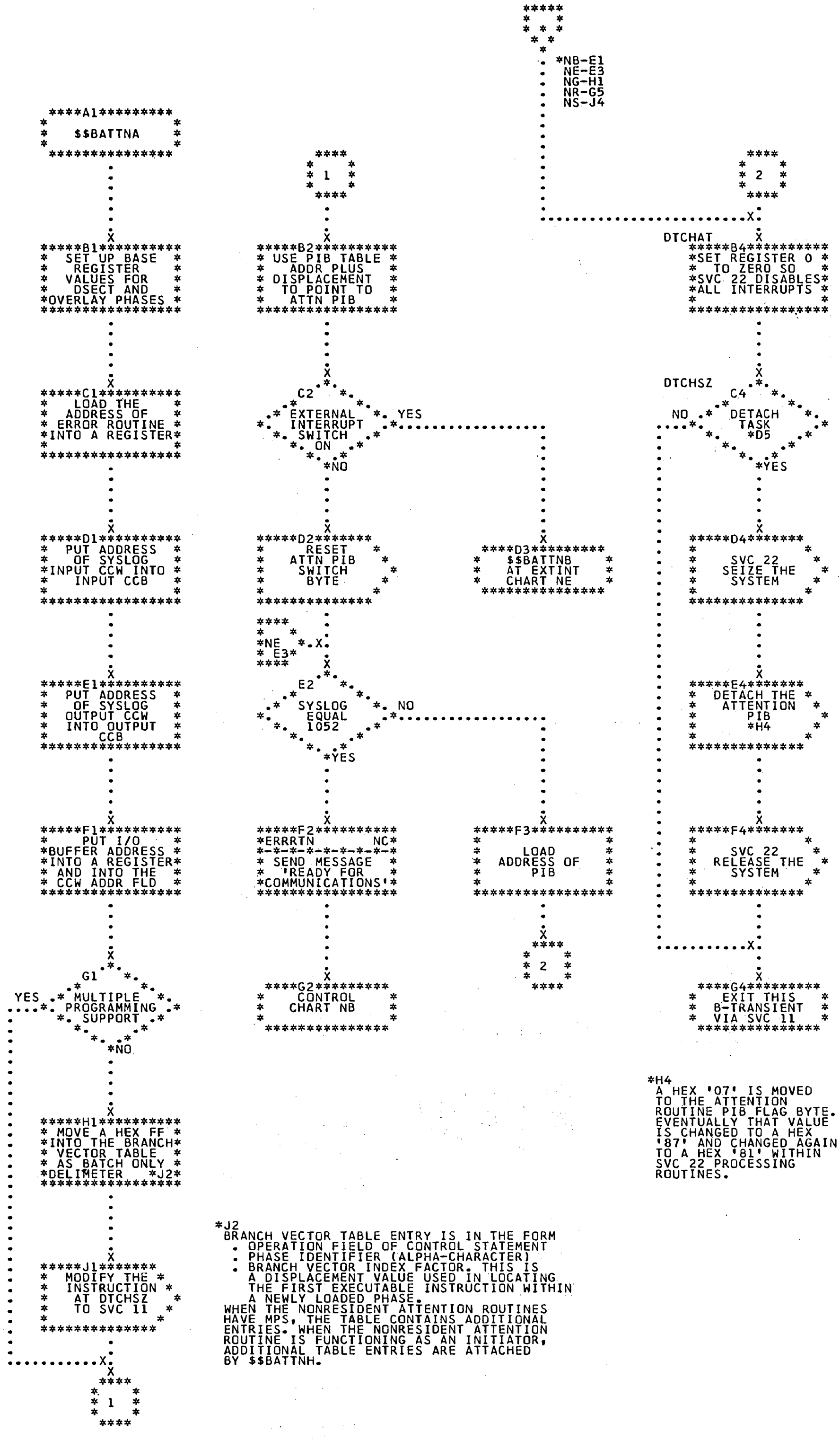

MX E5
B2
OVERRUN
YES
NO
CMDREJ
C1
INSERT X'18' COMMAND REJECT MESSAGE CODE
YES
C2
COMMAND REJECT
NO
D1
TURN OFF ALLOW RETRY AND IGNORE BITS IN ERQUE FLAG BYTE
1
D2
CHAINING CHECK
YES
NO
OVERRUN
D3
RETRIED 9 TIMES
YES
NO
ANNUL
D4
INSERT X'14' OVERRUN MESSAGE CODE
NO
E2
IS DEVICE A 1287
YES
E3
ADD 1 TO PUB RETRY COUNTER
E4
TURN ON INTERVENTION REQUIRED BIT IN ERQUE FLAG BYTE
1
MX H5
RETRY
F3
SVC 3 EXRTY CHART DD
TAPMODE2
F1
NO
KEYBOARD CORRECTION
YES
YES
F2
IS 1287 IN TAPE MODE
NO
G1
TURN ON KEYBOARD CORRECTION BIT IN CCB, BYTE 3, BIT 1
2
MX D2
G2
INVALID FONT
NO
YES
G3
KEYBOARD CORRECTION
*G4
YES
NO
*G4
INVALID SENSE BIT FOR 1287.
2
FONTERR
H2
INSERT X'37' INVALID FONT MESSAGE CODE
ANYERRS
H3
HAVE ANY ERRORS BEEN POSTED IN CCB
NO
YES
UNSUPTD
H4
INSERT X'19' UNKNOWN SENSE BIT MESSAGE CODE
MX J3
J2
TURN OFF ALLOW RETRY AND IGNORE BITS IN ERQUE FLAG BYTE
1
CONTINUE
J3
TURN ON ALLOW IGNORE BIT IN ERQUE FLAG BYTE
J4
TURN OFF ALLOW RETRY AND IGNORE BIT IN ERQUE FLAG BYTE
1
FETMSGW
K3
SVC 3 EXIGN CHART DD
K4
SVC 3 $$ANERRM CHART LV

Chart NA. $$BATTNA - Nonresident Attention/Initiator Root Phase
Refer to Chart 10.


*NB-E1
NE-E3
NG-H1
NR-G5
NS-J4
A1
$$BATTNA
1
2
B1
SET UP BASE REGISTER VALUES FOR DSECT AND OVERLAY PHASES
B2
USE PIB TABLE ADDR PLUS DISPLACEMENT TO POINT TO ATTN PIB
DTCHAT
B4
SET REGISTER 0 TO ZERO SO SVC 22 DISABLES ALL INTERRUPTS
C1
LOAD THE ADDRESS OF ERROR ROUTINE INTO A REGISTER
C2
EXTERNAL INTERRUPT SWITCH ON
YES
NO
DTCHSZ
C4
DETACH TASK *D5
NO
YES
D1
PUT ADDRESS OF SYSLOG INPUT CCW INTO INPUT CCB
D2
RESET ATTN PIB SWITCH BYTE
D3
$$BATTNB AT EXTINT CHART NE
D4
SVC 22 SEIZE THE SYSTEM
NE E3
E1
PUT ADDRESS OF SYSLOG OUTPUT CCW INTO OUTPUT CCB
E2
SYSLOG EQUAL 1052
NO
YES
E4
DETACH THE ATTENTION PIB *H4
F1
PUT I/O BUFFER ADDRESS INTO A REGISTER AND INTO THE CCW ADDR FLD
F2
ERRRTN NC
SEND MESSAGE 'READY FOR COMMUNICATIONS'
F3
LOAD ADDRESS OF PIB
F4
SVC 22 RELEASE THE SYSTEM
2
G1
MULTIPLE PROGRAMMING SUPPORT
YES
NO
G2
CONTROL CHART NB
G4
EXIT THIS B-TRANSIENT VIA SVC 11
H1
MOVE A HEX FF INTO THE BRANCH VECTOR TABLE AS BATCH ONLY DELIMETER *J2
J1
MODIFY THE INSTRUCTION AT DTCHSZ TO SVC 11
1

*D5
THIS INSTRUCTION WILL BE SET TO SVC 11 IF NO MPS.

*H4
A HEX '07' IS MOVED TO THE ATTENTION ROUTINE PIB FLAG BYTE. EVENTUALLY THAT VALUE IS CHANGED TO A HEX '87' AND CHANGED AGAIN TO A HEX '81' WITHIN SVC 22 PROCESSING ROUTINES.

*J2
BRANCH VECTOR TABLE ENTRY IS IN THE FORM

- OPERATION FIELD OF CONTROL STATEMENT
- PHASE IDENTIFIER (ALPHA-CHARACTER)
- BRANCH VECTOR INDEX FACTOR. THIS IS A DISPLACEMENT VALUE USED IN LOCATING THE FIRST EXECUTABLE INSTRUCTION WITHIN A NEWLY LOADED PHASE.

WHEN THE NONRESIDENT ATTENTION ROUTINES HAVE MPS, THE TABLE CONTAINS ADDITIONAL ENTRIES. WHEN THE NONRESIDENT ATTENTION ROUTINE IS FUNCTIONING AS AN INITIATOR, ADDITIONAL TABLE ENTRIES ARE ATTACHED BY $$BATTNH.

Chart NB. $$BATTNA - Control Routine
Refer to Chart 10.

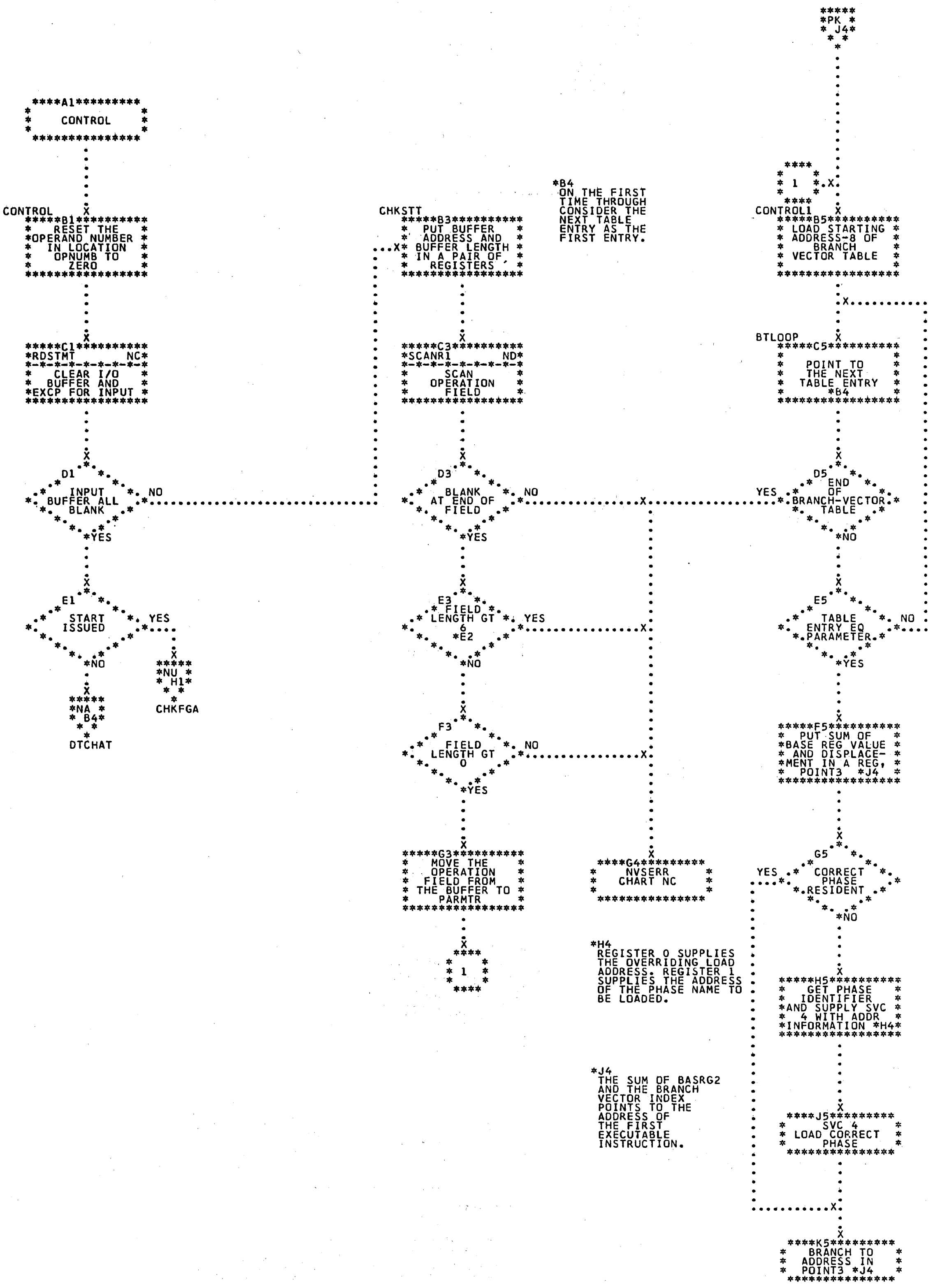
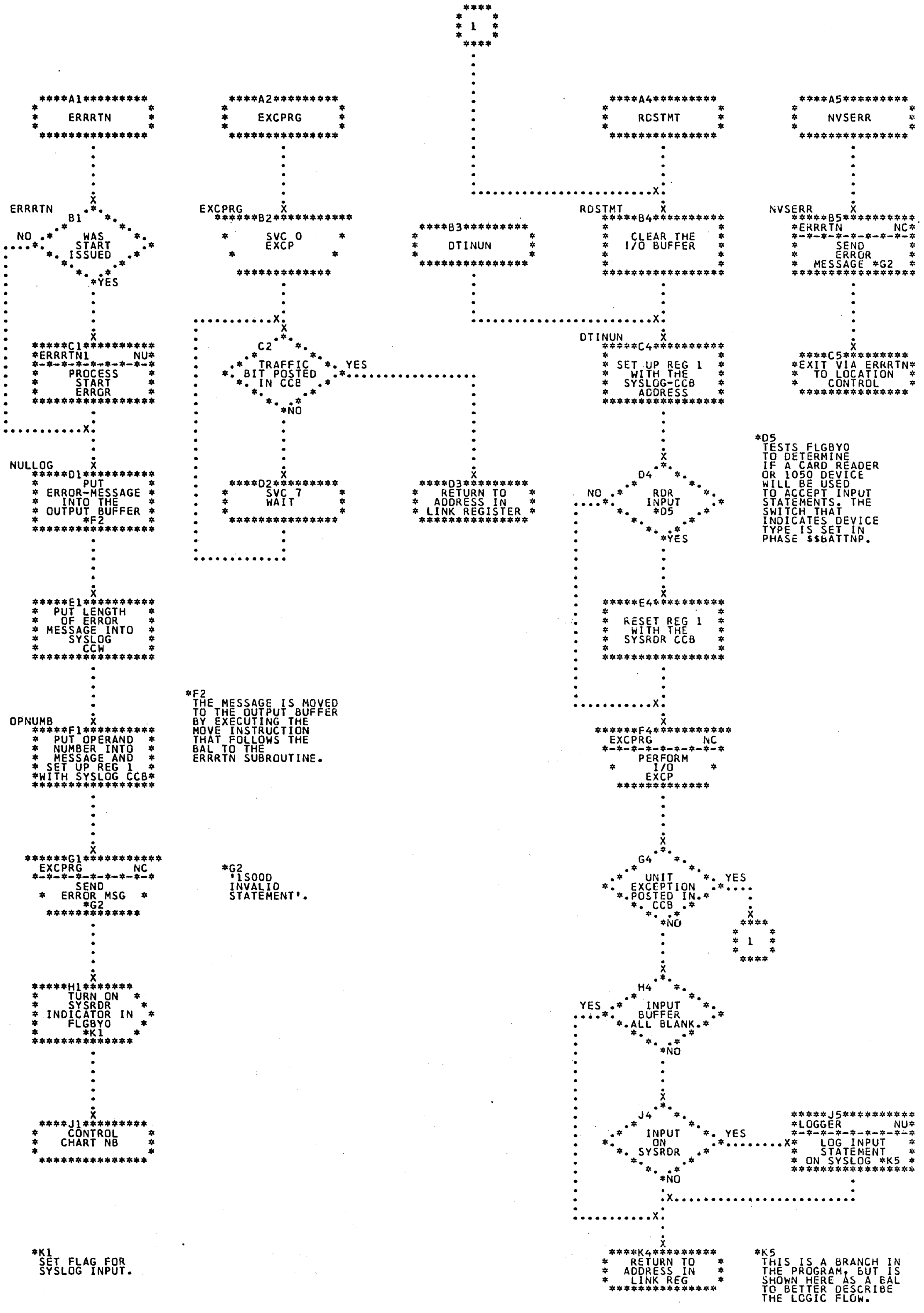

A1 CONTROL
CONTROL
B1 RESET THE OPERAND NUMBER IN LOCATION OPNUMB TO ZERO
C1 RDSTMT NC
CLEAR I/O BUFFER AND EXCP FOR INPUT
D1 INPUT BUFFER ALL BLANK
NO
YES
E1 START ISSUED
YES
NO
NA B4
DTCHAT
NU H1
CHKFGA
CHKSTT
B3 PUT BUFFER ADDRESS AND BUFFER LENGTH IN A PAIR OF REGISTERS
C3 SCANR1 ND
SCAN OPERATION FIELD
D3 BLANK AT END OF FIELD
NO
YES
E3 FIELD LENGTH GT 6 E2
YES
NO
F3 FIELD LENGTH GT 0
NO
YES
G3 MOVE THE OPERATION FIELD FROM THE BUFFER TO PARMTR
1
*B4
ON THE FIRST TIME THROUGH CONSIDER THE NEXT TABLE ENTRY AS THE FIRST ENTRY.
G4 NVSERR CHART NC
*H4
REGISTER 0 SUPPLIES THE OVERRIDING LOAD ADDRESS. REGISTER 1 SUPPLIES THE ADDRESS OF THE PHASE NAME TO BE LOADED.
*J4
THE SUM OF BASRG2 AND THE BRANCH VECTOR INDEX POINTS TO THE ADDRESS OF THE FIRST EXECUTABLE INSTRUCTION.
PK J4
1
CONTROL1
B5 LOAD STARTING ADDRESS-8 OF BRANCH VECTOR TABLE
BTLOOP
C5 POINT TO THE NEXT TABLE ENTRY *B4
D5 END OF BRANCH-VECTOR TABLE
YES
NO
E5 TABLE ENTRY EQ PARAMETER
NO
YES
F5 PUT SUM OF BASE REG VALUE AND DISPLACE-MENT IN A REG, POINT3 *J4
G5 CORRECT PHASE RESIDENT
YES
NO
H5 GET PHASE IDENTIFIER AND SUPPLY SVC 4 WITH ADDR INFORMATION *H4
J5 SVC 4 LOAD CORRECT PHASE
K5 BRANCH TO ADDRESS IN POINT3 *J4

Chart NC. $$BATTNA - Root Phase Subroutines
Refer to Chart 10.


1
A1 ERRRTN
A2 EXCPRG
A4 RDSTMT
A5 NVSERR
ERRRTN B1 WAS START ISSUED
NO
YES
EXCPRG B2 SVC 0 EXCP
B3 DTINUN
RDSTMT B4 CLEAR THE I/O BUFFER
NVSERR B5 ERRRTN NC SEND ERROR MESSAGE *G2
C1 ERRRTN1 NU PROCESS START ERROR
C2 TRAFFIC BIT POSTED IN CCB
YES
NO
DTINUN C4 SET UP REG 1 WITH THE SYSLOG-CCB ADDRESS
C5 EXIT VIA ERRRTN TO LOCATION CONTROL
NULLOG D1 PUT ERROR-MESSAGE INTO THE OUTPUT BUFFER *F2
D2 SVC 7 WAIT
D3 RETURN TO ADDRESS IN LINK REGISTER
D4 RDR INPUT *D5
NO
YES
E1 PUT LENGTH OF ERROR MESSAGE INTO SYSLOG CCW
E4 RESET REG 1 WITH THE SYSRDR CCB
OPNUMB F1 PUT OPERAND NUMBER INTO MESSAGE AND SET UP REG 1 WITH SYSLOG CCB
F4 EXCPRG NC PERFORM I/O EXCP
G1 EXCPRG NC SEND ERROR MSG *G2
G4 UNIT EXCEPTION POSTED IN CCB
YES
NO
1
H1 TURN ON SYSRDR INDICATOR IN FLGBYO *K1
H4 INPUT BUFFER ALL BLANK
YES
NO
J1 CONTROL CHART NB
J4 INPUT ON SYSRDR
YES
NO
J5 LOGGER NU LOG INPUT STATEMENT ON SYSLOG *K5
K4 RETURN TO ADDRESS IN LINK REG

*D5
TESTS FLGBYO TO DETERMINE IF A CARD READER OR 1050 DEVICE WILL BE USED TO ACCEPT INPUT STATEMENTS. THE SWITCH THAT INDICATES DEVICE TYPE IS SET IN PHASE $$BATTNP.

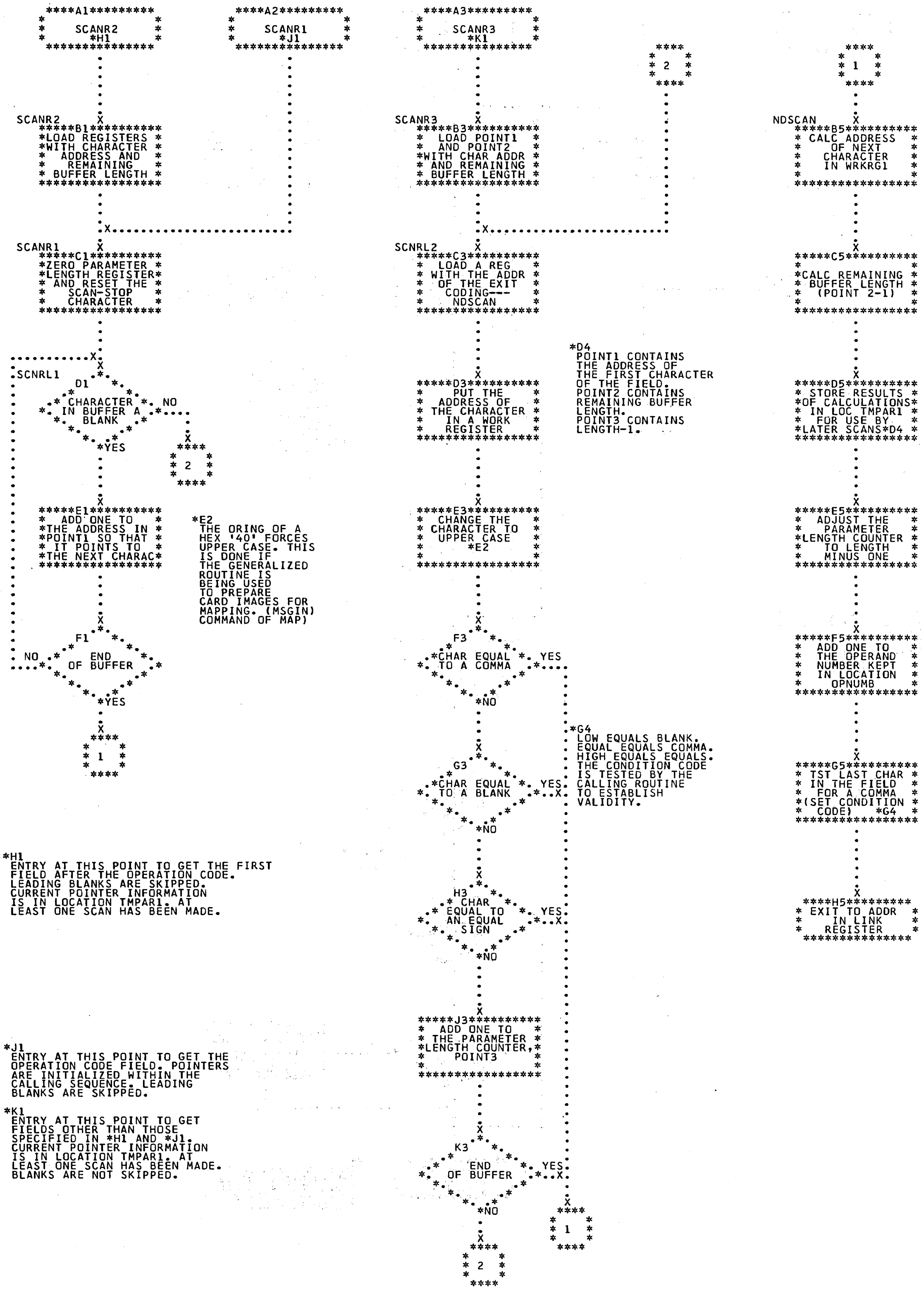
*F2
THE MESSAGE IS MOVED TO THE OUTPUT BUFFER BY EXECUTING THE MOVE INSTRUCTION THAT FOLLOWS THE BAL TO THE ERRRTN SUBROUTINE.

*G2
'1S00D INVALID STATEMENT'.

*K1
SET FLAG FOR SYSLOG INPUT.

*K5
THIS IS A BRANCH IN THE PROGRAM, BUT IS SHOWN HERE AS A BAL TO BETTER DESCRIBE THE LOGIC FLOW.

Chart ND. $$BATTNA - General Scan Routines
Refer to Chart 10.


A1 SCANR2 *H1
A2 SCANR1 *J1
A3 SCANR3 *K1
2
1
SCANR2 B1 LOAD REGISTERS WITH CHARACTER ADDRESS AND REMAINING BUFFER LENGTH
SCANR3 B3 LOAD POINT1 AND POINT2 WITH CHAR ADDR AND REMAINING BUFFER LENGTH
NDSCAN B5 CALC ADDRESS OF NEXT CHARACTER IN WRKRG1
SCANR1 C1 ZERO PARAMETER LENGTH REGISTER AND RESET THE SCAN-STOP CHARACTER
SCNRL2 C3 LOAD A REG WITH THE ADDR OF THE EXIT CODING--- NDSCAN
C5 CALC REMAINING BUFFER LENGTH (POINT 2-1)
SCNRL1 D1 CHARACTER IN BUFFER A BLANK
NO
YES
2
D3 PUT THE ADDRESS OF THE CHARACTER IN A WORK REGISTER
D5 STORE RESULTS OF CALCULATIONS IN LOC TMPAR1 FOR USE BY LATER SCANS *D4
E1 ADD ONE TO THE ADDRESS IN POINT1 SO THAT IT POINTS TO THE NEXT CHARAC
E3 CHANGE THE CHARACTER TO UPPER CASE *E2
E5 ADJUST THE PARAMETER LENGTH COUNTER TO LENGTH MINUS ONE
F1 END OF BUFFER
NO
YES
1
F3 CHAR EQUAL TO A COMMA
YES
NO
F5 ADD ONE TO THE OPERAND NUMBER KEPT IN LOCATION OPNUMB
G3 CHAR EQUAL TO A BLANK
YES
NO
G5 TST LAST CHAR IN THE FIELD FOR A COMMA (SET CONDITION CODE) *G4
H3 CHAR EQUAL TO AN EQUAL SIGN
YES
NO
H5 EXIT TO ADDR IN LINK REGISTER
J3 ADD ONE TO THE PARAMETER LENGTH COUNTER, POINT3
K3 END OF BUFFER
YES
NO
1
2

*D4
POINT1 CONTAINS THE ADDRESS OF THE FIRST CHARACTER OF THE FIELD. POINT2 CONTAINS REMAINING BUFFER LENGTH. POINT3 CONTAINS LENGTH-1.

*E2
THE ORING OF A HEX '40' FORCES UPPER CASE. THIS IS DONE IF THE GENERALIZED ROUTINE IS BEING USED TO PREPARE CARD IMAGES FOR MAPPING. (MSGIN) COMMAND OF MAP)

*G4
LOW EQUALS BLANK. EQUAL EQUALS COMMA. HIGH EQUALS EQUALS. THE CONDITION CODE IS TESTED BY THE CALLING ROUTINE TO ESTABLISH VALIDITY.

*H1
ENTRY AT THIS POINT TO GET THE FIRST FIELD AFTER THE OPERATION CODE. LEADING BLANKS ARE SKIPPED. CURRENT POINTER INFORMATION IS IN LOCATION TMPAR1. AT LEAST ONE SCAN HAS BEEN MADE.

*J1
ENTRY AT THIS POINT TO GET THE OPERATION CODE FIELD. POINTERS ARE INITIALIZED WITHIN THE CALLING SEQUENCE. LEADING BLANKS ARE SKIPPED.

*K1
ENTRY AT THIS POINT TO GET FIELDS OTHER THAN THOSE SPECIFIED IN *H1 AND *J1. CURRENT POINTER INFORMATION IS IN LOCATION TMPAR1. AT LEAST ONE SCAN HAS BEEN MADE. BLANKS ARE NOT SKIPPED.

Chart NE. $$BATTNB - MSG Statement Processor
Refer to Chart 11.

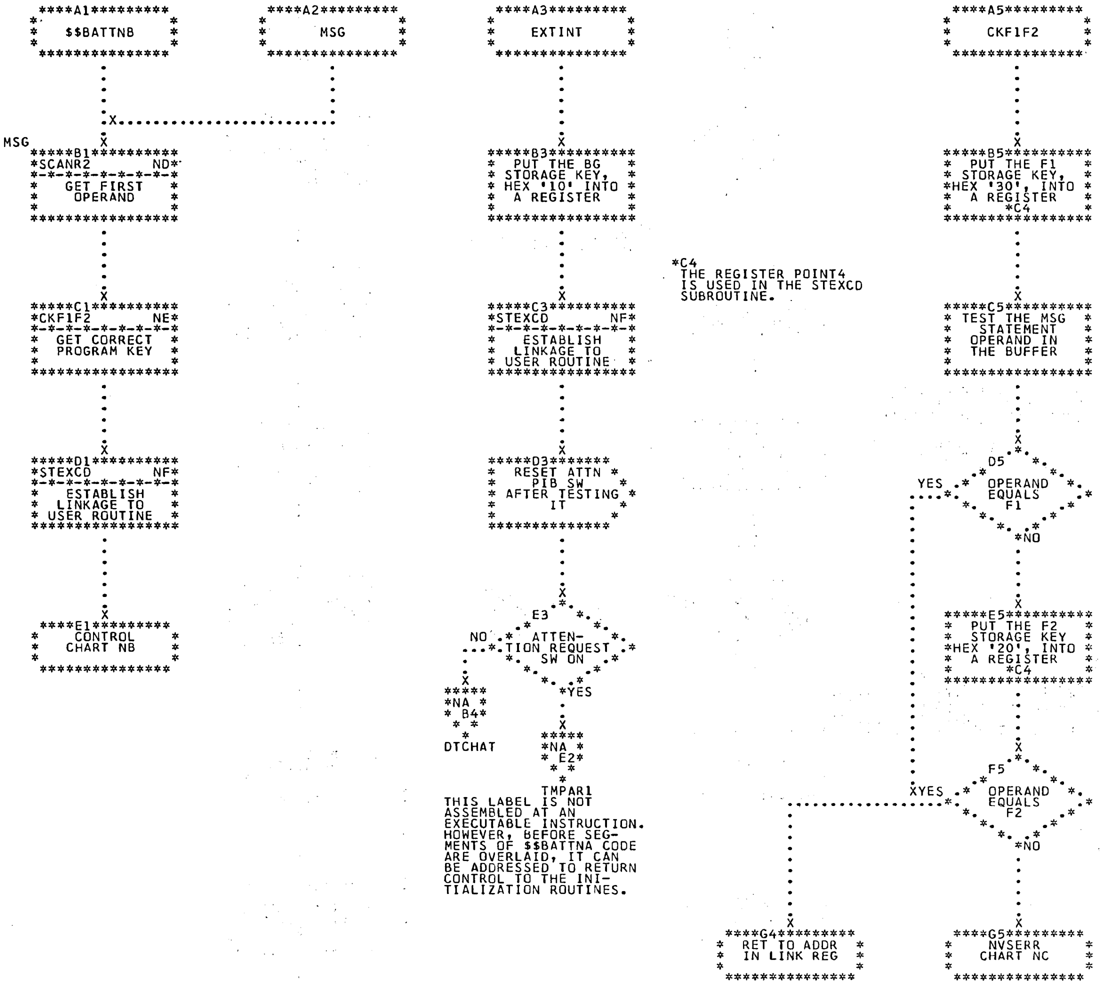

A1
$$BATTNB
A2
MSG
A3
EXTINT
A5
CKF1F2
MSG
B1
SCANR2 ND
GET FIRST OPERAND
B3
PUT THE BG STORAGE KEY, HEX '10' INTO A REGISTER
B5
PUT THE F1 STORAGE KEY, HEX '30', INTO A REGISTER
*C4
*C4
THE REGISTER POINT4 IS USED IN THE STEXCD SUBROUTINE.
C1
CKF1F2 NE
GET CORRECT PROGRAM KEY
C3
STEXCD NF
ESTABLISH LINKAGE TO USER ROUTINE
C5
TEST THE MSG STATEMENT OPERAND IN THE BUFFER
D1
STEXCD NF
ESTABLISH LINKAGE TO USER ROUTINE
D3
RESET ATTN PIB SW AFTER TESTING IT
D5
OPERAND EQUALS F1
YES
NO
E1
CONTROL CHART NB
E3
ATTEN-TION REQUEST SW ON
NO
YES
NA
B4
DTCHAT
NA
E2
TMPAR1
THIS LABEL IS NOT ASSEMBLED AT AN EXECUTABLE INSTRUCTION. HOWEVER, BEFORE SEG-MENTS OF $$BATTNA CODE ARE OVERLAID, IT CAN BE ADDRESSED TO RETURN CONTROL TO THE INI-TIALIZATION ROUTINES.
E5
PUT THE F2 STORAGE KEY HEX '20', INTO A REGISTER
*C4
F5
OPERAND EQUALS F2
YES
NO
G4
RET TO ADDR IN LINK REG
G5
NVSERR CHART NC

Chart NF. $$BATTNB - Set Operator Communications and Exit Table Linkage
Refer to Chart 11.

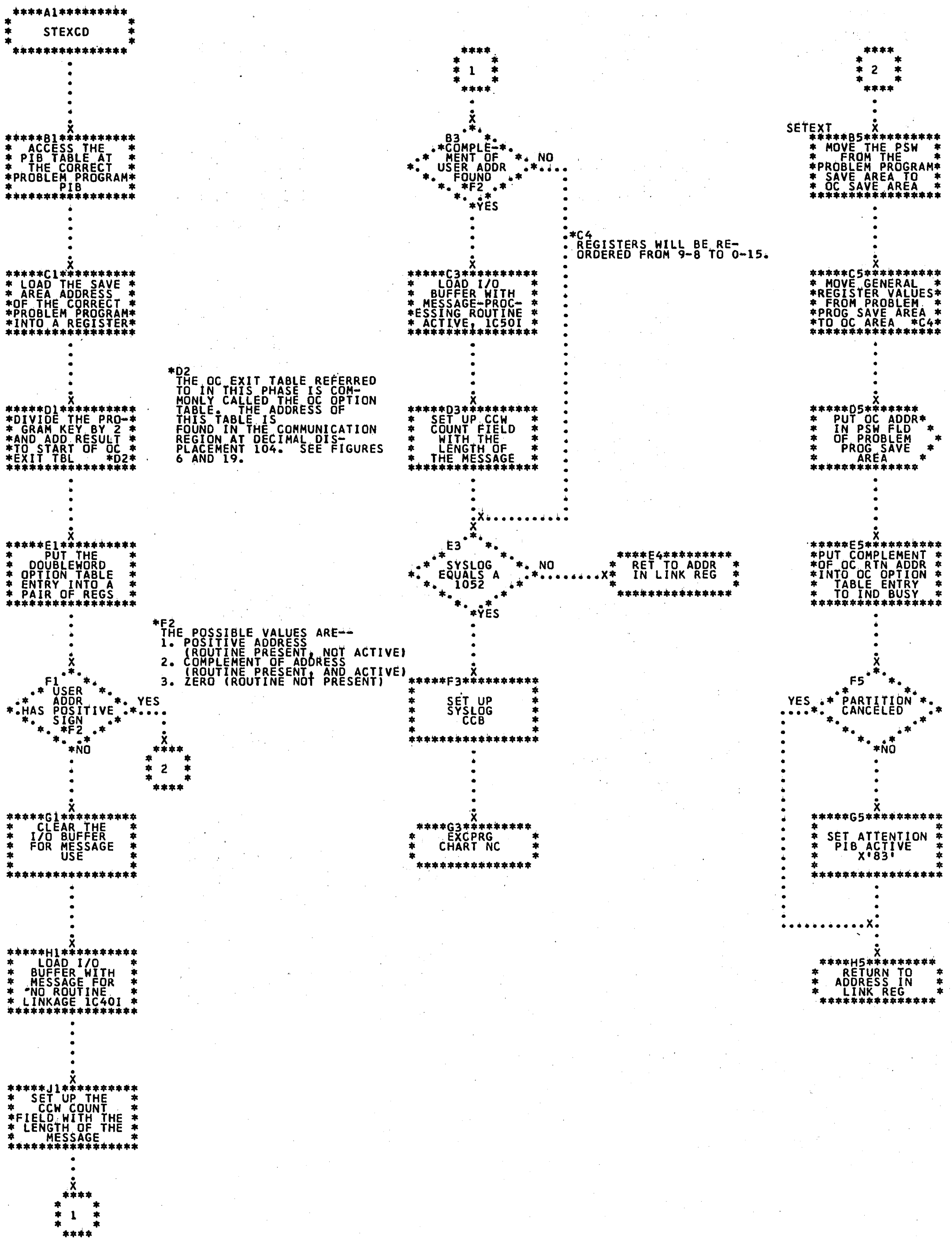

A1 STEXCD
B1 ACCESS THE PIB TABLE AT THE CORRECT PROBLEM PROGRAM PIB
C1 LOAD THE SAVE AREA ADDRESS OF THE CORRECT PROBLEM PROGRAM INTO A REGISTER
D1 DIVIDE THE PROGRAM KEY BY 2 AND ADD RESULT TO START OF OC EXIT TBL *D2
*D2 THE OC EXIT TABLE REFERRED TO IN THIS PHASE IS COMMONLY CALLED THE OC OPTION TABLE. THE ADDRESS OF THIS TABLE IS FOUND IN THE COMMUNICATION REGION AT DECIMAL DISPLACEMENT 104. SEE FIGURES 6 AND 19.
E1 PUT THE DOUBLEWORD OPTION TABLE ENTRY INTO A PAIR OF REGS
F1 USER ADDR HAS POSITIVE SIGN *F2
YES
NO
2
*F2 THE POSSIBLE VALUES ARE--
1. POSITIVE ADDRESS (ROUTINE PRESENT, NOT ACTIVE)
2. COMPLEMENT OF ADDRESS (ROUTINE PRESENT, AND ACTIVE)
3. ZERO (ROUTINE NOT PRESENT)
G1 CLEAR THE I/O BUFFER FOR MESSAGE USE
H1 LOAD I/O BUFFER WITH MESSAGE FOR "NO ROUTINE LINKAGE 1C40I
J1 SET UP THE CCW COUNT FIELD WITH THE LENGTH OF THE MESSAGE
1
1
B3 COMPLEMENT OF USER ADDR FOUND *F2
NO
YES
*C4 REGISTERS WILL BE REORDERED FROM 9-8 TO 0-15.
C3 LOAD I/O BUFFER WITH MESSAGE-PROCESSING ROUTINE ACTIVE, 1C50I
D3 SET UP CCW COUNT FIELD WITH THE LENGTH OF THE MESSAGE
E3 SYSLOG EQUALS A 1052
NO
YES
E4 RET TO ADDR IN LINK REG
F3 SET UP SYSLOG CCB
G3 EXCPRG CHART NC
2
SETEXT
B5 MOVE THE PSW FROM THE PROBLEM PROGRAM SAVE AREA TO OC SAVE AREA
C5 MOVE GENERAL REGISTER VALUES FROM PROBLEM PROG SAVE AREA TO OC AREA *C4
D5 PUT OC ADDR IN PSW FLD OF PROBLEM PROG SAVE AREA
E5 PUT COMPLEMENT OF OC RTN ADDR INTO OC OPTION TABLE ENTRY TO IND BUSY
F5 PARTITION CANCELED
YES
NO
G5 SET ATTENTION PIB ACTIVE X'83'
H5 RETURN TO ADDRESS IN LINK REG

Chart NG. $$BATTNC - CANCEL Statement Processor
Refer to Chart 11.

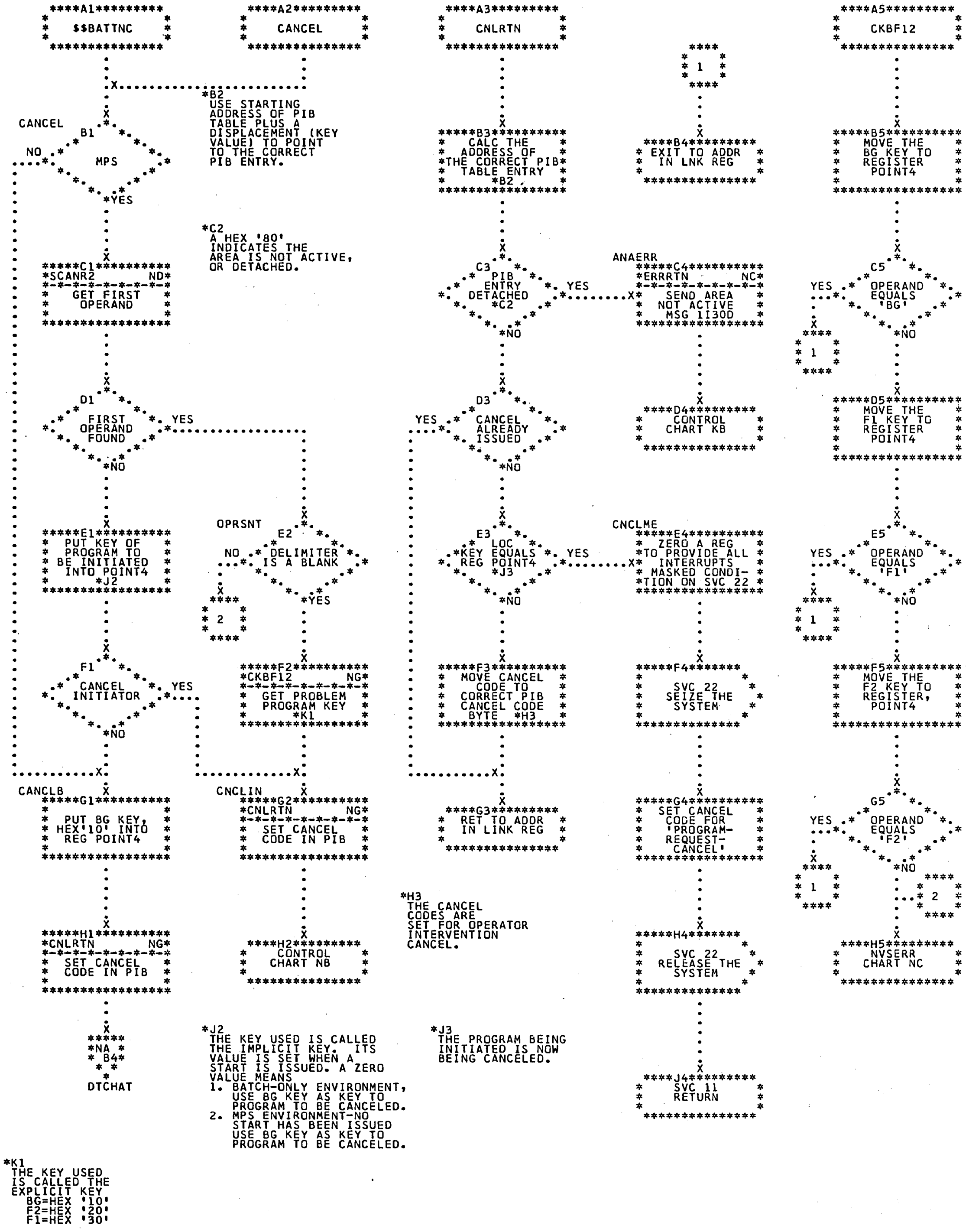

A1 $$BATTNC
A2 CANCEL
A3 CNLRTN
A5 CKBF12
B2 USE STARTING ADDRESS OF PIB TABLE PLUS A DISPLACEMENT (KEY VALUE) TO POINT TO THE CORRECT PIB ENTRY.
CANCEL B1 MPS
NO
YES
B3 CALC THE ADDRESS OF THE CORRECT PIB TABLE ENTRY *B2
1
B4 EXIT TO ADDR IN LNK REG
B5 MOVE THE BG KEY TO REGISTER POINT4
C2 A HEX '80' INDICATES THE AREA IS NOT ACTIVE, OR DETACHED.
C1 SCANR2 ND GET FIRST OPERAND
C3 PIB ENTRY DETACHED *C2
YES
NO
ANAERR
C4 ERRRTN NC SEND AREA NOT ACTIVE MSG 1130D
C5 OPERAND EQUALS 'BG'
YES
NO
1
D1 FIRST OPERAND FOUND
YES
NO
D3 CANCEL ALREADY ISSUED
YES
NO
D4 CONTROL CHART KB
D5 MOVE THE F1 KEY TO REGISTER POINT4
E1 PUT KEY OF PROGRAM TO BE INITIATED INTO POINT4 *J2
OPRSNT E2 DELIMITER IS A BLANK
NO
YES
2
E3 LOC KEY EQUALS REG POINT4 *J3
YES
NO
CNCLME
E4 ZERO A REG TO PROVIDE ALL INTERRUPTS MASKED CONDITION ON SVC 22
E5 OPERAND EQUALS 'F1'
YES
NO
1
F1 CANCEL INITIATOR
YES
NO
F2 CKBF12 NG GET PROBLEM PROGRAM KEY *K1
F3 MOVE CANCEL CODE TO CORRECT PIB CANCEL CODE BYTE *H3
F4 SVC 22 SEIZE THE SYSTEM
F5 MOVE THE F2 KEY TO REGISTER, POINT4
CANCLB
G1 PUT BG KEY, HEX'10' INTO REG POINT4
CNCLIN
G2 CNLRTN NG SET CANCEL CODE IN PIB
G3 RET TO ADDR IN LINK REG
G4 SET CANCEL CODE FOR 'PROGRAM-REQUEST-CANCEL'
G5 OPERAND EQUALS 'F2'
YES
NO
1
2
H3 THE CANCEL CODES ARE SET FOR OPERATOR INTERVENTION CANCEL.
H1 CNLRTN NG SET CANCEL CODE IN PIB
H2 CONTROL CHART NB
H4 SVC 22 RELEASE THE SYSTEM
H5 NVSERR CHART NC
NA B4
DTCHAT
J2 THE KEY USED IS CALLED THE IMPLICIT KEY. ITS VALUE IS SET WHEN A START IS ISSUED. A ZERO VALUE MEANS
1. BATCH-ONLY ENVIRONMENT, USE BG KEY AS KEY TO PROGRAM TO BE CANCELED.
2. MPS ENVIRONMENT-NO START HAS BEEN ISSUED USE BG KEY AS KEY TO PROGRAM TO BE CANCELED.
J3 THE PROGRAM BEING INITIATED IS NOW BEING CANCELED.
J4 SVC 11 RETURN
K1 THE KEY USED IS CALLED THE EXPLICIT KEY.
BG=HEX '10'
F2=HEX '20'
F1=HEX '30'

Chart NH. $$BATTNC - PAUSE, LOG, NOLOG, and IGNORE Statement Processors
Refer to Chart 11.

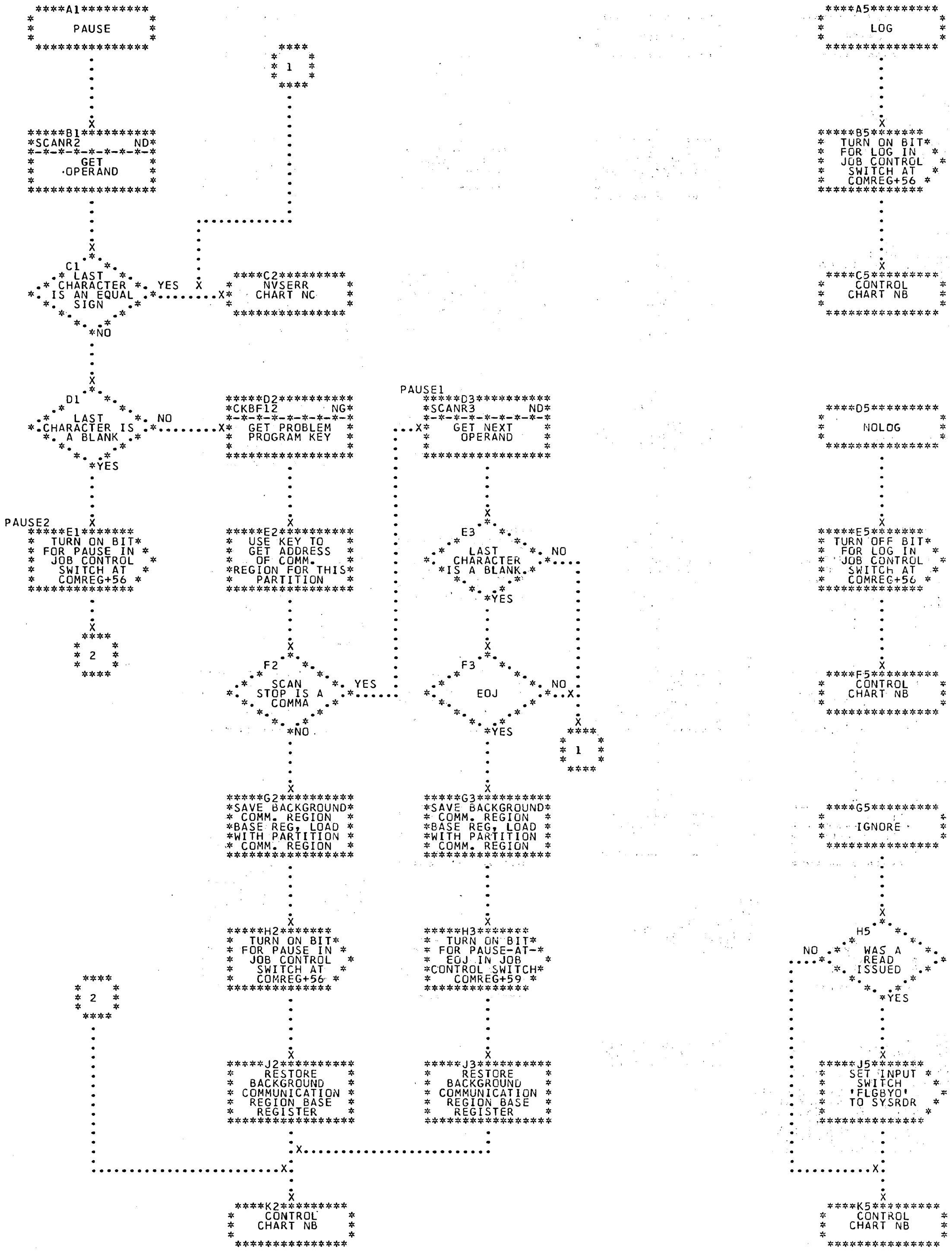

A1
PAUSE
1
B1
SCANR2 ND
GET OPERAND
C1
LAST CHARACTER IS AN EQUAL SIGN
YES
NO
C2
NVSERR CHART NC
D1
LAST CHARACTER IS A BLANK
NO
YES
D2
CKBF12 NG
GET PROBLEM PROGRAM KEY
PAUSE1
D3
SCANR3 ND
GET NEXT OPERAND
PAUSE2
E1
TURN ON BIT FOR PAUSE IN JOB CONTROL SWITCH AT COMREG+56
2
E2
USE KEY TO GET ADDRESS OF COMM. REGION FOR THIS PARTITION
E3
LAST CHARACTER IS A BLANK
NO
YES
F2
SCAN STOP IS A COMMA
YES
NO
F3
EOJ
NO
YES
1
G2
SAVE BACKGROUND COMM. REGION BASE REG, LOAD WITH PARTITION COMM. REGION
G3
SAVE BACKGROUND COMM. REGION BASE REG, LOAD WITH PARTITION COMM. REGION
H2
TURN ON BIT FOR PAUSE IN JOB CONTROL SWITCH AT COMREG+56
H3
TURN ON BIT FOR PAUSE-AT-EOJ IN JOB CONTROL SWITCH COMREG+59
2
J2
RESTORE BACKGROUND COMMUNICATION REGION BASE REGISTER
J3
RESTORE BACKGROUND COMMUNICATION REGION BASE REGISTER
K2
CONTROL CHART NB
A5
LOG
B5
TURN ON BIT FOR LOG IN JOB CONTROL SWITCH AT COMREG+56
C5
CONTROL CHART NB
D5
NOLOG
E5
TURN OFF BIT FOR LOG IN JOB CONTROL SWITCH AT COMREG+56
F5
CONTROL CHART NB
G5
IGNORE
H5
WAS A READ ISSUED
NO
YES
J5
SET INPUT SWITCH 'FLGBYO' TO SYSRDR
K5
CONTROL CHART NB

Chart NJ. $$BATTND - MAP Statement Processor
Refer to Chart 11.

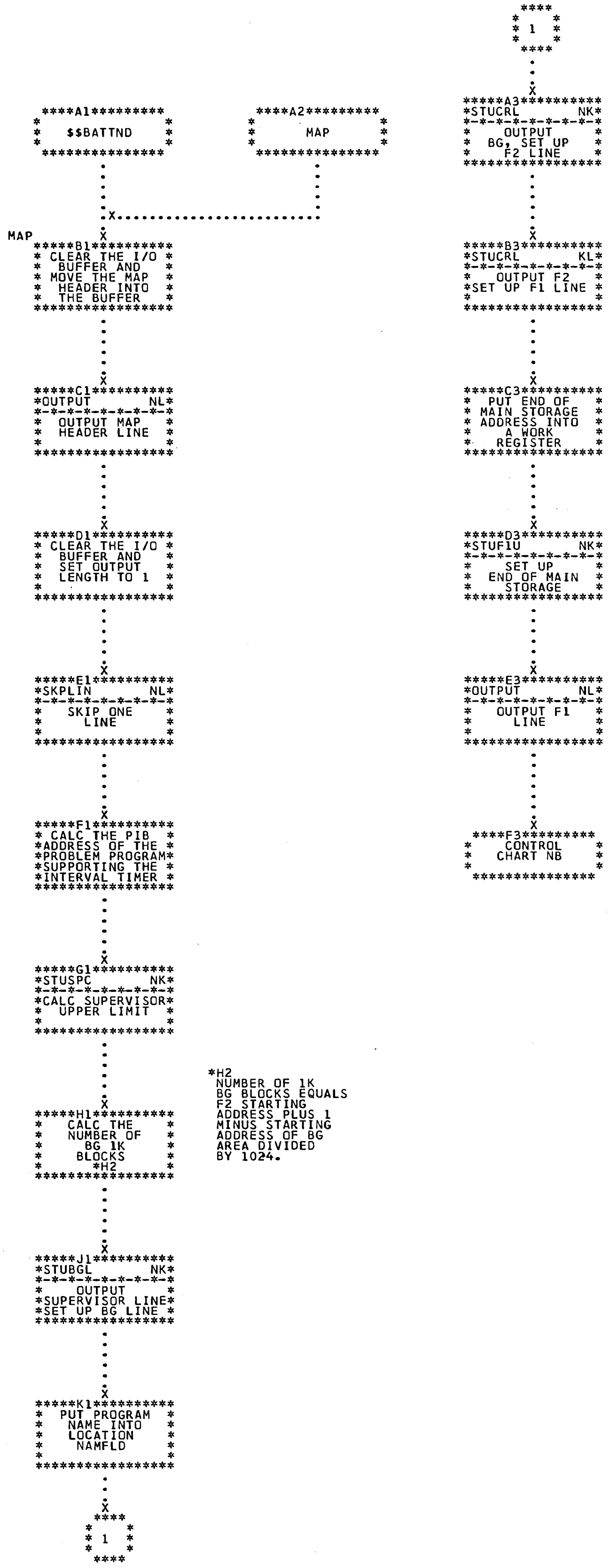

1
A1
$$BATTND
A2
MAP
A3
STUCRL NK
OUTPUT BG, SET UP F2 LINE
MAP
B1
CLEAR THE I/O BUFFER AND MOVE THE MAP HEADER INTO THE BUFFER
B3
STUCRL KL
OUTPUT F2 SET UP F1 LINE
C1
OUTPUT NL
OUTPUT MAP HEADER LINE
C3
PUT END OF MAIN STORAGE ADDRESS INTO A WORK REGISTER
D1
CLEAR THE I/O BUFFER AND SET OUTPUT LENGTH TO 1
D3
STUF1U NK
SET UP END OF MAIN STORAGE
E1
SKPLIN NL
SKIP ONE LINE
E3
OUTPUT NL
OUTPUT F1 LINE
F1
CALC THE PIB ADDRESS OF THE PROBLEM PROGRAM SUPPORTING THE INTERVAL TIMER
F3
CONTROL CHART NB
G1
STUSPC NK
CALC SUPERVISOR UPPER LIMIT
*H2
NUMBER OF 1K BG BLOCKS EQUALS F2 STARTING ADDRESS PLUS 1 MINUS STARTING ADDRESS OF BG AREA DIVIDED BY 1024.
H1
CALC THE NUMBER OF BG 1K BLOCKS *H2
J1
STUBGL NK
OUTPUT SUPERVISOR LINE SET UP BG LINE
K1
PUT PROGRAM NAME INTO LOCATION NAMFLD
1

Chart NK. $$BATTND - Output MAP Subroutines (Part 1 of 2)
Refer to Chart 11.

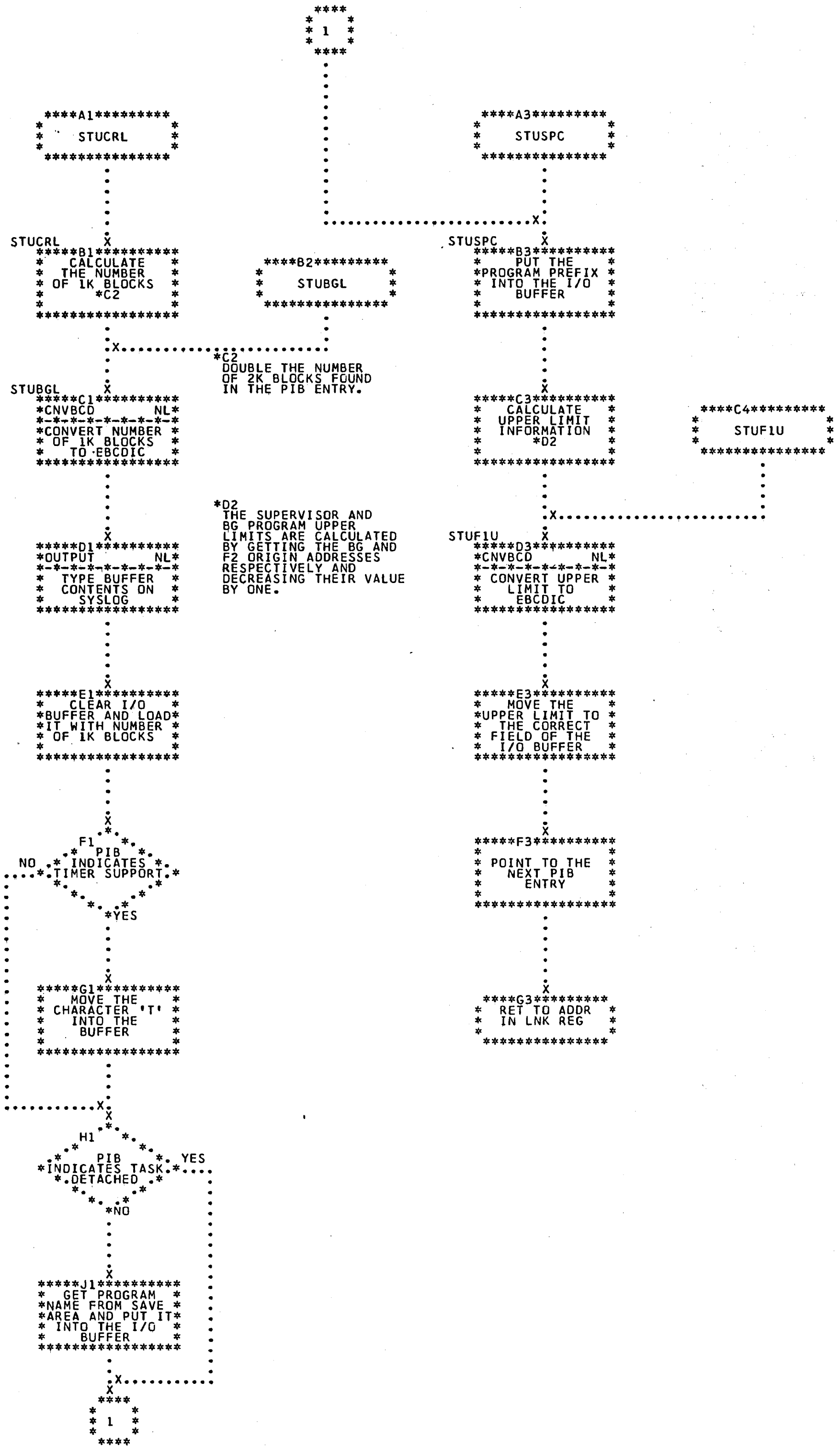

1
A1
STUCRL
A3
STUSPC
STUCRL
B1
CALCULATE THE NUMBER OF 1K BLOCKS *C2
B2
STUBGL
STUSPC
B3
PUT THE PROGRAM PREFIX INTO THE I/O BUFFER
*C2
DOUBLE THE NUMBER OF 2K BLOCKS FOUND IN THE PIB ENTRY.
STUBGL
C1
CNVBCD NL
CONVERT NUMBER OF 1K BLOCKS TO EBCDIC
C3
CALCULATE UPPER LIMIT INFORMATION *D2
C4
STUF1U
*D2
THE SUPERVISOR AND BG PROGRAM UPPER LIMITS ARE CALCULATED BY GETTING THE BG AND F2 ORIGIN ADDRESSES RESPECTIVELY AND DECREASING THEIR VALUE BY ONE.
D1
OUTPUT NL
TYPE BUFFER CONTENTS ON SYSLOG
STUF1U
D3
CNVBCD NL
CONVERT UPPER LIMIT TO EBCDIC
E1
CLEAR I/O BUFFER AND LOAD IT WITH NUMBER OF 1K BLOCKS
E3
MOVE THE UPPER LIMIT TO THE CORRECT FIELD OF THE I/O BUFFER
F1
PIB INDICATES TIMER SUPPORT
NO
YES
F3
POINT TO THE NEXT PIB ENTRY
G1
MOVE THE CHARACTER 'T' INTO THE BUFFER
G3
RET TO ADDR IN LNK REG
H1
PIB INDICATES TASK DETACHED
YES
NO
J1
GET PROGRAM NAME FROM SAVE AREA AND PUT IT INTO THE I/O BUFFER
1

Chart NL. $$BATTND - Output MAP Subroutines (Part 2 of 2)
Refer to Chart 11.

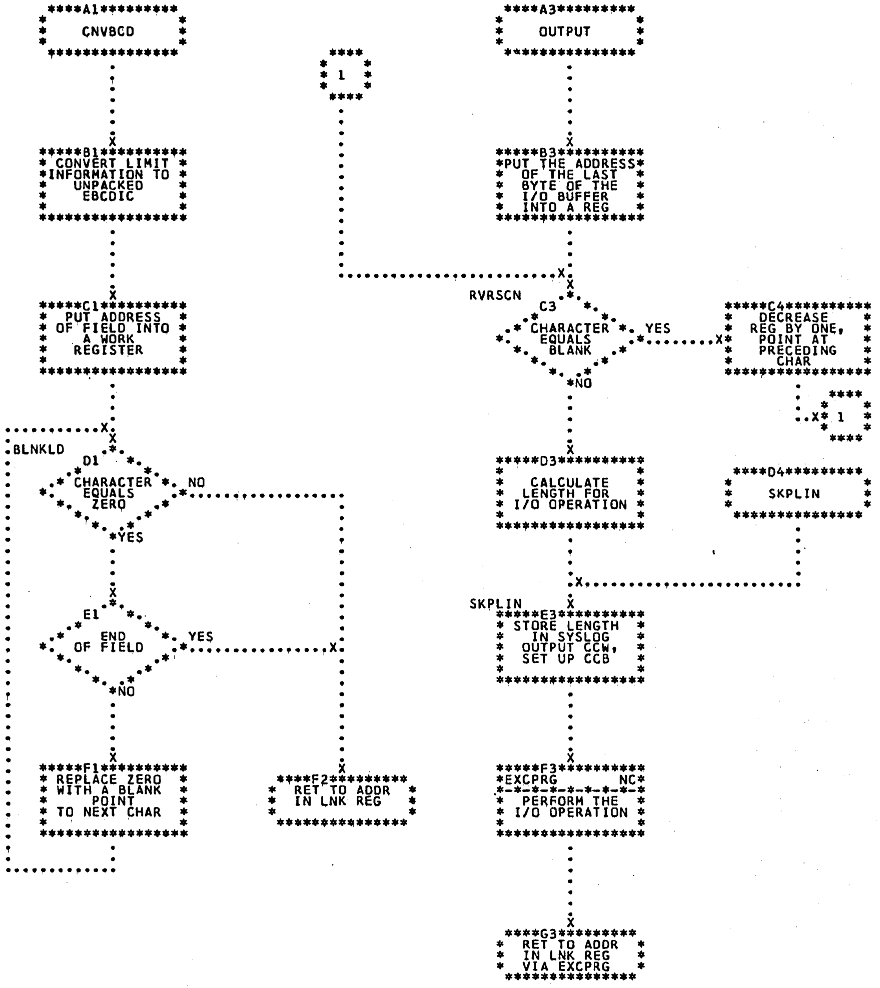
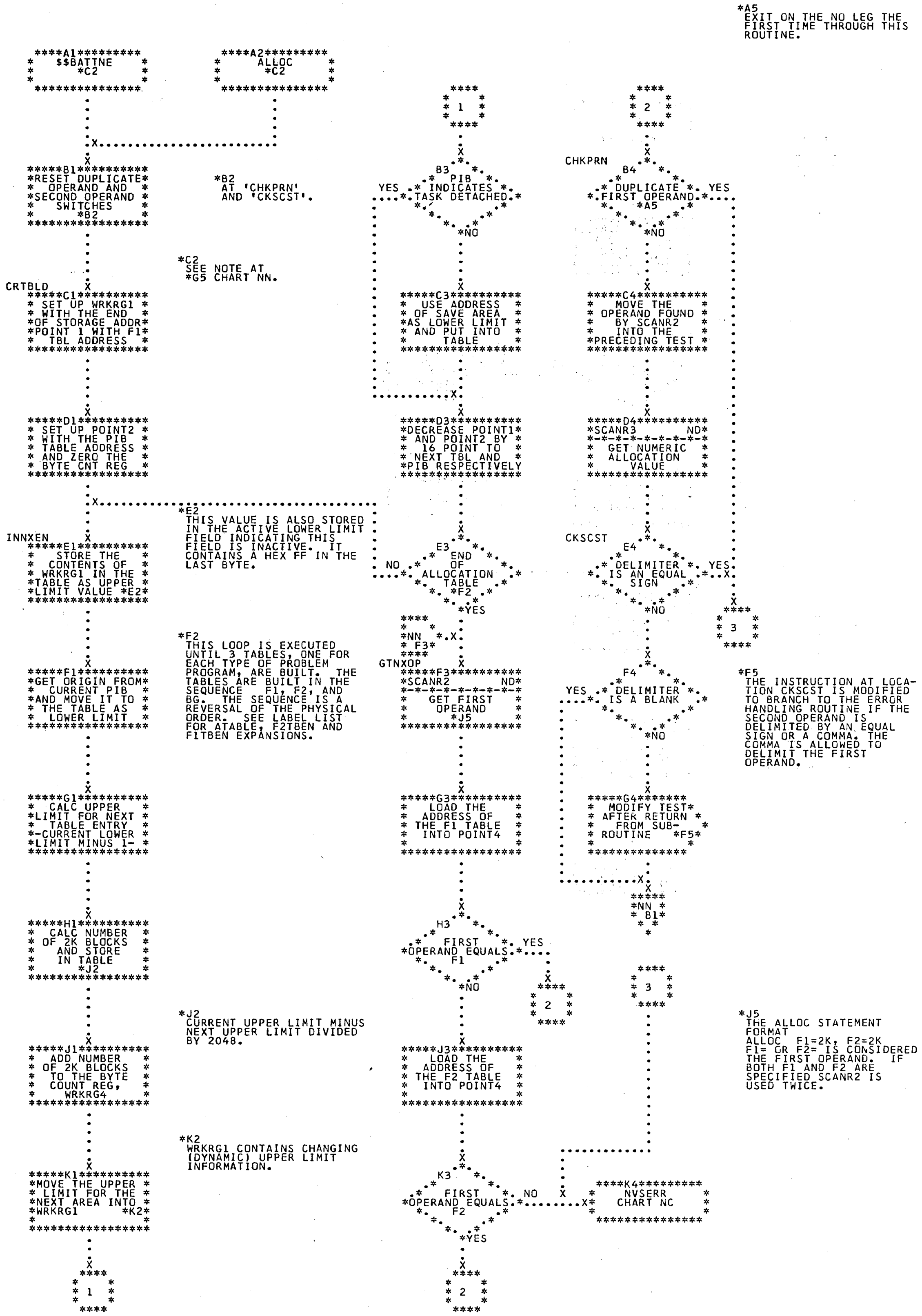

A1
CNVBCD
B1
CONVERT LIMIT INFORMATION TO UNPACKED EBCDIC
C1
PUT ADDRESS OF FIELD INTO A WORK REGISTER
BLNKLD
D1
CHARACTER EQUALS ZERO
NO
YES
E1
END OF FIELD
YES
NO
F1
REPLACE ZERO WITH A BLANK POINT TO NEXT CHAR
F2
RET TO ADDR IN LNK REG
1
A3
OUTPUT
B3
PUT THE ADDRESS OF THE LAST BYTE OF THE I/O BUFFER INTO A REG
RVRSCN
C3
CHARACTER EQUALS BLANK
YES
NO
C4
DECREASE REG BY ONE, POINT AT PRECEDING CHAR
1
D3
CALCULATE LENGTH FOR I/O OPERATION
D4
SKPLIN
SKPLIN
E3
STORE LENGTH IN SYSLOG OUTPUT CCW, SET UP CCB
F3
EXCPRG NC
PERFORM THE I/O OPERATION
G3
RET TO ADDR IN LNK REG VIA EXCPRG

Chart NM. $$BATTNE - ALLOC Statement Processor (Part 1 of 4)
Refer to Chart 12.


A1 $$BATTNE *C2
A2 ALLOC *C2
B1 RESET DUPLICATE OPERAND AND SECOND OPERAND SWITCHES *B2
CRTBLD
C1 SET UP WRKRG1 WITH THE END OF STORAGE ADDR POINT 1 WITH F1 TBL ADDRESS
D1 SET UP POINT2 WITH THE PIB TABLE ADDRESS AND ZERO THE BYTE CNT REG
INNXEN
E1 STORE THE CONTENTS OF WRKRG1 IN THE TABLE AS UPPER LIMIT VALUE *E2
F1 GET ORIGIN FROM CURRENT PIB AND MOVE IT TO THE TABLE AS LOWER LIMIT
G1 CALC UPPER LIMIT FOR NEXT TABLE ENTRY -CURRENT LOWER LIMIT MINUS 1-
H1 CALC NUMBER OF 2K BLOCKS AND STORE IN TABLE *J2
J1 ADD NUMBER OF 2K BLOCKS TO THE BYTE COUNT REG, WRKRG4
K1 MOVE THE UPPER LIMIT FOR THE NEXT AREA INTO WRKRG1 *K2
1
1
B3 PIB INDICATES TASK DETACHED
YES
NO
C3 USE ADDRESS OF SAVE AREA AS LOWER LIMIT AND PUT INTO TABLE
D3 DECREASE POINT1 AND POINT2 BY 16 POINT TO NEXT TBL AND PIB RESPECTIVELY
E3 END OF ALLOCATION TABLE *F2
NO
YES
NN F3
GTNXOP
F3 SCANR2 ND GET FIRST OPERAND *J5
G3 LOAD THE ADDRESS OF THE F1 TABLE INTO POINT4
H3 FIRST OPERAND EQUALS F1
YES
NO
2
J3 LOAD THE ADDRESS OF THE F2 TABLE INTO POINT4
K3 FIRST OPERAND EQUALS F2
NO
YES
2
2
CHKPRN
B4 DUPLICATE FIRST OPERAND *A5
YES
NO
C4 MOVE THE OPERAND FOUND BY SCANR2 INTO THE PRECEDING TEST
D4 SCANR3 ND GET NUMERIC ALLOCATION VALUE
CKSCST
E4 DELIMITER IS AN EQUAL SIGN
YES
NO
3
F4 DELIMITER IS A BLANK
YES
NO
G4 MODIFY TEST AFTER RETURN FROM SUB-ROUTINE *F5
NN B1
3
K4 NVSERR CHART NC

*A5
EXIT ON THE NO LEG THE FIRST TIME THROUGH THIS ROUTINE.

*B2
AT 'CHKPRN' AND 'CKSCST'.

*C2
SEE NOTE AT *G5 CHART NN.

*E2
THIS VALUE IS ALSO STORED IN THE ACTIVE LOWER LIMIT FIELD INDICATING THIS FIELD IS INACTIVE. IT CONTAINS A HEX FF IN THE LAST BYTE.

*F2
THIS LOOP IS EXECUTED UNTIL 3 TABLES, ONE FOR EACH TYPE OF PROBLEM PROGRAM, ARE BUILT. THE TABLES ARE BUILT IN THE SEQUENCE F1, F2, AND BG. THE SEQUENCE IS A REVERSAL OF THE PHYSICAL ORDER. SEE LABEL LIST FOR ATABLE, F2TBEN AND F1TBEN EXPANSIONS.

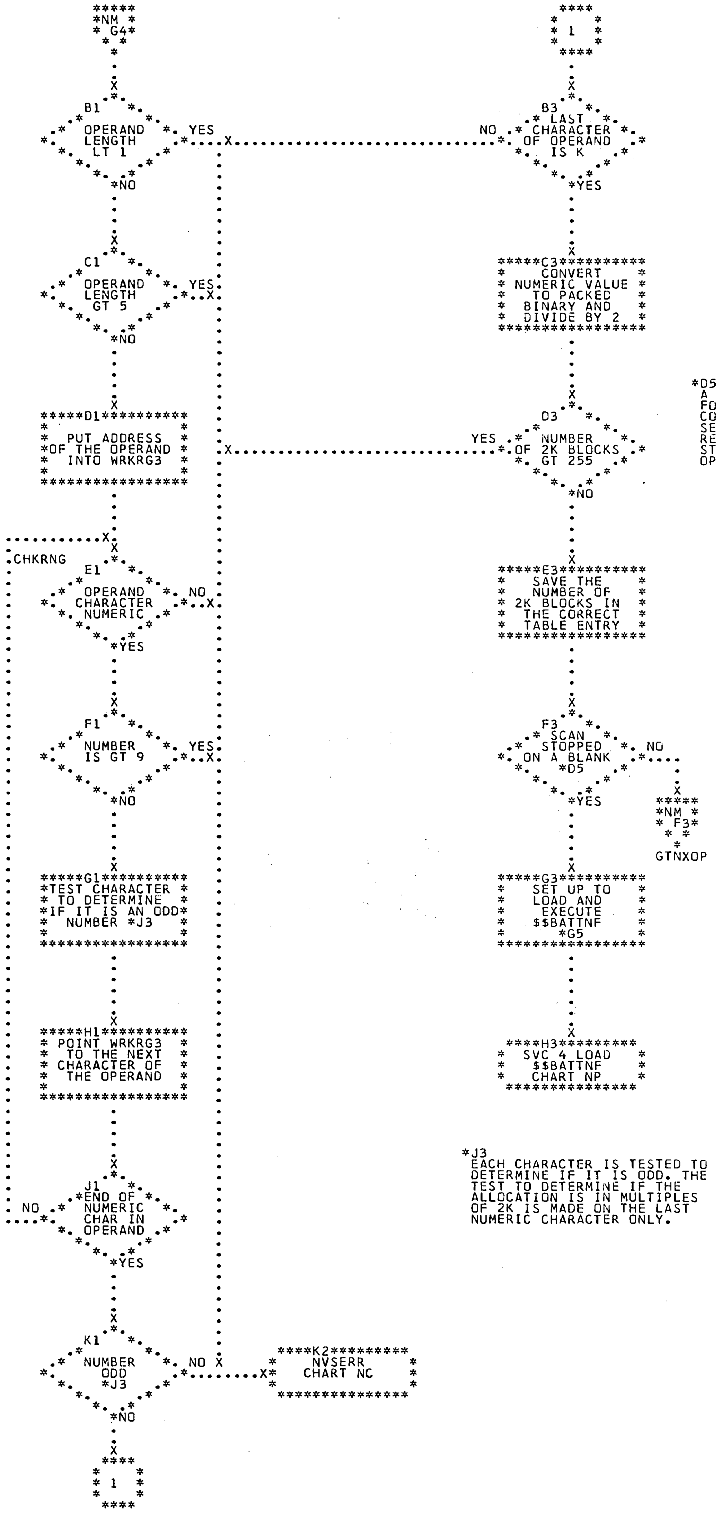
*F5
THE INSTRUCTION AT LOCATION CKSCST IS MODIFIED TO BRANCH TO THE ERROR HANDLING ROUTINE IF THE SECOND OPERAND IS DELIMITED BY AN EQUAL SIGN OR A COMMA. THE COMMA IS ALLOWED TO DELIMIT THE FIRST OPERAND.

*J2
CURRENT UPPER LIMIT MINUS NEXT UPPER LIMIT DIVIDED BY 2048.

*J5
THE ALLOC STATEMENT FORMAT
ALLOC F1=2K, F2=2K
F1= OR F2= IS CONSIDERED THE FIRST OPERAND. IF BOTH F1 AND F2 ARE SPECIFIED SCANR2 IS USED TWICE.

*K2
WRKRG1 CONTAINS CHANGING (DYNAMIC) UPPER LIMIT INFORMATION.

Chart NN. $$BATTNE - ALLOC Statement Processor (Part 2 of 4)
Refer to Chart 12.


NM G4
B1 OPERAND LENGTH LT 1
YES
NO
C1 OPERAND LENGTH GT 5
YES
NO
D1 PUT ADDRESS OF THE OPERAND INTO WRKRG3
CHKRNG
E1 OPERAND CHARACTER NUMERIC
NO
YES
F1 NUMBER IS GT 9
YES
NO
G1 TEST CHARACTER TO DETERMINE IF IT IS AN ODD NUMBER *J3
H1 POINT WRKRG3 TO THE NEXT CHARACTER OF THE OPERAND
J1 END OF NUMERIC CHAR IN OPERAND
NO
YES
K1 NUMBER ODD *J3
NO
NO
K2 NVSERR CHART NC
1
1
B3 LAST CHARACTER OF OPERAND IS K
NO
YES
C3 CONVERT NUMERIC VALUE TO PACKED BINARY AND DIVIDE BY 2
D3 NUMBER OF 2K BLOCKS GT 255
YES
NO
E3 SAVE THE NUMBER OF 2K BLOCKS IN THE CORRECT TABLE ENTRY
F3 SCAN STOPPED ON A BLANK *D5
NO
YES
NM F3
GTNXOP
G3 SET UP TO LOAD AND EXECUTE $$BATTNF *G5
H3 SVC 4 LOAD $$BATTNF CHART NP

*D5
A BLANK DELIMITER INDICATES NO MORE OPERANDS FOLLOW. HOWEVER, AN ALLOC STATEMENT CAN CONTAIN TWO OPERAND FIELDS (F1 AND F2) SEPARATED BY A COMMA. A COMMA DELIMITER REQUIRES ANOTHER PASS THROUGH THIS ROUTINE STARTING AT GTNXOP TO PROCESS THE SECOND OPERAND.

*G5
THE LOAD INSTRUCTION IS PHYSICALLY THE FIRST INSTRUCTION FOUND IN THE LISTING OF THIS TRANSIENT. THE BRANCH VECTOR TABLE FOUND IN THE ROOT PHASE, $$BATTNA, CAUSES PROCESSING TO BEGIN WITH THE SECOND EXECUTABLE INSTRUCTION. WHEN THIS TRANSIENT IS FINISHED PROCESSING, THE LOAD IS EXECUTED BY BRANCHING TO THE ADDRESS IN THE BASE REGISTER. THIS IS THE ADDRESS OF THE SVC 4 INSTRUCTION. $$BATTNF IS LOADED SO THAT IT OVERLAYS THE FIRST PART OF $$BATTNE. $$BATTNF ALSO HAS A SVC 4 AS ITS FIRST INSTRUCTION, BUT SUPERVISOR GIVES CONTROL TO $$BATTNF AT THE NEXT INSTRUCTION FOLLOWING A SVC 4.

*J3
EACH CHARACTER IS TESTED TO DETERMINE IF IT IS ODD. THE TEST TO DETERMINE IF THE ALLOCATION IS IN MULTIPLES OF 2K IS MADE ON THE LAST NUMERIC CHARACTER ONLY.

Chart NP. $$BATTNF - ALLOC Statement Processor (Part 3 of 4)
Refer to Chart 12.

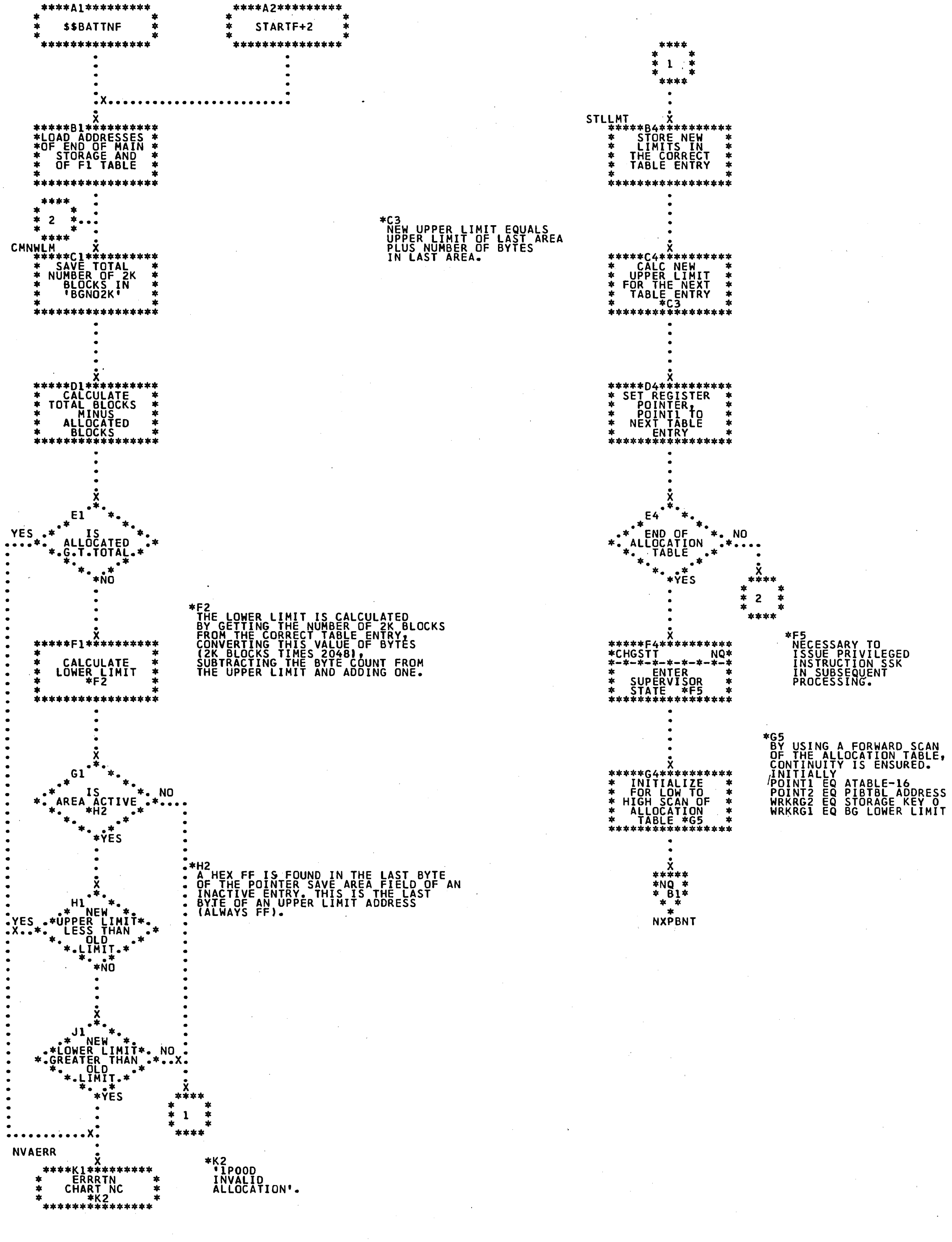
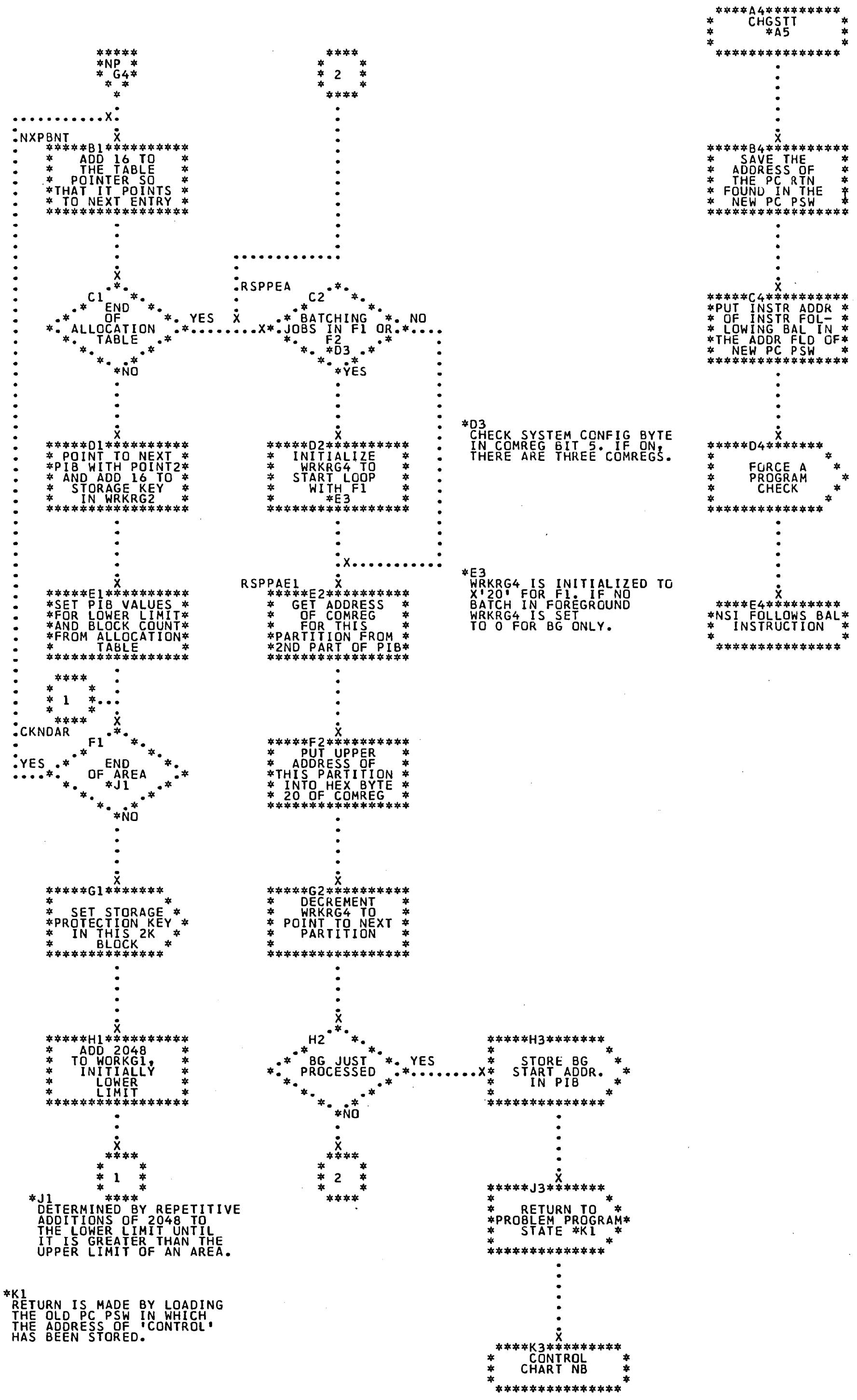

A1 $$BATTNF
A2 STARTF+2
B1 LOAD ADDRESSES OF END OF MAIN STORAGE AND OF F1 TABLE
2
CMNWLM
C1 SAVE TOTAL NUMBER OF 2K BLOCKS IN 'BGNO2K'
D1 CALCULATE TOTAL BLOCKS MINUS ALLOCATED BLOCKS
E1 IS ALLOCATED G.T.TOTAL
YES
NO
F1 CALCULATE LOWER LIMIT *F2
G1 IS AREA ACTIVE *H2
NO
YES
H1 NEW UPPER LIMIT LESS THAN OLD LIMIT
YES
NO
J1 NEW LOWER LIMIT GREATER THAN OLD LIMIT
NO
YES
1
NVAERR
K1 ERRRTN CHART NC *K2
*C3 NEW UPPER LIMIT EQUALS UPPER LIMIT OF LAST AREA PLUS NUMBER OF BYTES IN LAST AREA.
*F2 THE LOWER LIMIT IS CALCULATED BY GETTING THE NUMBER OF 2K BLOCKS FROM THE CORRECT TABLE ENTRY, CONVERTING THIS VALUE OF BYTES (2K BLOCKS TIMES 2048), SUBTRACTING THE BYTE COUNT FROM THE UPPER LIMIT AND ADDING ONE.
*H2 A HEX FF IS FOUND IN THE LAST BYTE OF THE POINTER SAVE AREA FIELD OF AN INACTIVE ENTRY. THIS IS THE LAST BYTE OF AN UPPER LIMIT ADDRESS (ALWAYS FF).
*K2 '1P00D INVALID ALLOCATION'.
1
STLLMT
B4 STORE NEW LIMITS IN THE CORRECT TABLE ENTRY
C4 CALC NEW UPPER LIMIT FOR THE NEXT TABLE ENTRY *C3
D4 SET REGISTER POINTER, POINT1 TO NEXT TABLE ENTRY
E4 END OF ALLOCATION TABLE
NO
YES
2
F4 CHGSTT NQ
ENTER SUPERVISOR STATE *F5
G4 INITIALIZE FOR LOW TO HIGH SCAN OF ALLOCATION TABLE *G5
NQ B1
NXPBNT
*F5 NECESSARY TO ISSUE PRIVILEGED INSTRUCTION SSK IN SUBSEQUENT PROCESSING.
*G5 BY USING A FORWARD SCAN OF THE ALLOCATION TABLE, CONTINUITY IS ENSURED. INITIALLY POINT1 EQ ATABLE-16 POINT2 EQ PIBTBL ADDRESS WRKRG2 EQ STORAGE KEY 0 WRKRG1 EQ BG LOWER LIMIT

Chart NQ. $$BATTNF - ALLOC Statement Processor (Part 4 of 4)
Refer to Chart 12.



NP G4
2
NXPBNT
B1 ADD 16 TO THE TABLE POINTER SO THAT IT POINTS TO NEXT ENTRY
C1 END OF ALLOCATION TABLE
YES
NO
RSPPEA
C2 BATCHING JOBS IN F1 OR F2 *D3
YES
NO
D1 POINT TO NEXT PIB WITH POINT2 AND ADD 16 TO STORAGE KEY IN WRKRG2
D2 INITIALIZE WRKRG4 TO START LOOP WITH F1 *E3
E1 SET PIB VALUES FOR LOWER LIMIT AND BLOCK COUNT FROM ALLOCATION TABLE
RSPPAE1
E2 GET ADDRESS OF COMREG FOR THIS PARTITION FROM 2ND PART OF PIB
1
CKNDAR
F1 END OF AREA *J1
YES
NO
F2 PUT UPPER ADDRESS OF THIS PARTITION INTO HEX BYTE 20 OF COMREG
G1 SET STORAGE PROTECTION KEY IN THIS 2K BLOCK
G2 DECREMENT WRKRG4 TO POINT TO NEXT PARTITION
H1 ADD 2048 TO WORKG1, INITIALLY LOWER LIMIT
H2 BG JUST PROCESSED
YES
NO
H3 STORE BG START ADDR. IN PIB
1
2
J3 RETURN TO PROBLEM PROGRAM STATE *K1
K3 CONTROL CHART NB
A4 CHGSTT *A5
B4 SAVE THE ADDRESS OF THE PC RTN FOUND IN THE NEW PC PSW
C4 PUT INSTR ADDR OF INSTR FOL- LOWING BAL IN THE ADDR FLD OF NEW PC PSW
D4 FORCE A PROGRAM CHECK
E4 NSI FOLLOWS BAL INSTRUCTION

*A5
USED TO ENTER AND EXIT THE SUPVERSIOR STATE. WHEN USED AS AN EXIT, CERTAIN BIT SWITCHES IN THE SVC NEW PSW ARE RESET TO THEIR ORIGINAL VALUES.

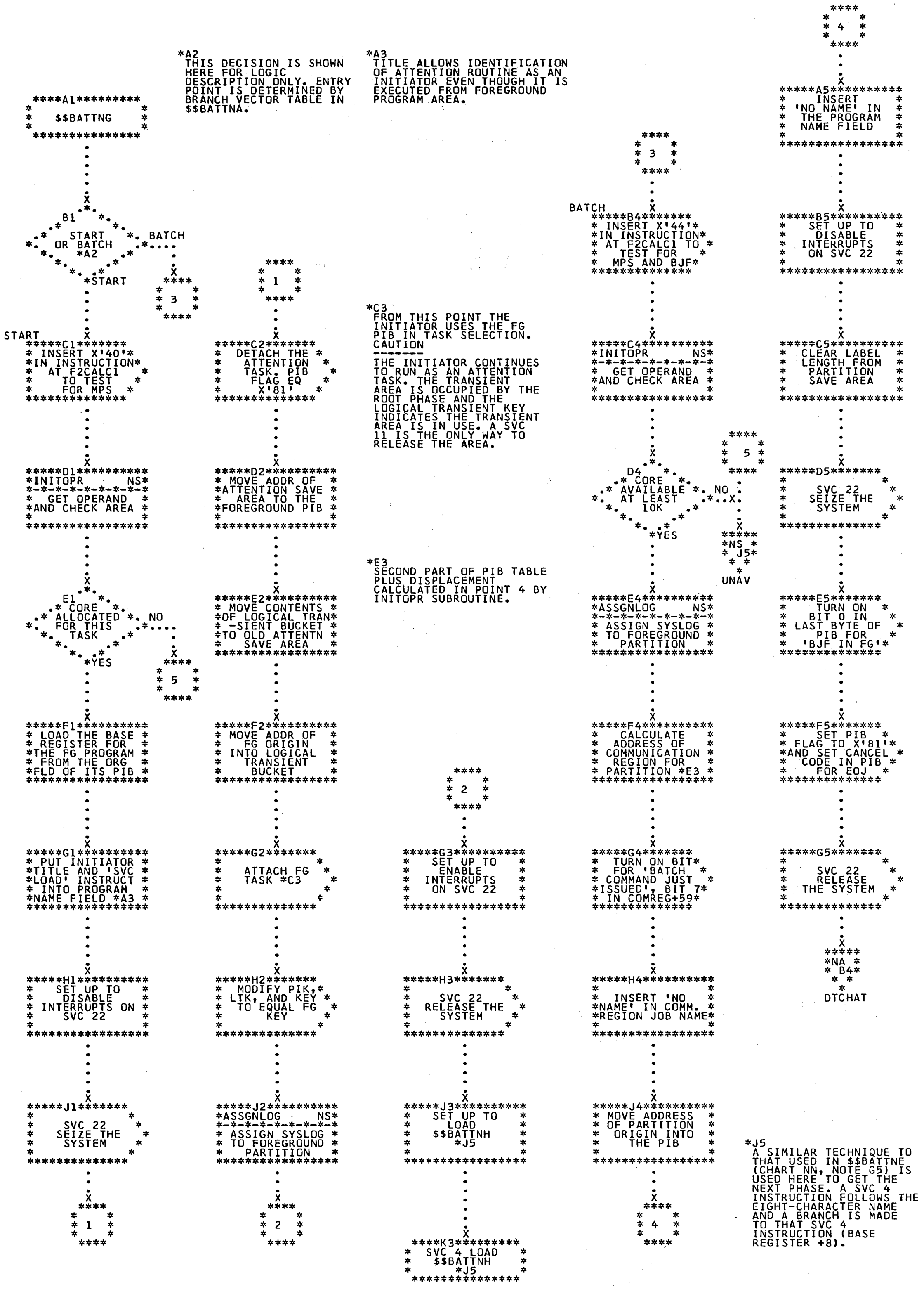
*D3
CHECK SYSTEM CONFIG BYTE IN COMREG BIT 5. IF ON, THERE ARE THREE COMREGS.

*E3
WRKRG4 IS INITIALIZED TO X'20' FOR F1. IF NO BATCH IN FOREGROUND WRKRG4 IS SET TO 0 FOR BG ONLY.

*J1
DETERMINED BY REPETITIVE ADDITIONS OF 2048 TO THE LOWER LIMIT UNTIL IT IS GREATER THAN THE UPPER LIMIT OF AN AREA.

*K1
RETURN IS MADE BY LOADING THE OLD PC PSW IN WHICH THE ADDRESS OF 'CONTROL' HAS BEEN STORED.

Chart NR. $$BATTNG - START and BATCH Statement Processors Refer to Chart 12.



A1 $$BATTNG
B1 START OR BATCH *A2
BATCH
START
3
START
C1 INSERT X'40' IN INSTRUCTION AT F2CALC1 TO TEST FOR MPS
D1 INITOPR NS GET OPERAND AND CHECK AREA
E1 CORE ALLOCATED FOR THIS TASK
NO
YES
5
F1 LOAD THE BASE REGISTER FOR THE FG PROGRAM FROM THE ORG FLD OF ITS PIB
G1 PUT INITIATOR TITLE AND 'SVC LOAD' INSTRUCT INTO PROGRAM NAME FIELD *A3
H1 SET UP TO DISABLE INTERRUPTS ON SVC 22
J1 SVC 22 SEIZE THE SYSTEM
1
1
C2 DETACH THE ATTENTION TASK. PIB FLAG EQ X'81'
D2 MOVE ADDR OF ATTENTION SAVE AREA TO THE FOREGROUND PIB
E2 MOVE CONTENTS OF LOGICAL TRAN-SIENT BUCKET TO OLD ATTENTN SAVE AREA
F2 MOVE ADDR OF FG ORIGIN INTO LOGICAL TRANSIENT BUCKET
G2 ATTACH FG TASK *C3
H2 MODIFY PIK, LTK, AND KEY TO EQUAL FG KEY
J2 ASSGNLOG NS ASSIGN SYSLOG TO FOREGROUND PARTITION
2
2
G3 SET UP TO ENABLE INTERRUPTS ON SVC 22
H3 SVC 22 RELEASE THE SYSTEM
J3 SET UP TO LOAD $$BATTNH *J5
K3 SVC 4 LOAD $$BATTNH *J5
3
BATCH
B4 INSERT X'44' IN INSTRUCTION AT F2CALC1 TO TEST FOR MPS AND BJF
C4 INITOPR NS GET OPERAND AND CHECK AREA
D4 CORE AVAILABLE AT LEAST 10K
NO
YES
5
NS J5
UNAV
E4 ASSGNLOG NS ASSIGN SYSLOG TO FOREGROUND PARTITION
F4 CALCULATE ADDRESS OF COMMUNICATION REGION FOR PARTITION *E3
G4 TURN ON BIT FOR 'BATCH COMMAND JUST ISSUED', BIT 7 IN COMREG+59
H4 INSERT 'NO NAME' IN COMM. REGION JOB NAME
J4 MOVE ADDRESS OF PARTITION ORIGIN INTO THE PIB
4
4
A5 INSERT 'NO NAME' IN THE PROGRAM NAME FIELD
B5 SET UP TO DISABLE INTERRUPTS ON SVC 22
C5 CLEAR LABEL LENGTH FROM PARTITION SAVE AREA
D5 SVC 22 SEIZE THE SYSTEM
E5 TURN ON BIT 0 IN LAST BYTE OF PIB FOR 'BJF IN FG'
F5 SET PIB FLAG TO X'81' AND SET CANCEL CODE IN PIB FOR EOJ
G5 SVC 22 RELEASE THE SYSTEM
NA B4
DTCHAT

*A2
THIS DECISION IS SHOWN HERE FOR LOGIC DESCRIPTION ONLY. ENTRY POINT IS DETERMINED BY BRANCH VECTOR TABLE IN $$BATTNA.

*A3
TITLE ALLOWS IDENTIFICATION OF ATTENTION ROUTINE AS AN INITIATOR EVEN THOUGH IT IS EXECUTED FROM FOREGROUND PROGRAM AREA.

*C3
FROM THIS POINT THE INITIATOR USES THE FG PIB IN TASK SELECTION.

CAUTION

THE INITIATOR CONTINUES TO RUN AS AN ATTENTION TASK. THE TRANSIENT AREA IS OCCUPIED BY THE ROOT PHASE AND THE LOGICAL TRANSIENT KEY INDICATES THE TRANSIENT AREA IS IN USE. A SVC 11 IS THE ONLY WAY TO RELEASE THE AREA.

*E3
SECOND PART OF PIB TABLE PLUS DISPLACEMENT CALCULATED IN POINT 4 BY INITOPR SUBROUTINE.

*J5
A SIMILAR TECHNIQUE TO THAT USED IN $$BATTNE (CHART NN, NOTE G5) IS USED HERE TO GET THE NEXT PHASE. A SVC 4 INSTRUCTION FOLLOWS THE EIGHT-CHARACTER NAME AND A BRANCH IS MADE TO THAT SVC 4 INSTRUCTION (BASE REGISTER +8).

Chart NS. $$BATTNG - START and BATCH Subroutines
Refer to Chart 12.

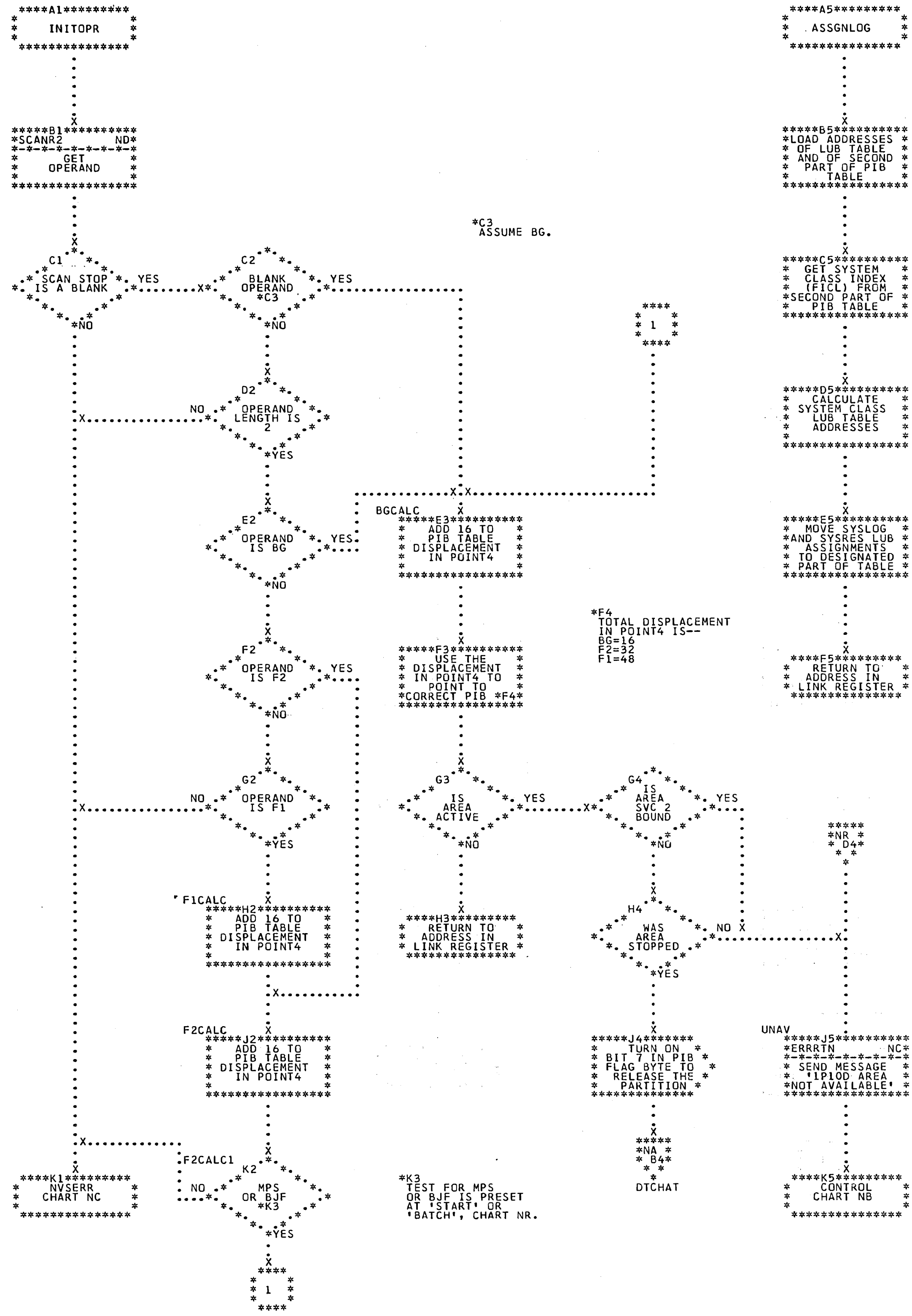
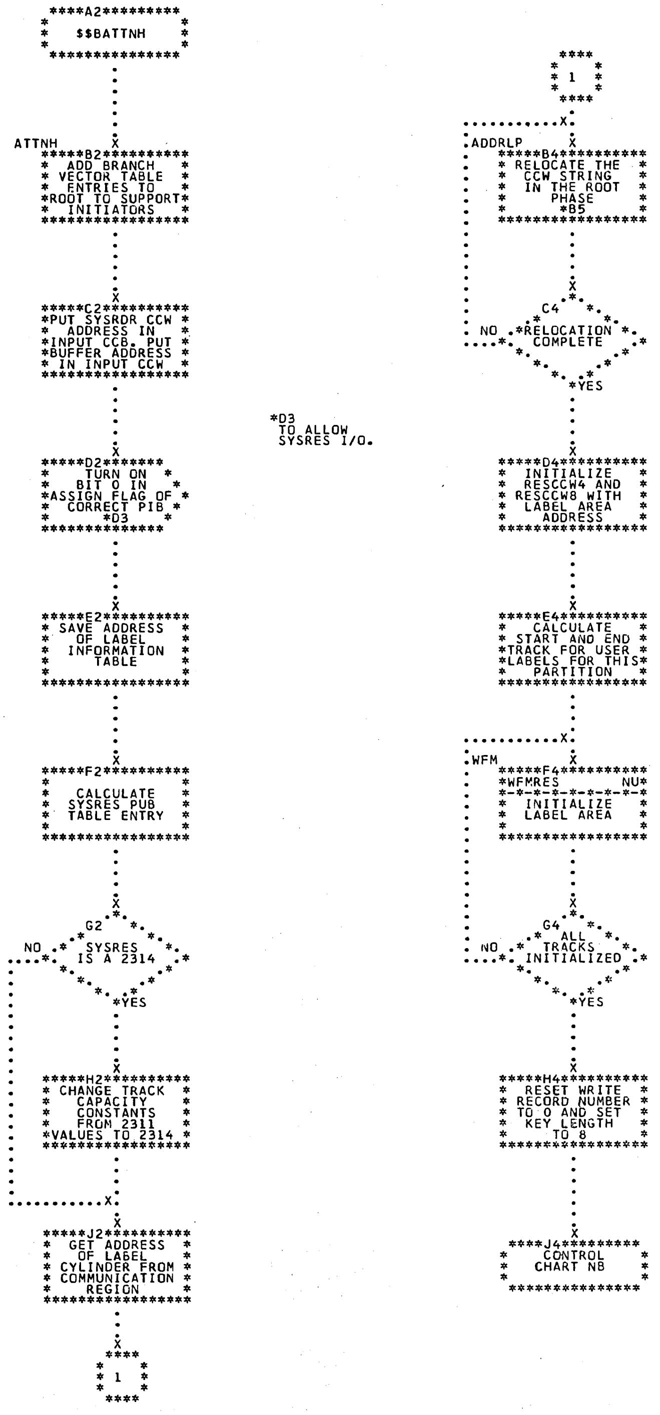

A1
INITOPR
B1
SCANR2 ND
GET OPERAND
C1
SCAN STOP IS A BLANK
YES
NO
C2
BLANK OPERAND
*C3
YES
NO
*C3
ASSUME BG.
D2
OPERAND LENGTH IS 2
NO
YES
1
E2
OPERAND IS BG
YES
NO
BGCALC
E3
ADD 16 TO PIB TABLE DISPLACEMENT IN POINT4
F2
OPERAND IS F2
YES
NO
F3
USE THE DISPLACEMENT IN POINT4 TO POINT TO CORRECT PIB *F4
*F4
TOTAL DISPLACEMENT IN POINT4 IS--
BG=16
F2=32
F1=48
G2
OPERAND IS F1
NO
YES
G3
IS AREA ACTIVE
YES
NO
G4
IS AREA SVC 2 BOUND
YES
NO
F1CALC
H2
ADD 16 TO PIB TABLE DISPLACEMENT IN POINT4
H3
RETURN TO ADDRESS IN LINK REGISTER
H4
WAS AREA STOPPED
NO
YES
F2CALC
J2
ADD 16 TO PIB TABLE DISPLACEMENT IN POINT4
J4
TURN ON BIT 7 IN PIB FLAG BYTE TO RELEASE THE PARTITION
NA
B4
DTCHAT
K1
NVSERR
CHART NC
F2CALC1
K2
MPS OR BJF
*K3
NO
YES
1
*K3
TEST FOR MPS OR BJF IS PRESET AT 'START' OR 'BATCH', CHART NR.
A5
ASSGNLOG
B5
LOAD ADDRESSES OF LUB TABLE AND OF SECOND PART OF PIB TABLE
C5
GET SYSTEM CLASS INDEX (FICL) FROM SECOND PART OF PIB TABLE
D5
CALCULATE SYSTEM CLASS LUB TABLE ADDRESSES
E5
MOVE SYSLOG AND SYSRES LUB ASSIGNMENTS TO DESIGNATED PART OF TABLE
F5
RETURN TO ADDRESS IN LINK REGISTER
NR
D4
UNAV
J5
ERRRTN NC
SEND MESSAGE '1P1OD AREA NOT AVAILABLE'
K5
CONTROL
CHART NB

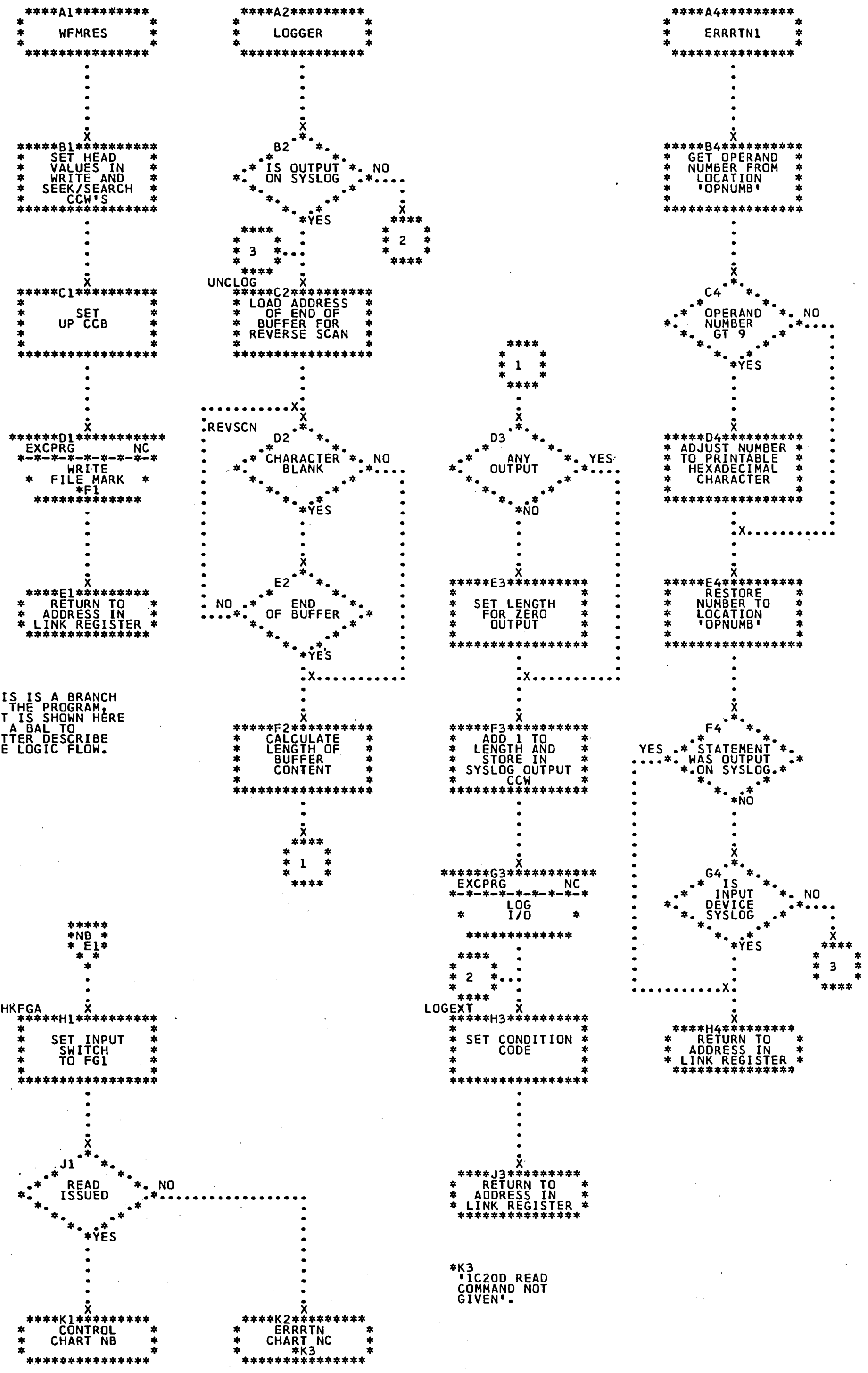
Chart NT. $$BATTNH - START Statement Processor Channel Program
Refer to Chart 12.


A2
$$BATTNH
ATTNH
B2
ADD BRANCH VECTOR TABLE ENTRIES TO ROOT TO SUPPORT INITIATORS
C2
PUT SYSRDR CCW ADDRESS IN INPUT CCB. PUT BUFFER ADDRESS IN INPUT CCW
D2
TURN ON BIT 0 IN ASSIGN FLAG OF CORRECT PIB
*D3
E2
SAVE ADDRESS OF LABEL INFORMATION TABLE
F2
CALCULATE SYSRES PUB TABLE ENTRY
G2
SYSRES IS A 2314
NO
YES
H2
CHANGE TRACK CAPACITY CONSTANTS FROM 2311 VALUES TO 2314
J2
GET ADDRESS OF LABEL CYLINDER FROM COMMUNICATION REGION
1
1
ADDRLP
B4
RELOCATE THE CCW STRING IN THE ROOT PHASE
*B5
C4
RELOCATION COMPLETE
NO
YES
D4
INITIALIZE RESCCW4 AND RESCCW8 WITH LABEL AREA ADDRESS
E4
CALCULATE START AND END TRACK FOR USER LABELS FOR THIS PARTITION
WFM
F4
WFMRES
NU
INITIALIZE LABEL AREA
G4
ALL TRACKS INITIALIZED
NO
YES
H4
RESET WRITE RECORD NUMBER TO 0 AND SET KEY LENGTH TO 8
J4
CONTROL CHART NB

*D3
TO ALLOW
SYSRES I/O.

*B5
ADD ROOT PHASE
BASE REGISTER
VALUE TO CCW
ADDRESS. RELOCATE
9 CCW'S (RESCCW,
RESCCW1-RESCCW8).

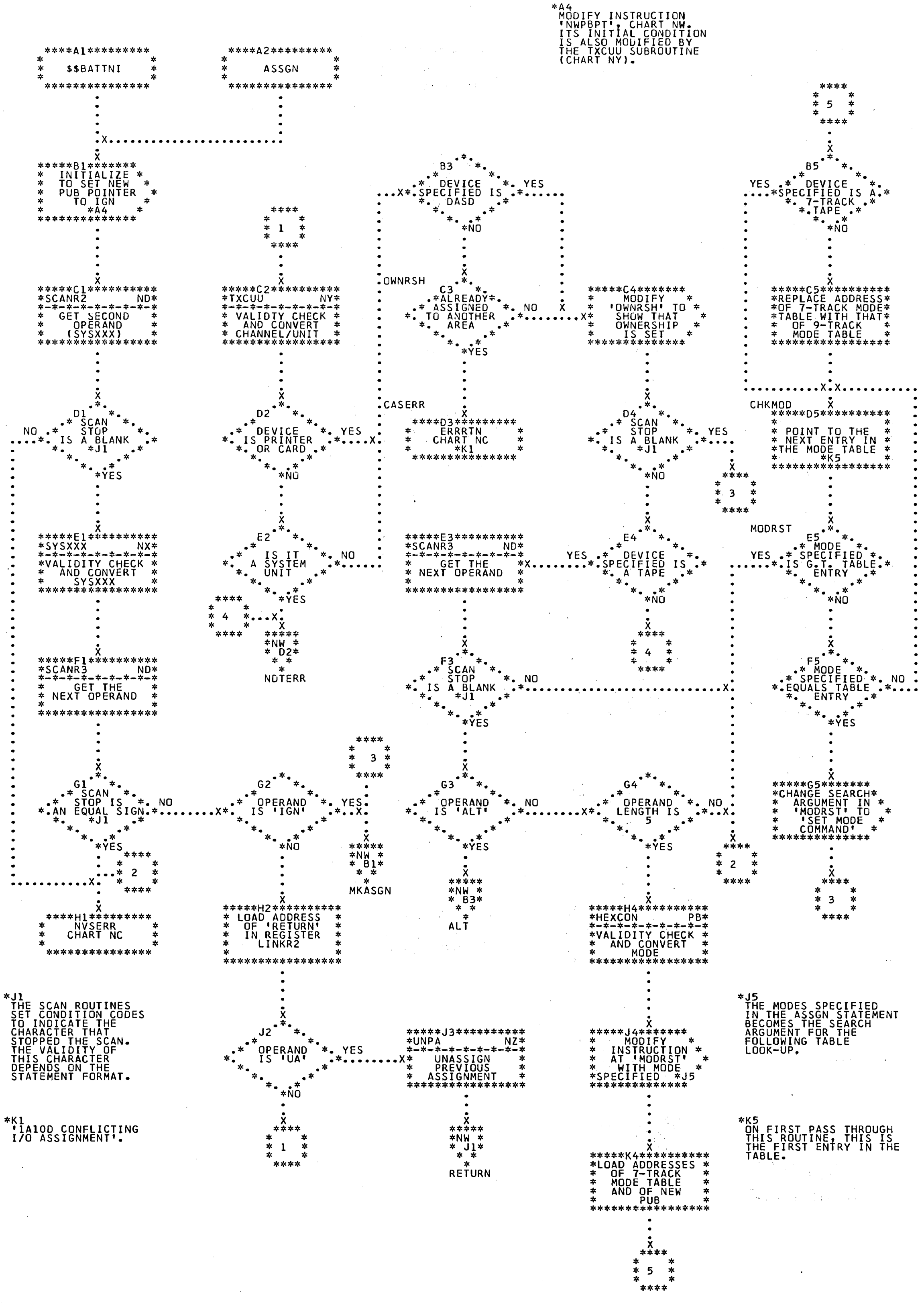
Chart NU. $$BATTNH - Subroutines
Refer to Chart 12.


A1 WFMRES
B1 SET HEAD VALUES IN WRITE AND SEEK/SEARCH CCW'S
C1 SET UP CCB
D1 EXCPRG NC WRITE FILE MARK *F1
E1 RETURN TO ADDRESS IN LINK REGISTER
NB E1
CHKFGA
H1 SET INPUT SWITCH TO FG1
J1 READ ISSUED
NO
YES
K1 CONTROL CHART NB
A2 LOGGER
B2 IS OUTPUT ON SYSLOG
NO
YES
2
3
UNCLOG
C2 LOAD ADDRESS OF END OF BUFFER FOR REVERSE SCAN
REVSCN
D2 CHARACTER BLANK
NO
YES
E2 END OF BUFFER
NO
YES
F2 CALCULATE LENGTH OF BUFFER CONTENT
1
K2 ERRRTN CHART NC *K3
1
D3 ANY OUTPUT
YES
NO
E3 SET LENGTH FOR ZERO OUTPUT
F3 ADD 1 TO LENGTH AND STORE IN SYSLOG OUTPUT CCW
G3 EXCPRG NC LOG I/O
2
LOGEXT
H3 SET CONDITION CODE
J3 RETURN TO ADDRESS IN LINK REGISTER
A4 ERRRTN1
B4 GET OPERAND NUMBER FROM LOCATION 'OPNUMB'
C4 OPERAND NUMBER GT 9
NO
YES
D4 ADJUST NUMBER TO PRINTABLE HEXADECIMAL CHARACTER
E4 RESTORE NUMBER TO LOCATION 'OPNUMB'
F4 STATEMENT WAS OUTPUT ON SYSLOG
YES
NO
G4 IS INPUT DEVICE SYSLOG
NO
YES
3
H4 RETURN TO ADDRESS IN LINK REGISTER

*F1
THIS IS A BRANCH IN THE PROGRAM, BUT IS SHOWN HERE AS A BAL TO BETTER DESCRIBE THE LOGIC FLOW.

*K3
'1C20D READ COMMAND NOT GIVEN'.

Chart NV. $$BATTNI - ASSGN Statement Processor (Part 1 of 2)
Refer to Chart 13.


A1 $$BATTNI
A2 ASSGN
B1 INITIALIZE TO SET NEW PUB POINTER TO IGN *A4
C1 SCANR2 ND GET SECOND OPERAND (SYSXXX)
D1 SCAN STOP IS A BLANK *J1
NO
YES
E1 SYSXXX NX VALIDITY CHECK AND CONVERT SYSXXX
F1 SCANR3 ND GET THE NEXT OPERAND
G1 SCAN STOP IS AN EQUAL SIGN *J1
NO
YES
2
H1 NVSERR CHART NC
1
C2 TXCUU NY VALIDTY CHECK AND CONVERT CHANNEL/UNIT
D2 DEVICE IS PRINTER OR CARD
YES
NO
E2 IS IT A SYSTEM UNIT
NO
YES
4
NW D2
NOTERR
3
G2 OPERAND IS 'IGN'
YES
NO
NW B1
MKASGN
H2 LOAD ADDRESS OF 'RETURN' IN REGISTER LINKR2
J2 OPERAND IS 'UA'
YES
NO
1
B3 DEVICE SPECIFIED IS DASD
YES
NO
OWNRSH
C3 ALREADY ASSIGNED TO ANOTHER AREA
NO
YES
CASERR
D3 ERRRTN CHART NC *K1
E3 SCANR3 ND GET THE NEXT OPERAND
F3 SCAN STOP IS A BLANK *J1
NO
YES
G3 OPERAND IS 'ALT'
NO
YES
NW B3
ALT
J3 UNPA NZ UNASSIGN PREVIOUS ASSIGNMENT
NW J1
RETURN
C4 MODIFY 'OWNRSH' TO SHOW THAT OWNERSHIP IS SET
D4 SCAN STOP IS A BLANK *J1
YES
NO
3
E4 DEVICE SPECIFIED IS A TAPE
YES
NO
4
G4 OPERAND LENGTH IS 5
NO
YES
2
H4 HEXCON PB VALIDITY CHECK AND CONVERT MODE
J4 MODIFY INSTRUCTION AT 'MODRST' WITH MODE SPECIFIED *J5
K4 LOAD ADDRESSES OF 7-TRACK MODE TABLE AND OF NEW PUB
5
5
B5 DEVICE SPECIFIED IS A 7-TRACK TAPE
YES
NO
C5 REPLACE ADDRESS OF 7-TRACK MODE TABLE WITH THAT OF 9-TRACK MODE TABLE
CHKMOD
D5 POINT TO THE NEXT ENTRY IN THE MODE TABLE *K5
MODRST
E5 MODE SPECIFIED IS G.T. TABLE ENTRY
YES
NO
F5 MODE SPECIFIED EQUALS TABLE ENTRY
NO
YES
G5 CHANGE SEARCH ARGUMENT IN 'MODRST' TO 'SET MODE COMMAND'
3

*A4
MODIFY INSTRUCTION 'NWPBPT', CHART NW. ITS INITIAL CONDITION IS ALSO MODIFIED BY THE TXCUU SUBROUTINE (CHART NY).

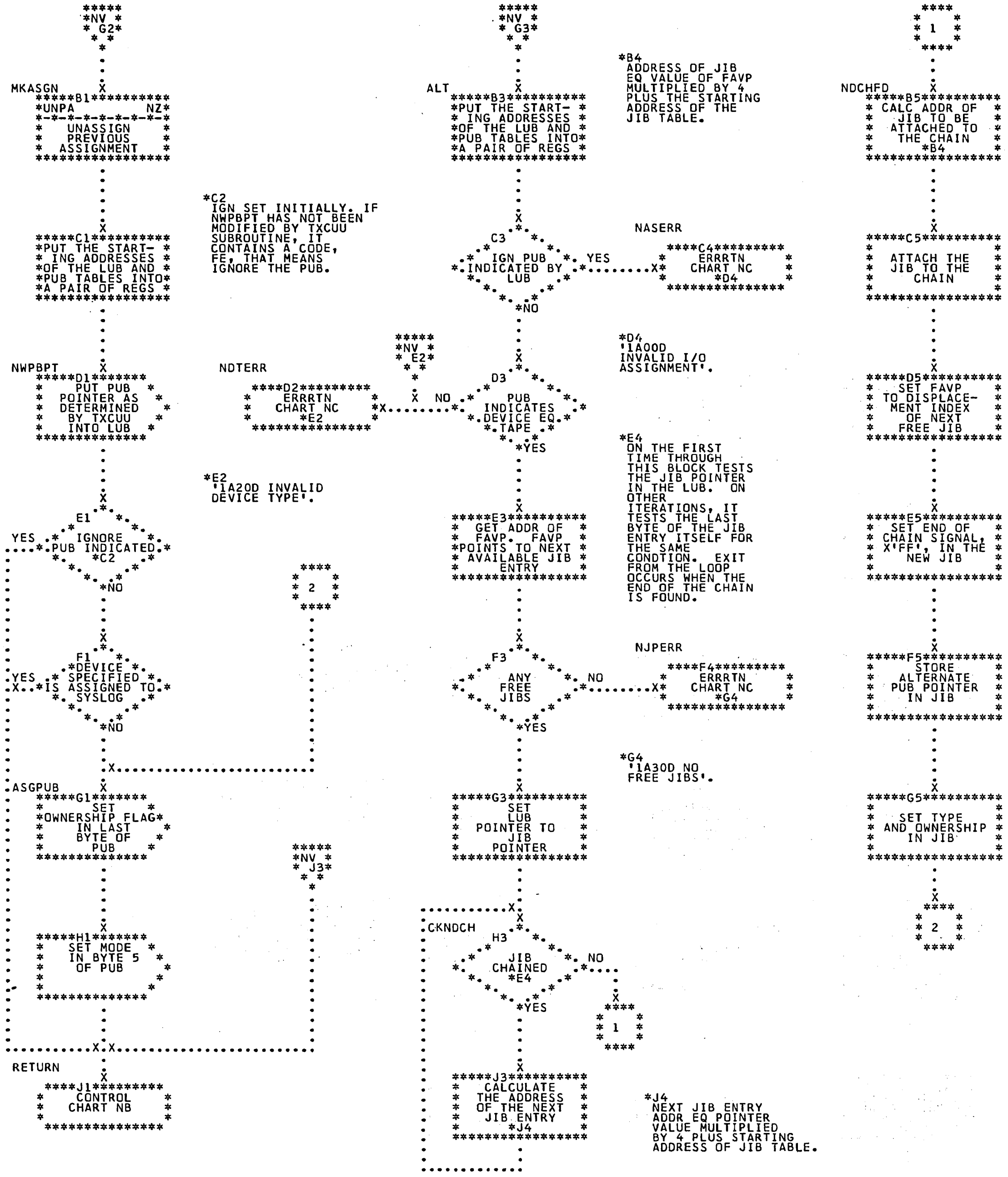
*J1
THE SCAN ROUTINES SET CONDITION CODES TO INDICATE THE CHARACTER THAT STOPPED THE SCAN. THE VALIDITY OF THIS CHARACTER DEPENDS ON THE STATEMENT FORMAT.

*K1
'1A10D CONFLICTING I/O ASSIGNMENT'.

*J5
THE MODES SPECIFIED IN THE ASSGN STATEMENT BECOMES THE SEARCH ARGUMENT FOR THE FOLLOWING TABLE LOOK-UP.

*K5
ON FIRST PASS THROUGH THIS ROUTINE, THIS IS THE FIRST ENTRY IN THE TABLE.

Chart NW. $$BATTNI - ASSGN Statement Processor (Part 2 of 2)
Refer to Chart 13.


NV G2
MKASGN
B1 UNPA NZ
UNASSIGN PREVIOUS ASSIGNMENT
C1
PUT THE START- ING ADDRESSES OF THE LUB AND PUB TABLES INTO A PAIR OF REGS
NWPBPT
D1
PUT PUB POINTER AS DETERMINED BY TXCUU INTO LUB
E1
IGNORE PUB INDICATED C2
YES
NO
F1
DEVICE SPECIFIED IS ASSIGNED TO SYSLOG
YES
NO
ASGPUB
G1
SET OWNERSHIP FLAG IN LAST BYTE OF PUB
H1
SET MODE IN BYTE 5 OF PUB
RETURN
J1
CONTROL CHART NB
NDTERR
D2
ERRRTN CHART NC E2
2
NV J3
NV G3
ALT
B3
PUT THE START- ING ADDRESSES OF THE LUB AND PUB TABLES INTO A PAIR OF REGS
C3
IGN PUB INDICATED BY LUB
YES
NO
NASERR
C4
ERRRTN CHART NC D4
NV E2
D3
PUB INDICATES DEVICE EQ. TAPE
NO
YES
E3
GET ADDR OF FAVP. FAVP POINTS TO NEXT AVAILABLE JIB ENTRY
F3
ANY FREE JIBS
NO
YES
NJPERR
F4
ERRRTN CHART NC G4
G3
SET LUB POINTER TO JIB POINTER
CKNDCH
H3
JIB CHAINED E4
NO
YES
1
J3
CALCULATE THE ADDRESS OF THE NEXT JIB ENTRY J4
1
NDCHFD
B5
CALC ADDR OF JIB TO BE ATTACHED TO THE CHAIN B4
C5
ATTACH THE JIB TO THE CHAIN
D5
SET FAVP TO DISPLACE- MENT INDEX OF NEXT FREE JIB
E5
SET END OF CHAIN SIGNAL, X'FF', IN THE NEW JIB
F5
STORE ALTERNATE PUB POINTER IN JIB
G5
SET TYPE AND OWNERSHIP IN JIB
2

*C2
IGN SET INITIALLY. IF NWPBPT HAS NOT BEEN MODIFIED BY TXCUU SUBROUTINE, IT CONTAINS A CODE, FE, THAT MEANS IGNORE THE PUB.

*E2
'1A20D INVALID DEVICE TYPE'.

*B4
ADDRESS OF JIB EQ VALUE OF FAVP MULTIPLIED BY 4 PLUS THE STARTING ADDRESS OF THE JIB TABLE.

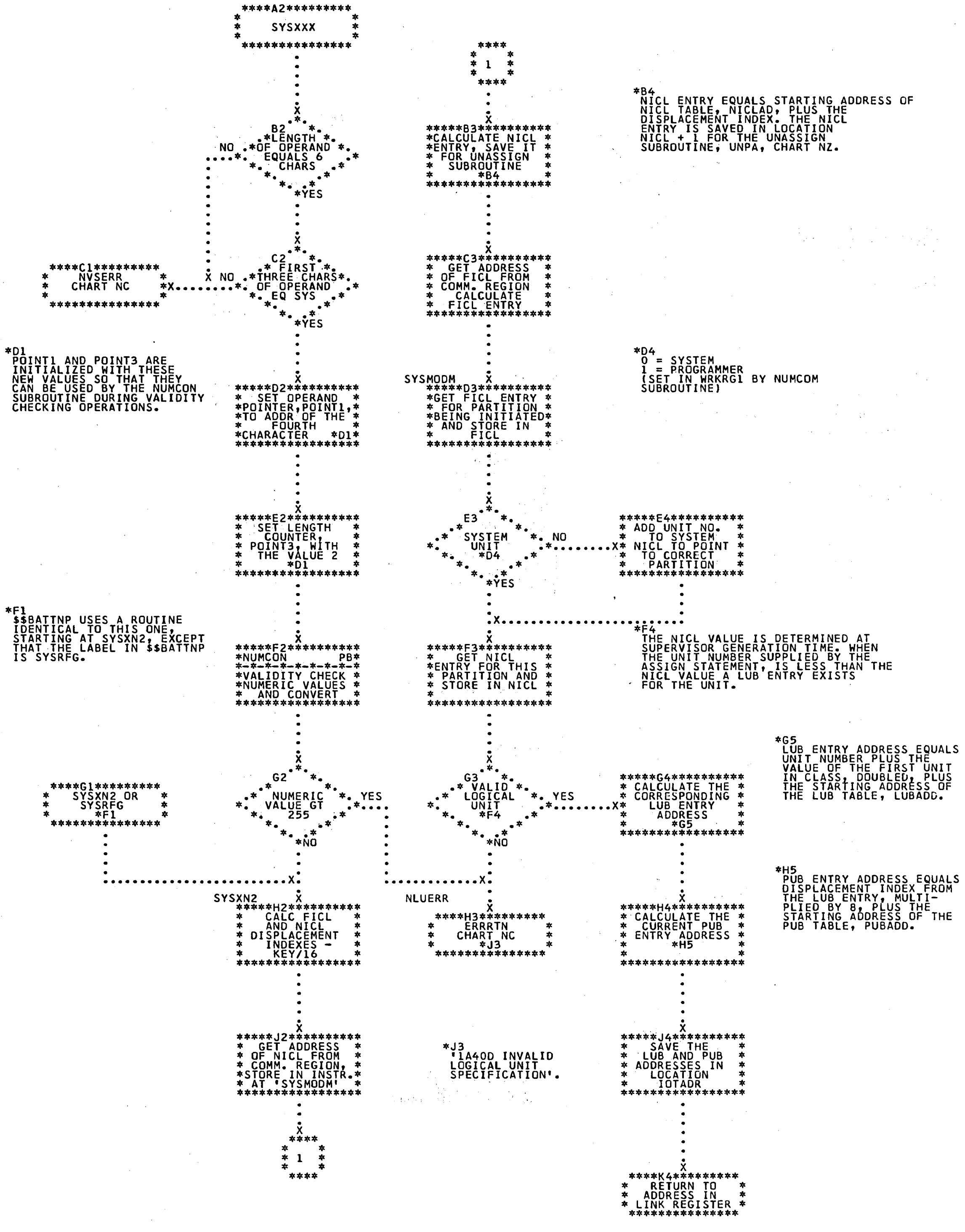
*D4
'1A00D INVALID I/O ASSIGNMENT'.

*E4
ON THE FIRST TIME THROUGH THIS BLOCK TESTS THE JIB POINTER IN THE LUB. ON OTHER ITERATIONS, IT TESTS THE LAST BYTE OF THE JIB ENTRY ITSELF FOR THE SAME CONDTION. EXIT FROM THE LOOP OCCURS WHEN THE END OF THE CHAIN IS FOUND.

*G4
'1A30D NO FREE JIBS'.

*J4
NEXT JIB ENTRY ADDR EQ POINTER VALUE MULTIPLIED BY 4 PLUS STARTING ADDRESS OF JIB TABLE.

Chart NX. $$BATTNI - Validate SYSXXX Subroutine
Refer to Chart 13.


A2
SYSXXX
B2
LENGTH OF OPERAND EQUALS 6 CHARS
NO
YES
C1
NVSERR CHART NC
C2
FIRST THREE CHARS OF OPERAND EQ SYS
NO
YES
D2
SET OPERAND POINTER,POINT1, TO ADDR OF THE FOURTH CHARACTER *D1
E2
SET LENGTH COUNTER, POINT3, WITH THE VALUE 2 *D1
F2
NUMCON PB
VALIDITY CHECK NUMERIC VALUES AND CONVERT
G1
SYSXN2 OR SYSRFG *F1
G2
NUMERIC VALUE GT 255
YES
NO
SYSXN2
H2
CALC FICL AND NICL DISPLACEMENT INDEXES - KEY/16
J2
GET ADDRESS OF NICL FROM COMM. REGION, STORE IN INSTR. AT 'SYSMODM'
1
1
B3
CALCULATE NICL ENTRY, SAVE IT FOR UNASSIGN SUBROUTINE *B4
C3
GET ADDRESS OF FICL FROM COMM. REGION CALCULATE FICL ENTRY
SYSMODM
D3
GET FICL ENTRY FOR PARTITION BEING INITIATED AND STORE IN FICL
E3
SYSTEM UNIT *D4
NO
YES
E4
ADD UNIT NO. TO SYSTEM NICL TO POINT TO CORRECT PARTITION
F3
GET NICL ENTRY FOR THIS PARTITION AND STORE IN NICL
G3
VALID LOGICAL UNIT *F4
YES
NO
G4
CALCULATE THE CORRESPONDING LUB ENTRY ADDRESS *G5
NLUERR
H3
ERRRTN CHART NC *J3
H4
CALCULATE THE CURRENT PUB ENTRY ADDRESS *H5
J4
SAVE THE LUB AND PUB ADDRESSES IN LOCATION IOTADR
K4
RETURN TO ADDRESS IN LINK REGISTER

*B4
NICL ENTRY EQUALS STARTING ADDRESS OF NICL TABLE, NICLAD, PLUS THE DISPLACEMENT INDEX. THE NICL ENTRY IS SAVED IN LOCATION NICL + 1 FOR THE UNASSIGN SUBROUTINE, UNPA, CHART NZ.

*D1
POINT1 AND POINT3 ARE INITIALIZED WITH THESE NEW VALUES SO THAT THEY CAN BE USED BY THE NUMCON SUBROUTINE DURING VALIDITY CHECKING OPERATIONS.

*D4
0 = SYSTEM
1 = PROGRAMMER
(SET IN WRKRG1 BY NUMCOM SUBROUTINE)

*F1
$$BATTNP USES A ROUTINE IDENTICAL TO THIS ONE, STARTING AT SYSXN2, EXCEPT THAT THE LABEL IN $$BATTNP IS SYSRFG.

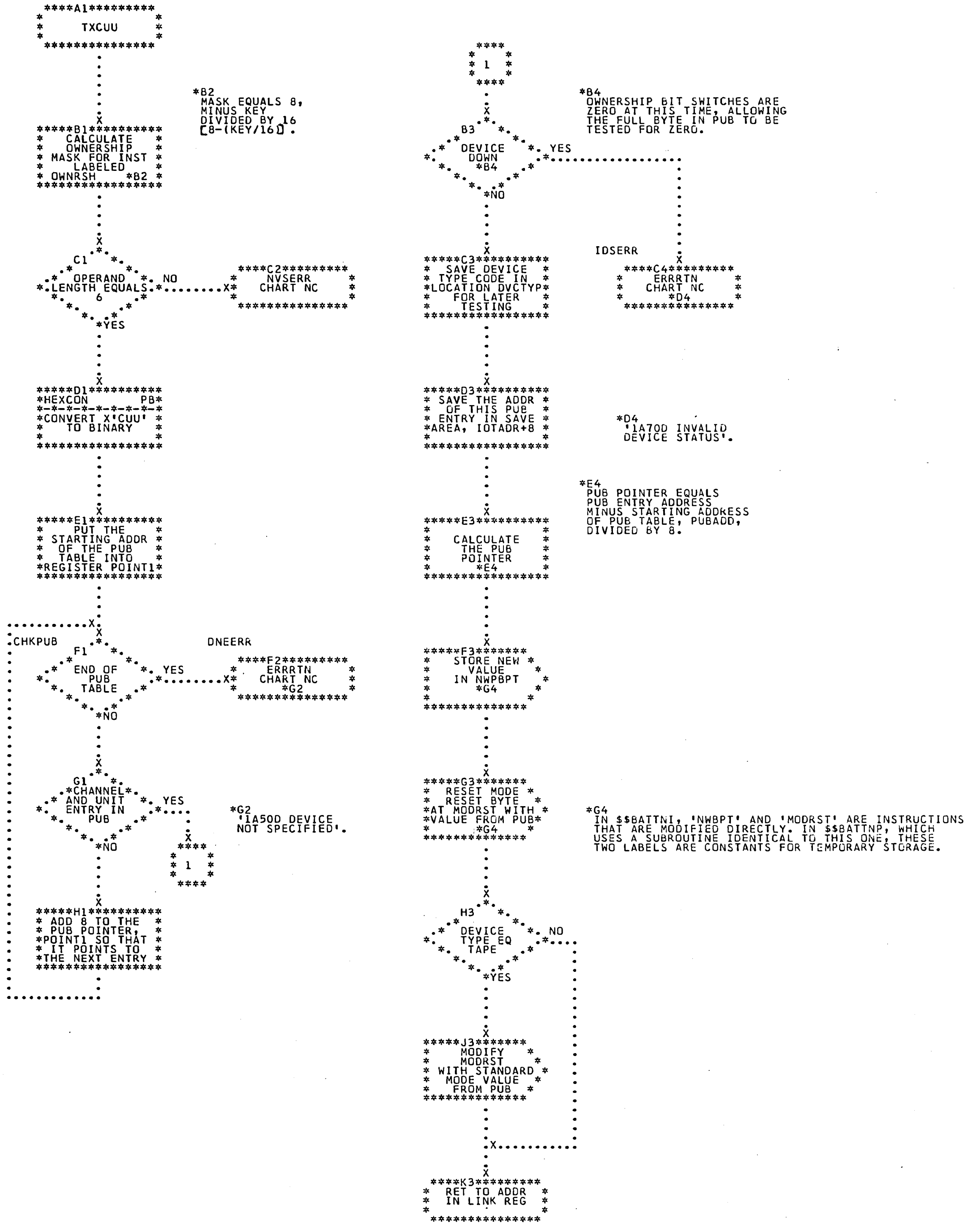
*F4
THE NICL VALUE IS DETERMINED AT SUPERVISOR GENERATION TIME. WHEN THE UNIT NUMBER SUPPLIED BY THE ASSIGN STATEMENT, IS LESS THAN THE NICL VALUE A LUB ENTRY EXISTS FOR THE UNIT.

*G5
LUB ENTRY ADDRESS EQUALS UNIT NUMBER PLUS THE VALUE OF THE FIRST UNIT IN CLASS, DOUBLED, PLUS THE STARTING ADDRESS OF THE LUB TABLE, LUBADD.

*H5
PUB ENTRY ADDRESS EQUALS DISPLACEMENT INDEX FROM THE LUB ENTRY, MULTIPLIED BY 8, PLUS THE STARTING ADDRESS OF THE PUB TABLE, PUBADD.

*J3
'1A40D INVALID LOGICAL UNIT SPECIFICATION'.

Chart NY. $$BATTNI - Validity Check Channel and Unit
Refer to Chart 13.


A1 TXCUU
B1 CALCULATE OWNERSHIP MASK FOR INST LABELED OWNRSH *B2
C1 OPERAND LENGTH EQUALS 6
NO
YES
C2 NVSERR CHART NC
D1 HEXCON PB
CONVERT X'CUU' TO BINARY
E1 PUT THE STARTING ADDR OF THE PUB TABLE INTO REGISTER POINT1
CHKPUB
F1 END OF PUB TABLE
YES
NO
DNEERR
F2 ERRRTN CHART NC *G2
G1 CHANNEL AND UNIT ENTRY IN PUB
YES
NO
1
H1 ADD 8 TO THE PUB POINTER, POINT1 SO THAT IT POINTS TO THE NEXT ENTRY
1
B3 DEVICE DOWN *B4
YES
NO
IDSERR
C4 ERRRTN CHART NC *D4
C3 SAVE DEVICE TYPE CODE IN LOCATION DVCTYP FOR LATER TESTING
D3 SAVE THE ADDR OF THIS PUB ENTRY IN SAVE AREA, IOTADR+8
E3 CALCULATE THE PUB POINTER *E4
F3 STORE NEW VALUE IN NWPBPT *G4
G3 RESET MODE RESET BYTE AT MODRST WITH VALUE FROM PUB *G4
H3 DEVICE TYPE EQ TAPE
NO
YES
J3 MODIFY MODRST WITH STANDARD MODE VALUE FROM PUB
K3 RET TO ADDR IN LINK REG

*B2
MASK EQUALS 8,
MINUS KEY
DIVIDED BY 16
[8-(KEY/16] .

*B4
OWNERSHIP BIT SWITCHES ARE
ZERO AT THIS TIME, ALLOWING
THE FULL BYTE IN PUB TO BE
TESTED FOR ZERO.

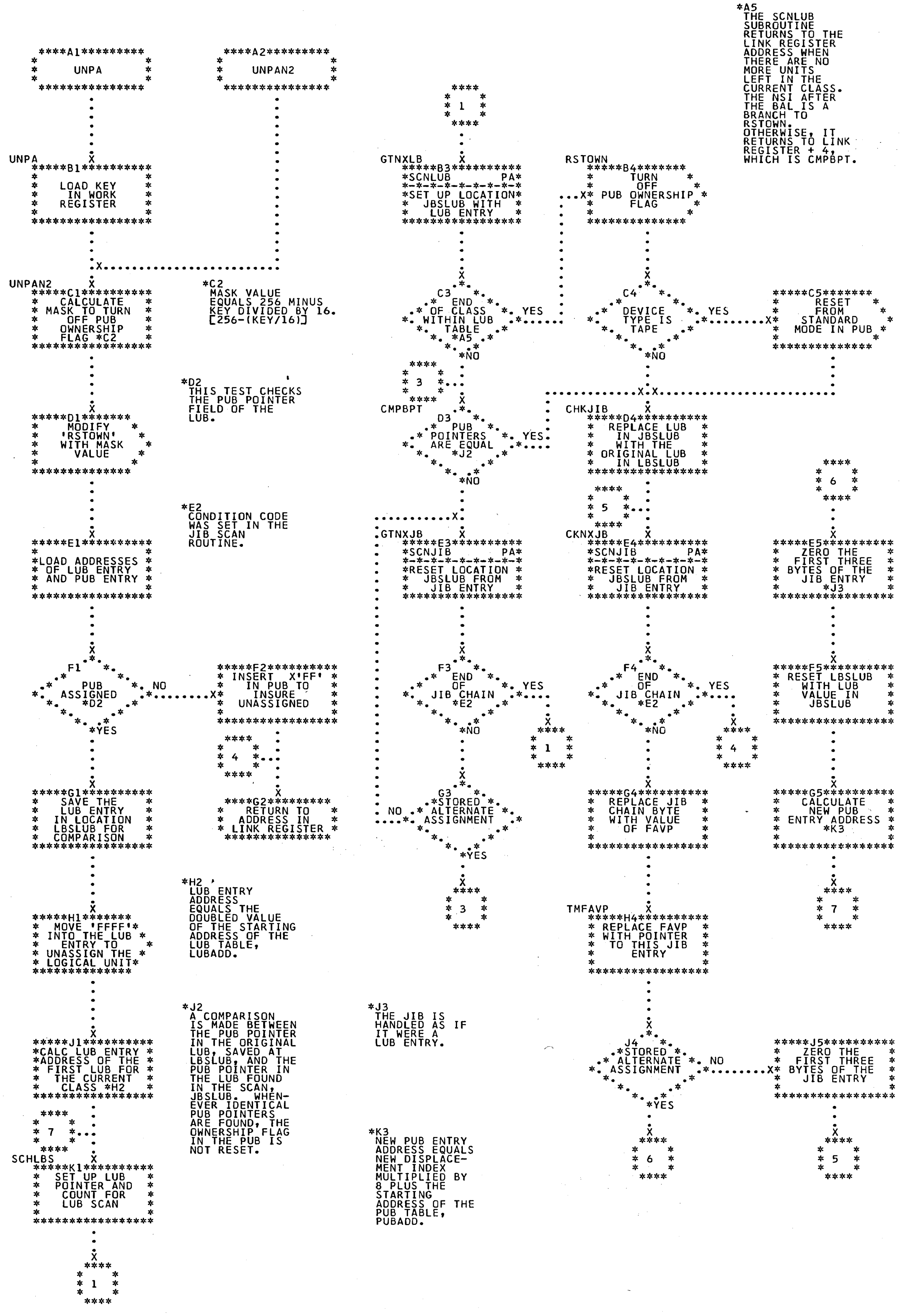
*D4
'1A70D INVALID
DEVICE STATUS'.

*E4
PUB POINTER EQUALS
PUB ENTRY ADDRESS
MINUS STARTING ADDRESS
OF PUB TABLE, PUBADD,
DIVIDED BY 8.

*G2
'1A50D DEVICE
NOT SPECIFIED'.

*G4
IN $$BATTNI, 'NWBPT' AND 'MODRST' ARE INSTRUCTIONS
THAT ARE MODIFIED DIRECTLY. IN $$BATTNP, WHICH
USES A SUBROUTINE IDENTICAL TO THIS ONE, THESE
TWO LABELS ARE CONSTANTS FOR TEMPORARY STORAGE.

Chart NZ. $$BATTNI - Unassign Subroutine
Refer to Chart 13.


A1 UNPA
A2 UNPAN2
1
UNPA
B1 LOAD KEY IN WORK REGISTER
GTNXLB
B3 SCNLUB PA
SET UP LOCATION JBSLUB WITH LUB ENTRY
RSTOWN
B4 TURN OFF PUB OWNERSHIP FLAG
UNPAN2
C1 CALCULATE MASK TO TURN OFF PUB OWNERSHIP FLAG *C2
C3 END OF CLASS WITHIN LUB TABLE *A5
YES
NO
C4 DEVICE TYPE IS TAPE
YES
NO
C5 RESET FROM STANDARD MODE IN PUB
3
CMPBPT
D3 PUB POINTERS ARE EQUAL *J2
YES
NO
CHKJIB
D4 REPLACE LUB IN JBSLUB WITH THE ORIGINAL LUB IN LBSLUB
D1 MODIFY 'RSTOWN' WITH MASK VALUE
6
5
GTNXJB
E3 SCNJIB PA
RESET LOCATION JBSLUB FROM JIB ENTRY
CKNXJB
E4 SCNJIB PA
RESET LOCATION JBSLUB FROM JIB ENTRY
E5 ZERO THE FIRST THREE BYTES OF THE JIB ENTRY *J3
E1 LOAD ADDRESSES OF LUB ENTRY AND PUB ENTRY
F1 PUB ASSIGNED *D2
NO
YES
F2 INSERT X'FF' IN PUB TO INSURE UNASSIGNED
F3 END OF JIB CHAIN *E2
YES
NO
1
F4 END OF JIB CHAIN *E2
YES
NO
4
F5 RESET LBSLUB WITH LUB VALUE IN JBSLUB
4
G1 SAVE THE LUB ENTRY IN LOCATION LBSLUB FOR COMPARISON
G2 RETURN TO ADDRESS IN LINK REGISTER
G3 STORED ALTERNATE ASSIGNMENT
NO
YES
G4 REPLACE JIB CHAIN BYTE WITH VALUE OF FAVP
G5 CALCULATE NEW PUB ENTRY ADDRESS *K3
3
TMFAVP
H4 REPLACE FAVP WITH POINTER TO THIS JIB ENTRY
7
H1 MOVE 'FFFF' INTO THE LUB ENTRY TO UNASSIGN THE LOGICAL UNIT
J1 CALC LUB ENTRY ADDRESS OF THE FIRST LUB FOR THE CURRENT CLASS *H2
J4 STORED ALTERNATE ASSIGNMENT
NO
YES
J5 ZERO THE FIRST THREE BYTES OF THE JIB ENTRY
7
SCHLBS
K1 SET UP LUB POINTER AND COUNT FOR LUB SCAN
1
6
5

*A5 THE SCNLUB SUBROUTINE RETURNS TO THE LINK REGISTER ADDRESS WHEN THERE ARE NO MORE UNITS LEFT IN THE CURRENT CLASS. THE NSI AFTER THE BAL IS A BRANCH TO RSTOWN. OTHERWISE, IT RETURNS TO LINK REGISTER + 4, WHICH IS CMPBPT.

*C2 MASK VALUE EQUALS 256 MINUS KEY DIVIDED BY 16. [256-(KEY/16)]

*D2 THIS TEST CHECKS THE PUB POINTER FIELD OF THE LUB.

*E2 CONDITION CODE WAS SET IN THE JIB SCAN ROUTINE.

*H2 LUB ENTRY ADDRESS EQUALS THE DOUBLED VALUE OF THE STARTING ADDRESS OF THE LUB TABLE, LUBADD.

*J2 A COMPARISON IS MADE BETWEEN THE PUB POINTER IN THE ORIGINAL LUB, SAVED AT LBSLUB, AND THE PUB POINTER IN THE LUB FOUND IN THE SCAN, JBSLUB. WHENEVER IDENTICAL PUB POINTERS ARE FOUND, THE OWNERSHIP FLAG IN THE PUB IS NOT RESET.

*J3 THE JIB IS HANDLED AS IF IT WERE A LUB ENTRY.

*K3 NEW PUB ENTRY ADDRESS EQUALS NEW DISPLACEMENT INDEX MULTIPLIED BY 8 PLUS THE STARTING ADDRESS OF THE PUB TABLE, PUBADD.

Chart PA. $$BATTNI - Scan LUBs and JIBs Subroutines
Refer to Chart 13.

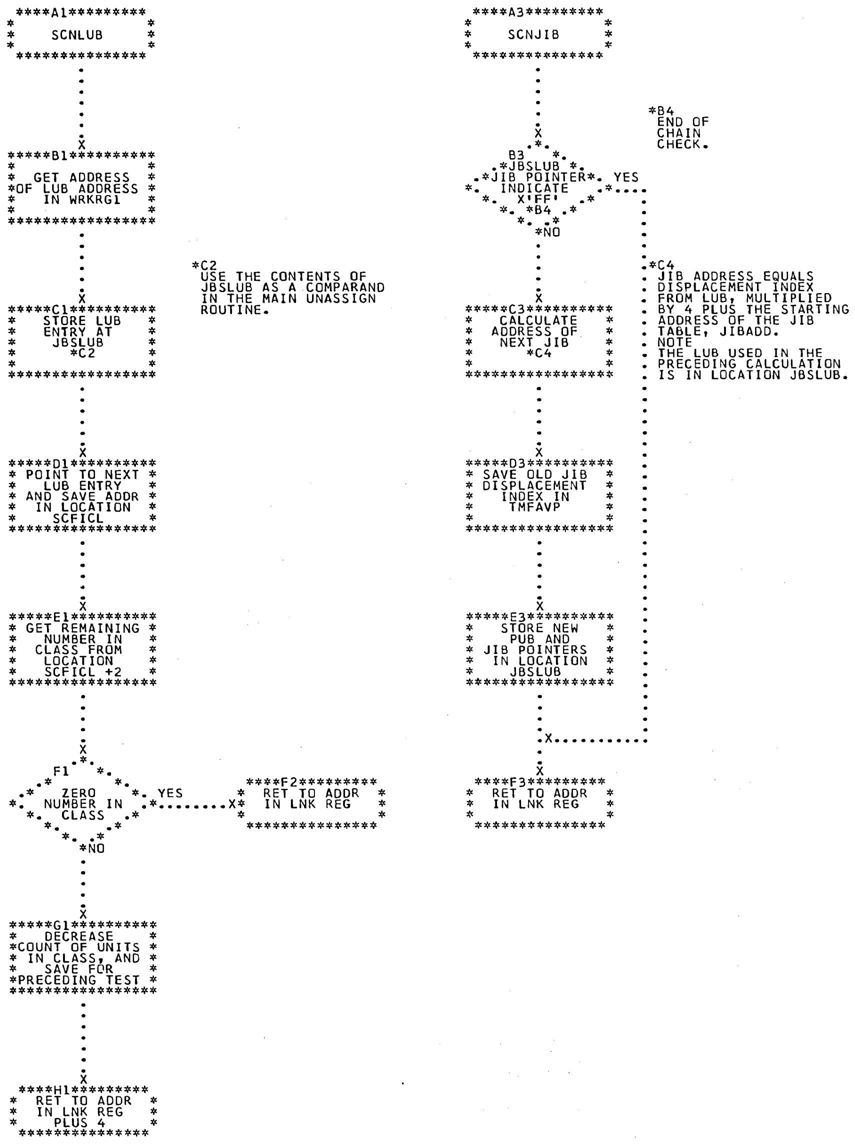
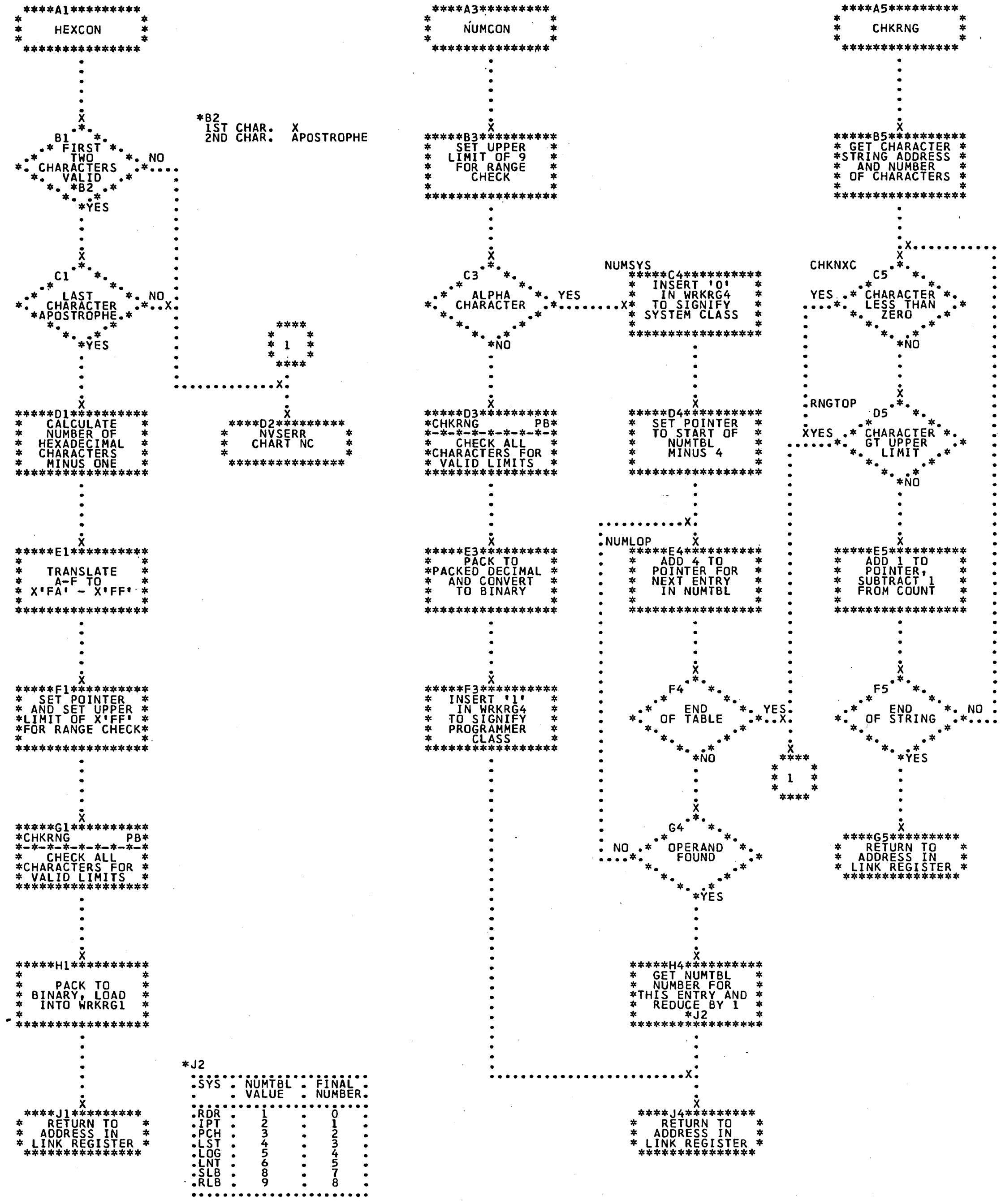

A1
SCNLUB
B1
GET ADDRESS OF LUB ADDRESS IN WRKRG1
C1
STORE LUB ENTRY AT JBSLUB *C2
*C2
USE THE CONTENTS OF JBSLUB AS A COMPARAND IN THE MAIN UNASSIGN ROUTINE.
D1
POINT TO NEXT LUB ENTRY AND SAVE ADDR IN LOCATION SCFICL
E1
GET REMAINING NUMBER IN CLASS FROM LOCATION SCFICL +2
F1
ZERO NUMBER IN CLASS
YES
NO
F2
RET TO ADDR IN LNK REG
G1
DECREASE COUNT OF UNITS IN CLASS, AND SAVE FOR PRECEDING TEST
H1
RET TO ADDR IN LNK REG PLUS 4
A3
SCNJIB
*B4
END OF CHAIN CHECK.
B3
JBSLUB JIB POINTER INDICATE X'FF' *B4
YES
NO
C3
CALCULATE ADDRESS OF NEXT JIB *C4
*C4
JIB ADDRESS EQUALS DISPLACEMENT INDEX FROM LUB, MULTIPLIED BY 4 PLUS THE STARTING ADDRESS OF THE JIB TABLE, JIBADD.
NOTE
THE LUB USED IN THE PRECEDING CALCULATION IS IN LOCATION JBSLUB.
D3
SAVE OLD JIB DISPLACEMENT INDEX IN TMFAVP
E3
STORE NEW PUB AND JIB POINTERS IN LOCATION JBSLUB
F3
RET TO ADDR IN LNK REG

Chart PB. $$BATTNI - Conversion Subroutines
Refer to Chart 13.


A1
HEXCON
B1
FIRST TWO CHARACTERS VALID
*B2
NO
YES
*B2
1ST CHAR. X
2ND CHAR. APOSTROPHE
C1
LAST CHARACTER APOSTROPHE
NO
YES
1
D1
CALCULATE NUMBER OF HEXADECIMAL CHARACTERS MINUS ONE
D2
NVSERR
CHART NC
E1
TRANSLATE A-F TO X'FA' - X'FF'
F1
SET POINTER AND SET UPPER LIMIT OF X'FF' FOR RANGE CHECK
G1
CHKRNG PB
CHECK ALL CHARACTERS FOR VALID LIMITS
H1
PACK TO BINARY, LOAD INTO WRKRG1
J1
RETURN TO ADDRESS IN LINK REGISTER
A3
NUMCON
B3
SET UPPER LIMIT OF 9 FOR RANGE CHECK
C3
ALPHA CHARACTER
YES
NO
D3
CHKRNG PB
CHECK ALL CHARACTERS FOR VALID LIMITS
E3
PACK TO PACKED DECIMAL AND CONVERT TO BINARY
F3
INSERT '1' IN WRKRG4 TO SIGNIFY PROGRAMMER CLASS
NUMSYS
C4
INSERT '0' IN WRKRG4 TO SIGNIFY SYSTEM CLASS
D4
SET POINTER TO START OF NUMTBL MINUS 4
NUMLOP
E4
ADD 4 TO POINTER FOR NEXT ENTRY IN NUMTBL
F4
END OF TABLE
YES
NO
1
G4
OPERAND FOUND
NO
YES
H4
GET NUMTBL NUMBER FOR THIS ENTRY AND REDUCE BY 1
*J2
J4
RETURN TO ADDRESS IN LINK REGISTER
A5
CHKRNG
B5
GET CHARACTER STRING ADDRESS AND NUMBER OF CHARACTERS
CHKNXC
C5
CHARACTER LESS THAN ZERO
YES
NO
RNGTOP
D5
CHARACTER GT UPPER LIMIT
YES
NO
E5
ADD 1 TO POINTER, SUBTRACT 1 FROM COUNT
F5
END OF STRING
NO
YES
G5
RETURN TO ADDRESS IN LINK REGISTER

*J2

SYS	NUMTBL VALUE	FINAL NUMBER
RDR	1	0
IPT	2	1
PCH	3	2
LST	4	3
LOG	5	4
LNT	6	5
SLB	8	7
RLB	9	8

Chart PC. $$BATTNI - UNA Statement Processor
Refer to Chart 13.

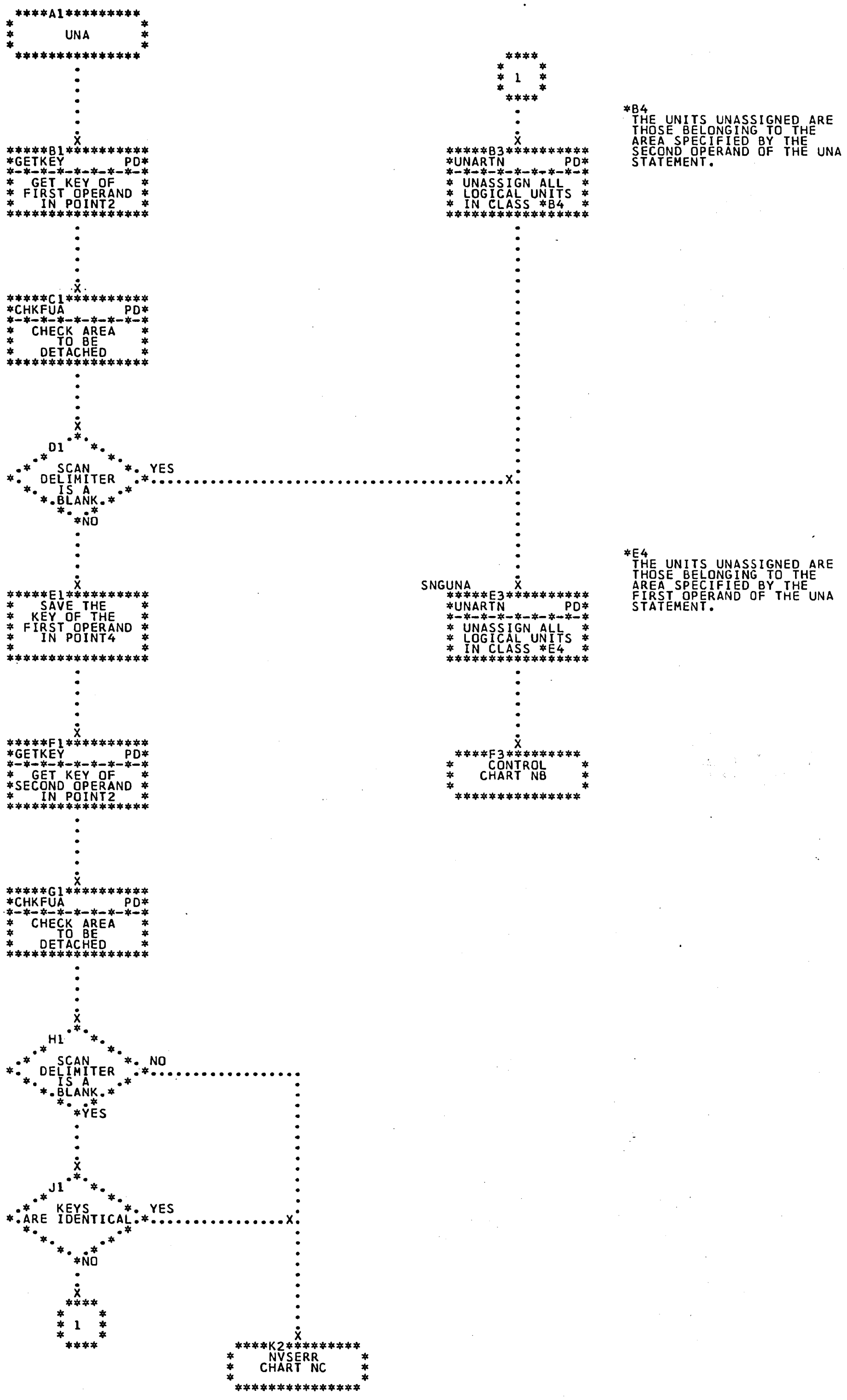

A1
UNA
B1
GETKEY PD
GET KEY OF FIRST OPERAND IN POINT2
C1
CHKFUA PD
CHECK AREA TO BE DETACHED
D1
SCAN DELIMITER IS A BLANK
YES
NO
E1
SAVE THE KEY OF THE FIRST OPERAND IN POINT4
F1
GETKEY PD
GET KEY OF SECOND OPERAND IN POINT2
G1
CHKFUA PD
CHECK AREA TO BE DETACHED
H1
SCAN DELIMITER IS A BLANK
NO
YES
J1
KEYS ARE IDENTICAL
YES
NO
1
K2
NVSERR CHART NC
1
B3
UNARTN PD
UNASSIGN ALL LOGICAL UNITS IN CLASS *B4
SNGUNA
E3
UNARTN PD
UNASSIGN ALL LOGICAL UNITS IN CLASS *E4
F3
CONTROL CHART NB
*B4
THE UNITS UNASSIGNED ARE THOSE BELONGING TO THE AREA SPECIFIED BY THE SECOND OPERAND OF THE UNA STATEMENT.
*E4
THE UNITS UNASSIGNED ARE THOSE BELONGING TO THE AREA SPECIFIED BY THE FIRST OPERAND OF THE UNA STATEMENT.

Chart PD. $$BATTNI - Miscellaneous Subroutines
Refer to Chart 13.

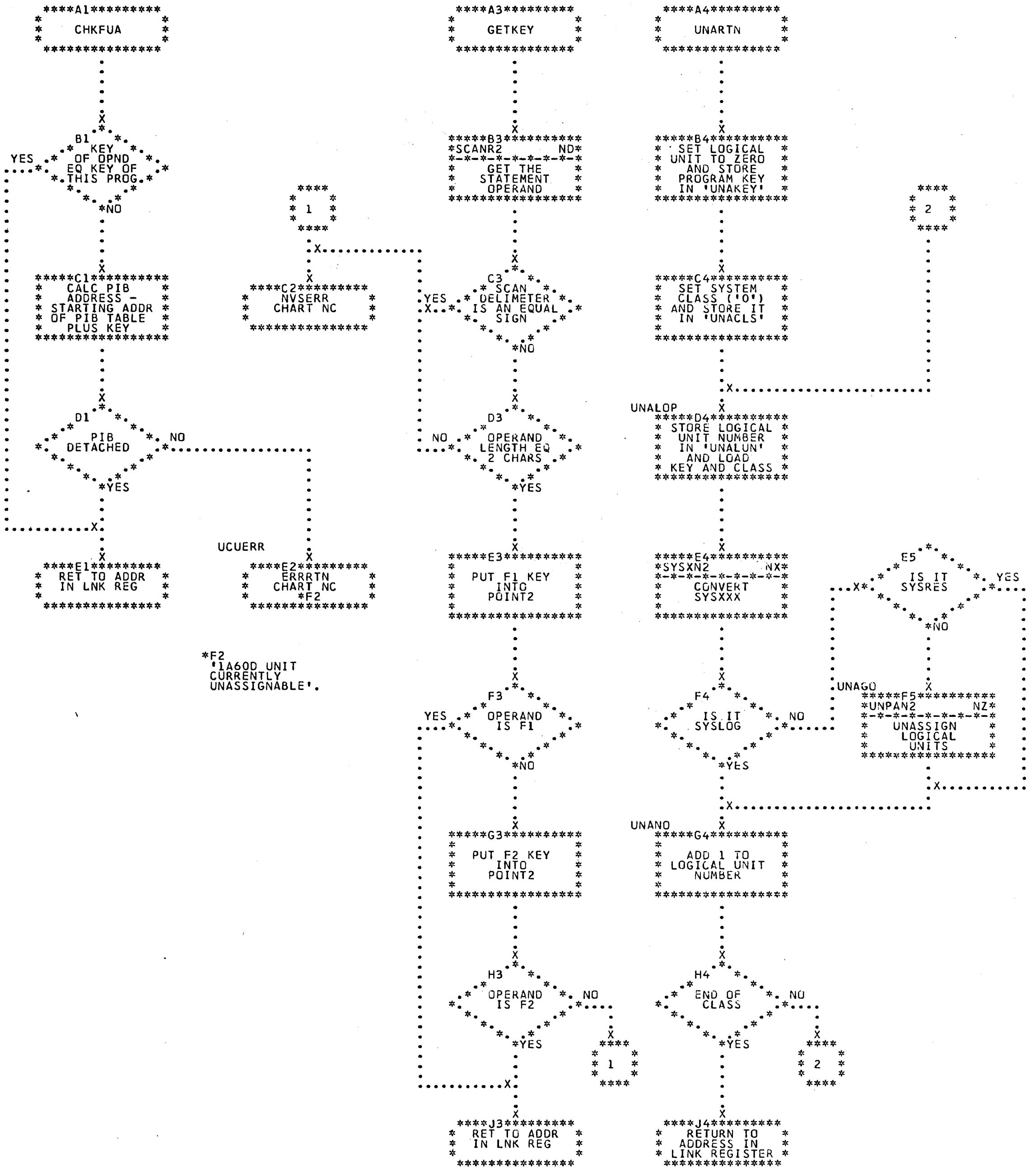
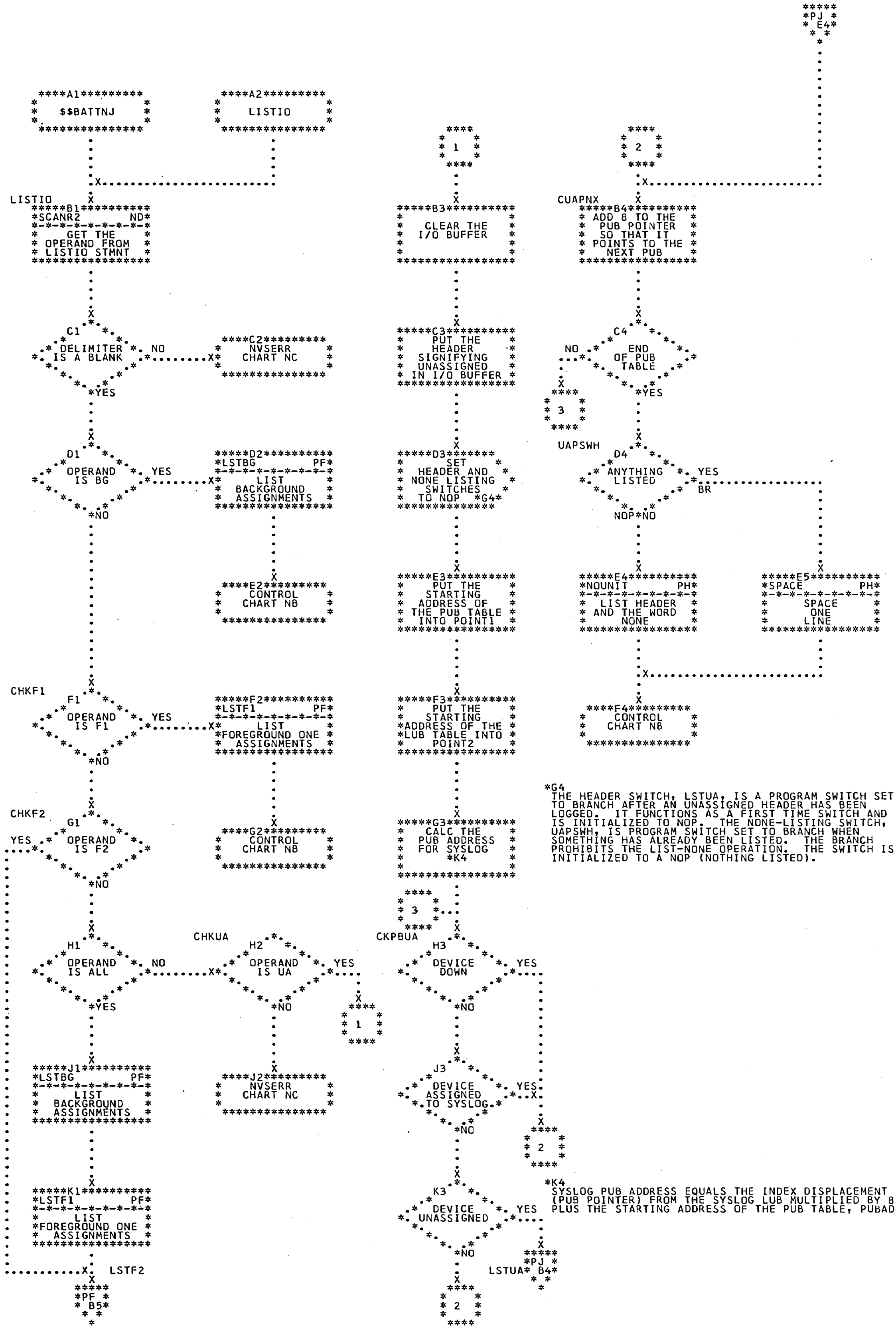

A1 CHKFUA
A3 GETKEY
A4 UNARTN
B1 KEY OF OPND EQ KEY OF THIS PROG.
YES
NO
1
B3 SCANR2 ND
GET THE STATEMENT OPERAND
B4 SET LOGICAL UNIT TO ZERO AND STORE PROGRAM KEY IN 'UNAKEY'
2
C1 CALC PIB ADDRESS - STARTING ADDR OF PIB TABLE PLUS KEY
C2 NVSERR CHART NC
C3 SCAN DELIMETER IS AN EQUAL SIGN
YES
NO
C4 SET SYSTEM CLASS ('0') AND STORE IT IN 'UNACLS'
D1 PIB DETACHED
NO
YES
D3 OPERAND LENGTH EQ 2 CHARS
NO
YES
UNALOP
D4 STORE LOGICAL UNIT NUMBER IN 'UNALUN' AND LOAD KEY AND CLASS
E1 RET TO ADDR IN LNK REG
UCUERR
E2 ERRRTN CHART NC *F2
E3 PUT F1 KEY INTO POINT2
E4 SYSXN2 NX
CONVERT SYSXXX
E5 IS IT SYSRES
YES
NO
*F2 '1A60D UNIT CURRENTLY UNASSIGNABLE'.
F3 OPERAND IS F1
YES
NO
F4 IS IT SYSLOG
NO
YES
UNAGO
F5 UNPAN2 NZ
UNASSIGN LOGICAL UNITS
G3 PUT F2 KEY INTO POINT2
UNANO
G4 ADD 1 TO LOGICAL UNIT NUMBER
H3 OPERAND IS F2
NO
YES
1
H4 END OF CLASS
NO
YES
2
J3 RET TO ADDR IN LNK REG
J4 RETURN TO ADDRESS IN LINK REGISTER

Chart PE. $$BATTNJ - LISTIO Statement Processor
Refer to Chart 13.


PJ E4
A1 $$BATTNJ
A2 LISTIO
1
2
LISTIO
B1 SCANR2 ND GET THE OPERAND FROM LISTIO STMNT
B3 CLEAR THE I/O BUFFER
CUAPNX
B4 ADD 8 TO THE PUB POINTER SO THAT IT POINTS TO THE NEXT PUB
C1 DELIMITER IS A BLANK
NO
YES
C2 NVSERR CHART NC
C3 PUT THE HEADER SIGNIFYING UNASSIGNED IN I/O BUFFER
C4 END OF PUB TABLE
NO
YES
3
D1 OPERAND IS BG
YES
NO
D2 LSTBG PF LIST BACKGROUND ASSIGNMENTS
D3 SET HEADER AND NONE LISTING SWITCHES TO NOP *G4
UAPSWH
D4 ANYTHING LISTED
YES
BR
NOP NO
E2 CONTROL CHART NB
E3 PUT THE STARTING ADDRESS OF THE PUB TABLE INTO POINT1
E4 NOUNIT PH LIST HEADER AND THE WORD NONE
E5 SPACE PH SPACE ONE LINE
CHKF1
F1 OPERAND IS F1
YES
NO
F2 LSTF1 PF LIST FOREGROUND ONE ASSIGNMENTS
F3 PUT THE STARTING ADDRESS OF THE LUB TABLE INTO POINT2
F4 CONTROL CHART NB
CHKF2
G1 OPERAND IS F2
YES
NO
G2 CONTROL CHART NB
G3 CALC THE PUB ADDRESS FOR SYSLOG *K4
3
H1 OPERAND IS ALL
NO
YES
CHKUA
H2 OPERAND IS UA
YES
NO
1
CKPBUA
H3 DEVICE DOWN
YES
NO
J1 LSTBG PF LIST BACKGROUND ASSIGNMENTS
J2 NVSERR CHART NC
J3 DEVICE ASSIGNED TO SYSLOG
YES
NO
2
K1 LSTF1 PF LIST FOREGROUND ONE ASSIGNMENTS
K3 DEVICE UNASSIGNED
YES
NO
LSTF2
PF B5
LSTUA PJ B4
2

*G4
THE HEADER SWITCH, LSTUA, IS A PROGRAM SWITCH SET TO BRANCH AFTER AN UNASSIGNED HEADER HAS BEEN LOGGED. IT FUNCTIONS AS A FIRST TIME SWITCH AND IS INITIALIZED TO NOP. THE NONE-LISTING SWITCH, UAPSWH, IS PROGRAM SWITCH SET TO BRANCH WHEN SOMETHING HAS ALREADY BEEN LISTED. THE BRANCH PROHIBITS THE LIST-NONE OPERATION. THE SWITCH IS INITIALIZED TO A NOP (NOTHING LISTED).

*K4
SYSLOG PUB ADDRESS EQUALS THE INDEX DISPLACEMENT (PUB POINTER) FROM THE SYSLOG LUB MULTIPLIED BY 8 PLUS THE STARTING ADDRESS OF THE PUB TABLE, PUBADD.

Chart PF. $$BATTNJ - Subroutines
Refer to Chart 13.

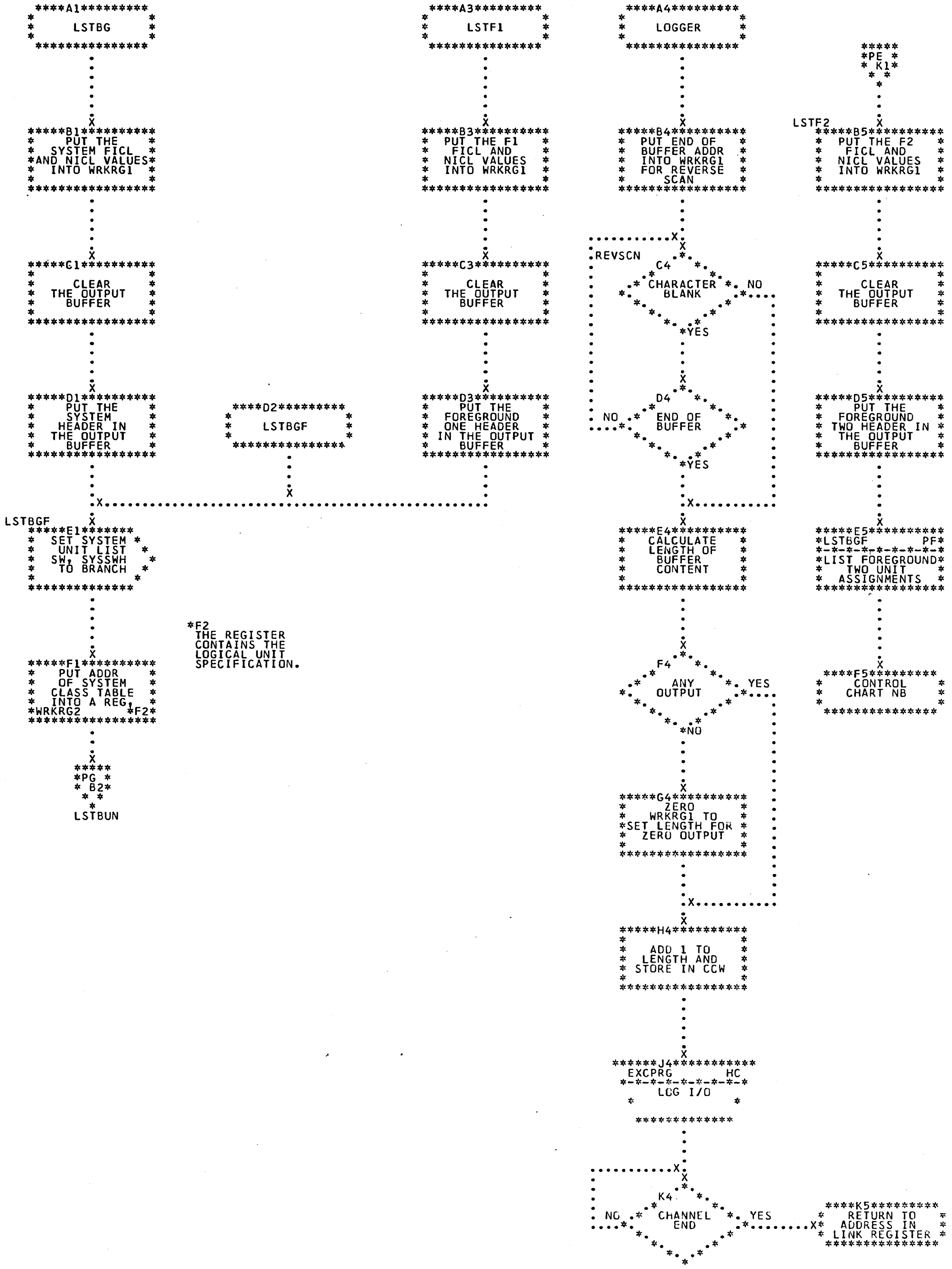
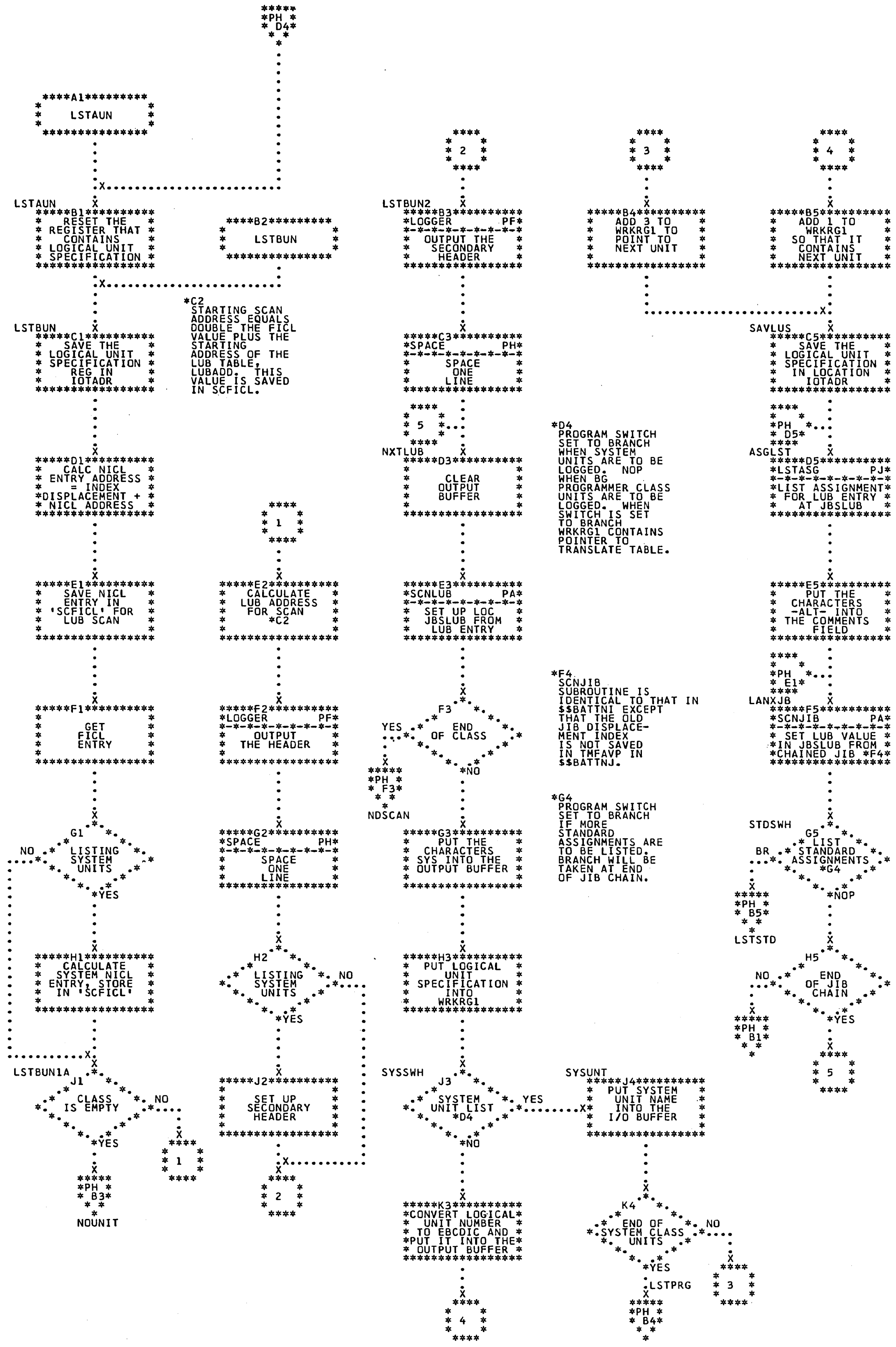
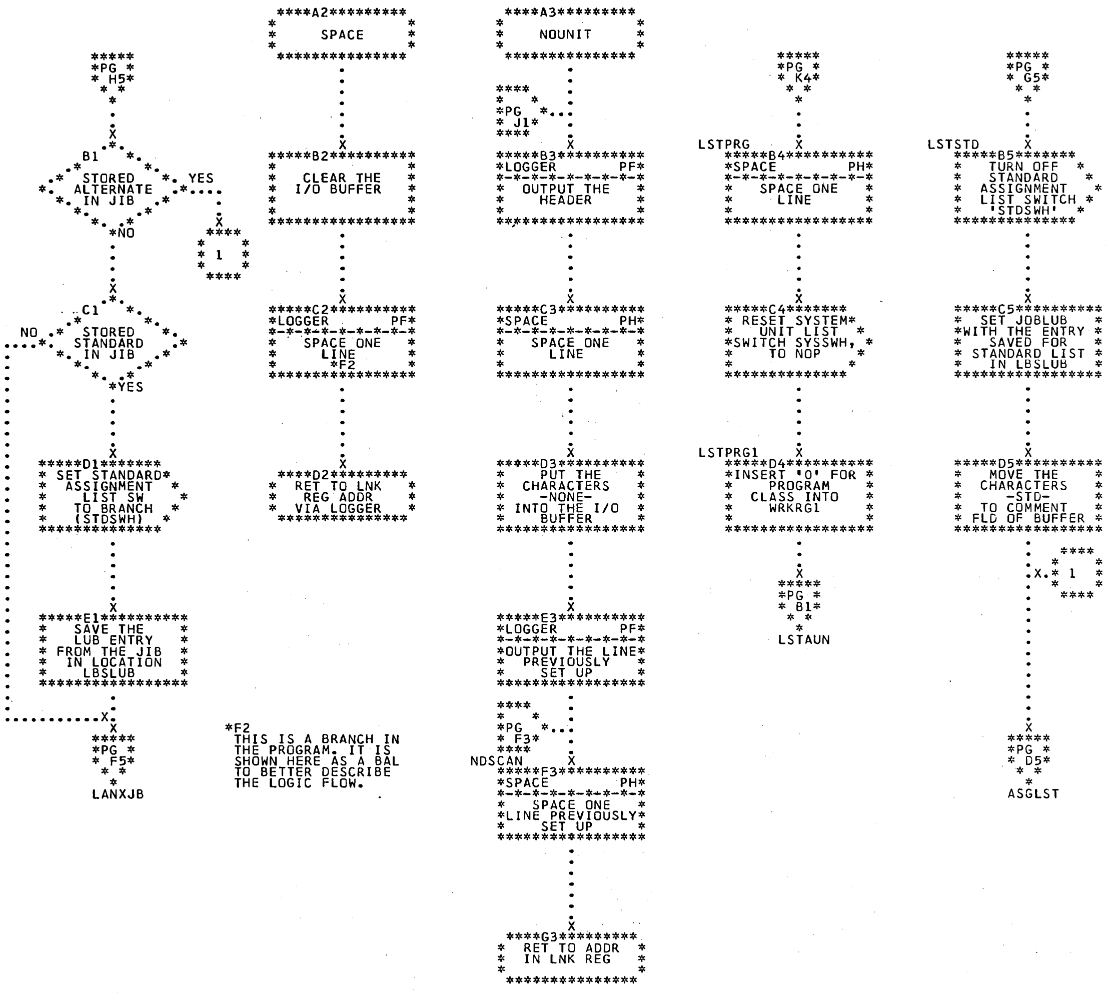

A1
LSTBG
B1
PUT THE SYSTEM FICL AND NICL VALUES INTO WRKRG1
C1
CLEAR THE OUTPUT BUFFER
D1
PUT THE SYSTEM HEADER IN THE OUTPUT BUFFER
D2
LSTBGF
LSTBGF
E1
SET SYSTEM UNIT LIST SW, SYSSWH TO BRANCH
F1
PUT ADDR OF SYSTEM CLASS TABLE INTO A REG, WRKRG2
F2
*F2
THE REGISTER CONTAINS THE LOGICAL UNIT SPECIFICATION.
PG
B2
LSTBUN
A3
LSTF1
B3
PUT THE F1 FICL AND NICL VALUES INTO WRKRG1
C3
CLEAR THE OUTPUT BUFFER
D3
PUT THE FOREGROUND ONE HEADER IN THE OUTPUT BUFFER
A4
LOGGER
B4
PUT END OF BUFFER ADDR INTO WRKRG1 FOR REVERSE SCAN
REVSCN
C4
CHARACTER BLANK
NO
YES
D4
END OF BUFFER
NO
YES
E4
CALCULATE LENGTH OF BUFFER CONTENT
F4
ANY OUTPUT
YES
NO
G4
ZERO WRKRG1 TO SET LENGTH FOR ZERO OUTPUT
H4
ADD 1 TO LENGTH AND STORE IN CCW
J4
EXCPRG
HC
LCG I/O
K4
CHANNEL END
NO
YES
K5
RETURN TO ADDRESS IN LINK REGISTER
PE
K1
LSTF2
B5
PUT THE F2 FICL AND NICL VALUES INTO WRKRG1
C5
CLEAR THE OUTPUT BUFFER
D5
PUT THE FOREGROUND TWO HEADER IN THE OUTPUT BUFFER
E5
LSTBGF
PF
LIST FOREGROUND TWO UNIT ASSIGNMENTS
F5
CONTROL CHART NB

Chart PG. $$BATTNJ - Locate Assignment Routine
Refer to Chart 13.



PH D4
A1 LSTAUN
LSTAUN B1 RESET THE REGISTER THAT CONTAINS LOGICAL UNIT SPECIFICATION
B2 LSTBUN
LSTBUN C1 SAVE THE LOGICAL UNIT SPECIFICATION REG IN IOTADR
C2 STARTING SCAN ADDRESS EQUALS DOUBLE THE FICL VALUE PLUS THE STARTING ADDRESS OF THE LUB TABLE, LUBADD. THIS VALUE IS SAVED IN SCFICL.
D1 CALC NICL ENTRY ADDRESS = INDEX DISPLACEMENT + NICL ADDRESS
1
E1 SAVE NICL ENTRY IN 'SCFICL' FOR LUB SCAN
E2 CALCULATE LUB ADDRESS FOR SCAN *C2
F1 GET FICL ENTRY
F2 LOGGER PF OUTPUT THE HEADER
G1 LISTING SYSTEM UNITS
NO
YES
G2 SPACE PH SPACE ONE LINE
H1 CALCULATE SYSTEM NICL ENTRY, STORE IN 'SCFICL'
H2 LISTING SYSTEM UNITS
NO
YES
LSTBUN1A J1 CLASS IS EMPTY
NO
YES
1
PH B3
NOUNIT
J2 SET UP SECONDARY HEADER
2
2
LSTBUN2 B3 LOGGER PF OUTPUT THE SECONDARY HEADER
C3 SPACE PH SPACE ONE LINE
5
NXTLUB D3 CLEAR OUTPUT BUFFER
E3 SCNLUB PA SET UP LOC JBSLUB FROM LUB ENTRY
F3 END OF CLASS
YES
NO
PH F3
NDSCAN
G3 PUT THE CHARACTERS SYS INTO THE OUTPUT BUFFER
H3 PUT LOGICAL UNIT SPECIFICATION INTO WRKRG1
SYSSWH J3 SYSTEM UNIT LIST *D4
YES
NO
K3 CONVERT LOGICAL UNIT NUMBER TO EBCDIC AND PUT IT INTO THE OUTPUT BUFFER
4
3
B4 ADD 3 TO WRKRG1 TO POINT TO NEXT UNIT
D4 PROGRAM SWITCH SET TO BRANCH WHEN SYSTEM UNITS ARE TO BE LOGGED. NOP WHEN BG PROGRAMMER CLASS UNITS ARE TO BE LOGGED. WHEN SWITCH IS SET TO BRANCH WRKRG1 CONTAINS POINTER TO TRANSLATE TABLE.
F4 SCNJIB SUBROUTINE IS IDENTICAL TO THAT IN $$BATTNI EXCEPT THAT THE OLD JIB DISPLACEMENT INDEX IS NOT SAVED IN TMFAVP IN $$BATTNJ.
G4 PROGRAM SWITCH SET TO BRANCH IF MORE STANDARD ASSIGNMENTS ARE TO BE LISTED. BRANCH WILL BE TAKEN AT END OF JIB CHAIN.
SYSUNT J4 PUT SYSTEM UNIT NAME INTO THE I/O BUFFER
K4 END OF SYSTEM CLASS UNITS
NO
YES
LSTPRG
PH B4
3
4
B5 ADD 1 TO WRKRG1 SO THAT IT CONTAINS NEXT UNIT
SAVLUS C5 SAVE THE LOGICAL UNIT SPECIFICATION IN LOCATION IOTADR
PH D5
ASGLST D5 LSTASG PJ LIST ASSIGNMENT FOR LUB ENTRY AT JBSLUB
E5 PUT THE CHARACTERS -ALT- INTO THE COMMENTS FIELD
PH E1
LANXJB F5 SCNJIB PA SET LUB VALUE IN JBSLUB FROM CHAINED JIB *F4
STDSWH G5 LIST STANDARD ASSIGNMENTS *G4
BR
NOP
PH B5
LSTSTD
H5 END OF JIB CHAIN
NO
YES
PH B1
5

Chart PH. $$BATTNJ - Output List (Part 1 of 2)
Refer to Chart 13.


PG H5
B1
STORED ALTERNATE IN JIB
YES
NO
1
C1
STORED STANDARD IN JIB
NO
YES
D1
SET STANDARD ASSIGNMENT LIST SW TO BRANCH (STDSWH)
E1
SAVE THE LUB ENTRY FROM THE JIB IN LOCATION LBSLUB
PG F5
LANXJB
A2
SPACE
B2
CLEAR THE I/O BUFFER
C2
LOGGER PF
SPACE ONE LINE
*F2
D2
RET TO LNK REG ADDR VIA LOGGER
A3
NOUNIT
PG J1
B3
LOGGER PF
OUTPUT THE HEADER
C3
SPACE PH
SPACE ONE LINE
D3
PUT THE CHARACTERS -NONE- INTO THE I/O BUFFER
E3
LOGGER PF
OUTPUT THE LINE PREVIOUSLY SET UP
PG F3
NDSCAN
F3
SPACE PH
SPACE ONE LINE PREVIOUSLY SET UP
G3
RET TO ADDR IN LNK REG
PG K4
LSTPRG
B4
SPACE PH
SPACE ONE LINE
C4
RESET SYSTEM UNIT LIST SWITCH SYSSWH, TO NOP
LSTPRG1
D4
INSERT '0' FOR PROGRAM CLASS INTO WRKRG1
PG B1
LSTAUN
PG G5
LSTSTD
B5
TURN OFF STANDARD ASSIGNMENT LIST SWITCH 'STDSWH'
C5
SET JOBLUB WITH THE ENTRY SAVED FOR STANDARD LIST IN LBSLUB
D5
MOVE THE CHARACTERS -STD- TO COMMENT FLD OF BUFFER
1
PG D5
ASGLST

*F2
THIS IS A BRANCH IN THE PROGRAM. IT IS SHOWN HERE AS A BAL TO BETTER DESCRIBE THE LOGIC FLOW.

Chart PJ. $$BATTNJ - Output List (Part 2 of 2)
Refer to Chart 13.

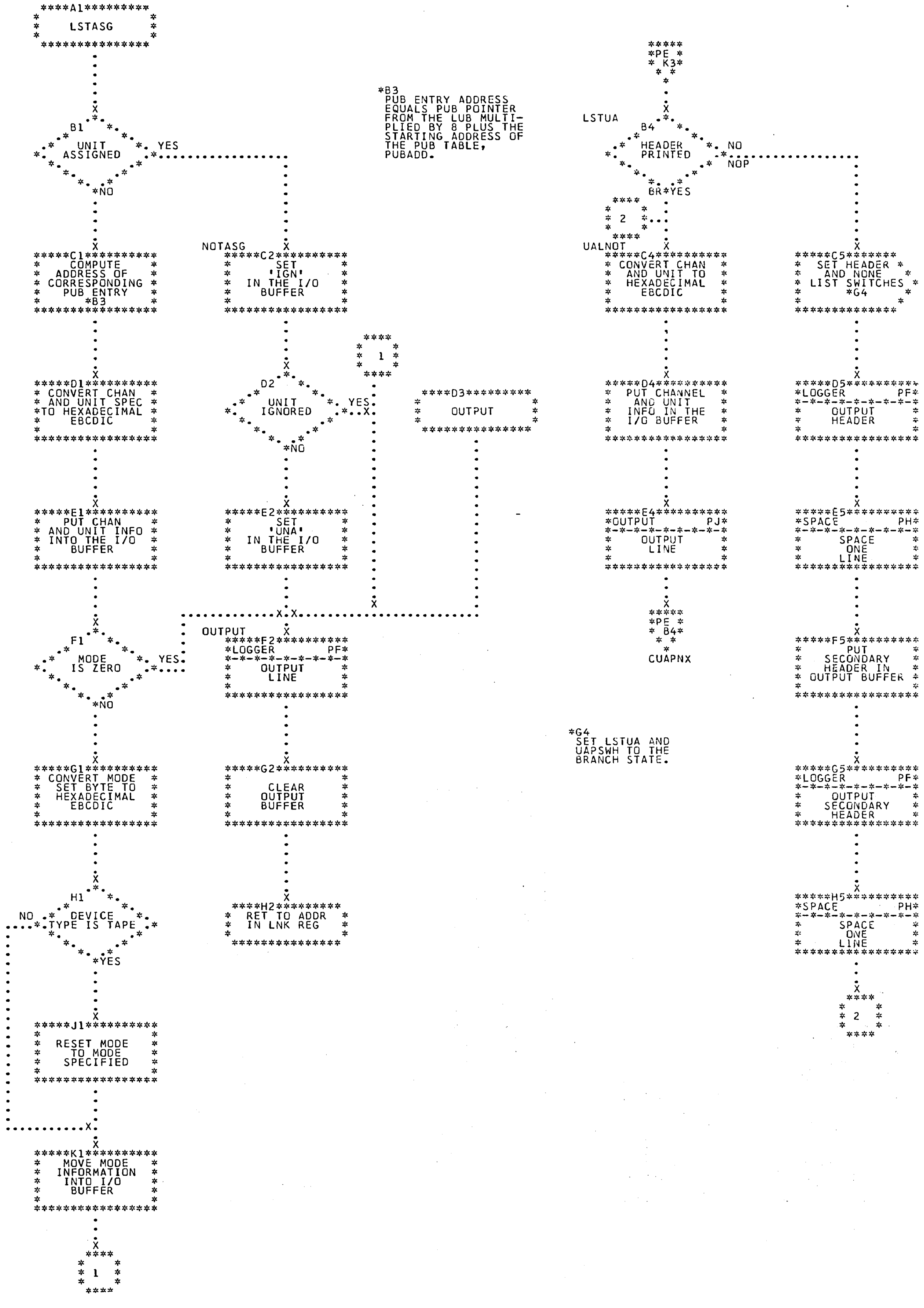
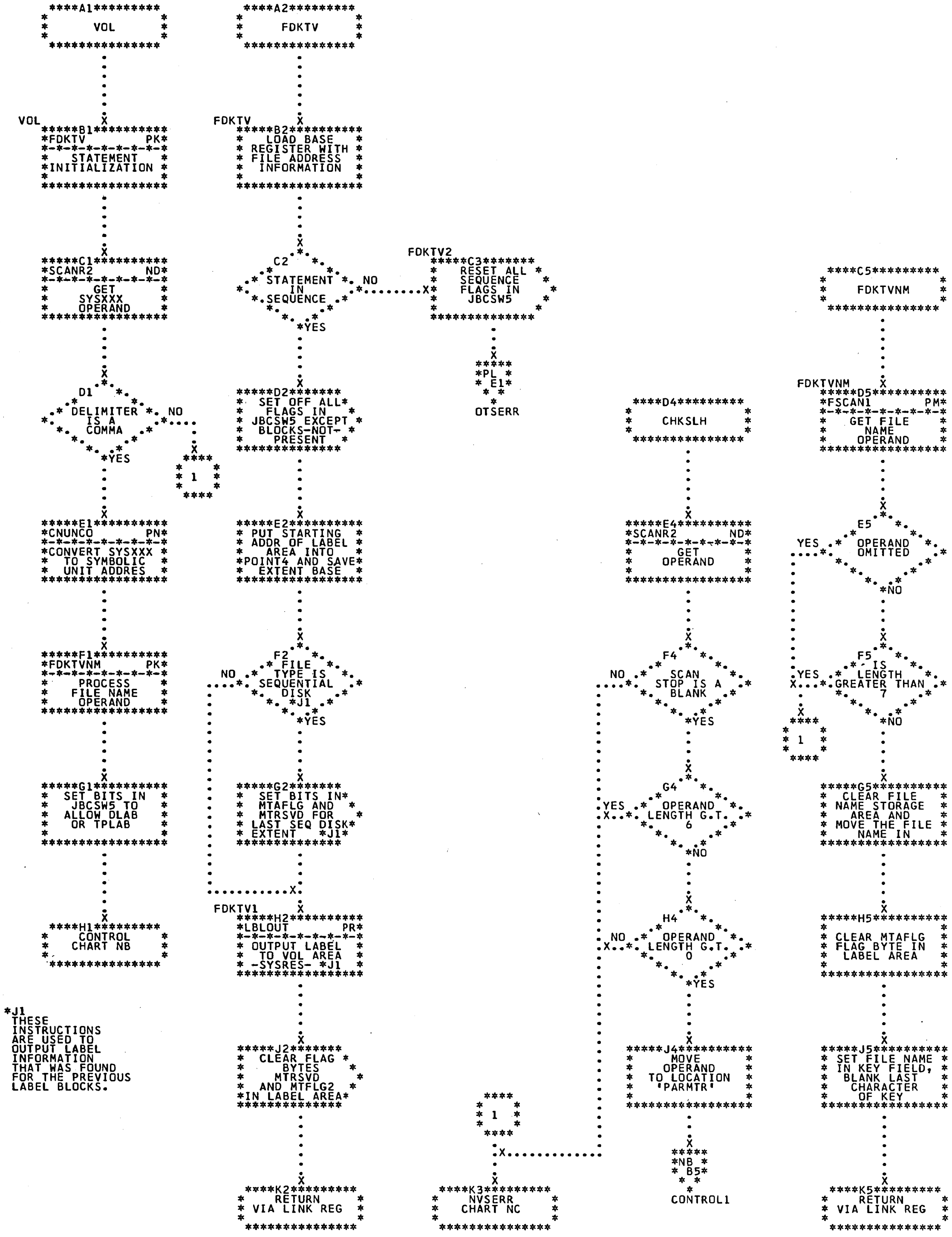

A1 LSTASG
B1 UNIT ASSIGNED
YES
NO
C1 COMPUTE ADDRESS OF CORRESPONDING PUB ENTRY *B3
D1 CONVERT CHAN AND UNIT SPEC TO HEXADECIMAL EBCDIC
E1 PUT CHAN AND UNIT INFO INTO THE I/O BUFFER
F1 MODE IS ZERO
YES
NO
G1 CONVERT MODE SET BYTE TO HEXADECIMAL EBCDIC
H1 DEVICE TYPE IS TAPE
NO
YES
J1 RESET MODE TO MODE SPECIFIED
K1 MOVE MODE INFORMATION INTO I/O BUFFER
1
NOTASG
C2 SET 'IGN' IN THE I/O BUFFER
D2 UNIT IGNORED
YES
NO
E2 SET 'UNA' IN THE I/O BUFFER
OUTPUT
F2 LOGGER PF OUTPUT LINE
G2 CLEAR OUTPUT BUFFER
H2 RET TO ADDR IN LNK REG
1
D3 OUTPUT
*B3
PUB ENTRY ADDRESS EQUALS PUB POINTER FROM THE LUB MULTIPLIED BY 8 PLUS THE STARTING ADDRESS OF THE PUB TABLE, PUBADD.
PE K3
LSTUA
B4 HEADER PRINTED
NO
NOP
BR YES
2
UALNOT
C4 CONVERT CHAN AND UNIT TO HEXADECIMAL EBCDIC
D4 PUT CHANNEL AND UNIT INFO IN THE I/O BUFFER
E4 OUTPUT PJ OUTPUT LINE
PE B4
CUAPNX
*G4
SET LSTUA AND UAPSWH TO THE BRANCH STATE.
C5 SET HEADER AND NONE LIST SWITCHES *G4
D5 LOGGER PF OUTPUT HEADER
E5 SPACE PH SPACE ONE LINE
F5 PUT SECONDARY HEADER IN OUTPUT BUFFER
G5 LOGGER PF OUTPUT SECONDARY HEADER
H5 SPACE PH SPACE ONE LINE
2

Chart PK. $$BATTNK - VOL Statement Processor
Refer to Chart 13.


A1 VOL
A2 FDKTV
VOL B1 FDKTV PK STATEMENT INITIALIZATION
FDKTV B2 LOAD BASE REGISTER WITH FILE ADDRESS INFORMATION
C1 SCANR2 ND GET SYSXXX OPERAND
C2 STATEMENT IN SEQUENCE
NO
YES
FDKTV2 C3 RESET ALL SEQUENCE FLAGS IN JBCSW5
PL E1 OTSERR
C5 FDKTVNM
D1 DELIMITER IS A COMMA
NO
YES
1
D2 SET OFF ALL FLAGS IN JBCSW5 EXCEPT BLOCKS-NOT- PRESENT
D4 CHKSLH
FDKTVNM D5 FSCAN1 PM GET FILE NAME OPERAND
E1 CNUNCO PN CONVERT SYSXXX TO SYMBOLIC UNIT ADDRES
E2 PUT STARTING ADDR OF LABEL AREA INTO POINT4 AND SAVE EXTENT BASE
E4 SCANR2 ND GET OPERAND
E5 OPERAND OMITTED
YES
NO
F1 FDKTVNM PK PROCESS FILE NAME OPERAND
F2 FILE TYPE IS SEQUENTIAL DISK J1
NO
YES
F4 SCAN STOP IS A BLANK
NO
YES
F5 IS LENGTH GREATER THAN 7
YES
NO
1
G1 SET BITS IN JBCSW5 TO ALLOW DLAB OR TPLAB
G2 SET BITS IN MTAFLG AND MTRSVD FOR LAST SEQ DISK EXTENT J1
G4 OPERAND LENGTH G.T. 6
YES
NO
G5 CLEAR FILE NAME STORAGE AREA AND MOVE THE FILE NAME IN
H1 CONTROL CHART NB
FDKTV1 H2 LBLOUT PR OUTPUT LABEL TO VOL AREA -SYSRES- J1
H4 OPERAND LENGTH G.T. 0
NO
YES
H5 CLEAR MTAFLG FLAG BYTE IN LABEL AREA
J2 CLEAR FLAG BYTES MTRSVD AND MTFLG2 IN LABEL AREA
J4 MOVE OPERAND TO LOCATION 'PARMTR'
J5 SET FILE NAME IN KEY FIELD, BLANK LAST CHARACTER OF KEY
1
NB B5 CONTROL1
K2 RETURN VIA LINK REG
K3 NVSERR CHART NC
K5 RETURN VIA LINK REG

*J1
THESE INSTRUCTIONS ARE USED TO OUTPUT LABEL INFORMATION THAT WAS FOUND FOR THE PREVIOUS LABEL BLOCKS.

Chart PL. $$BATTNK - TPLAB Statement Processor
Refer to Chart 13.

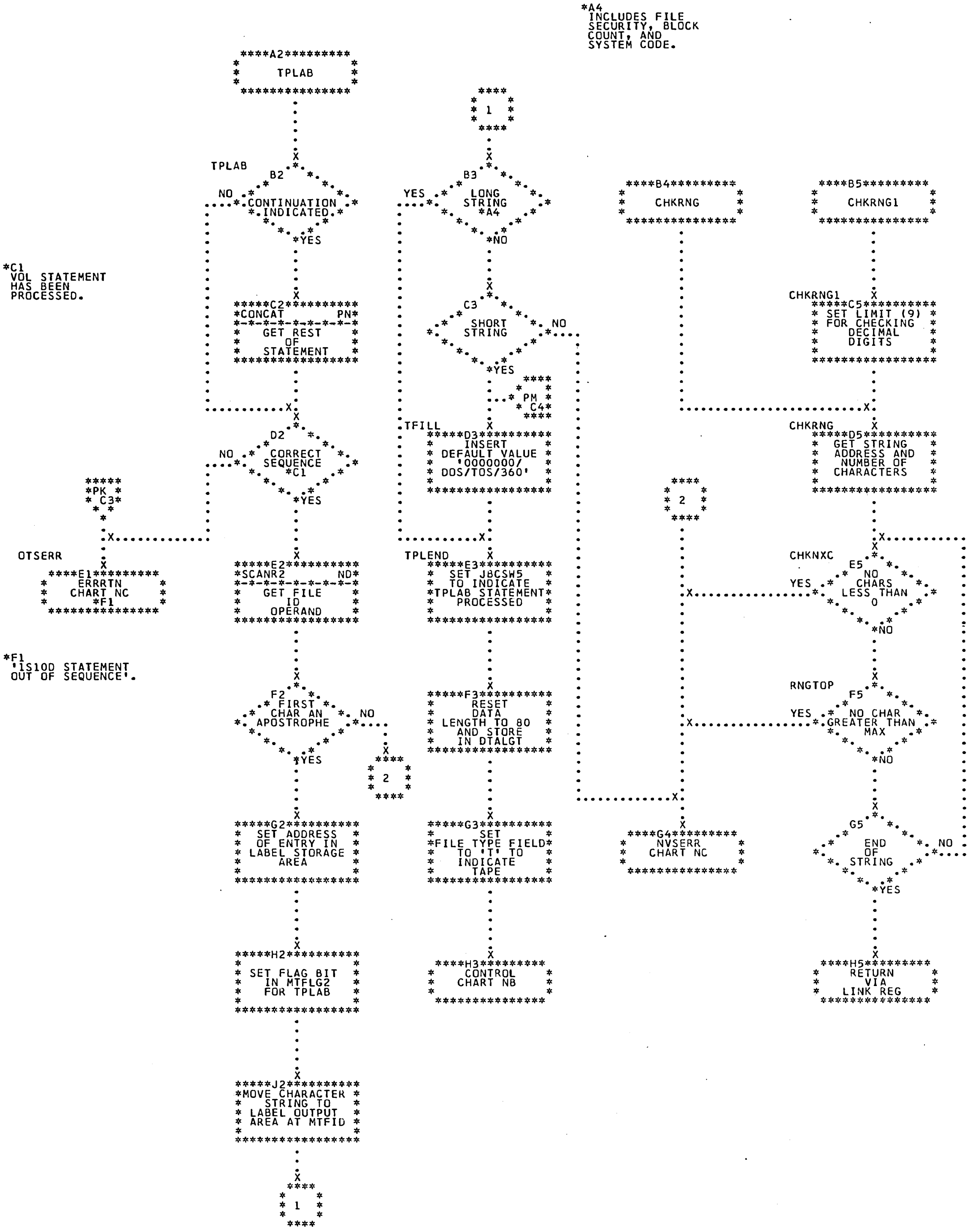
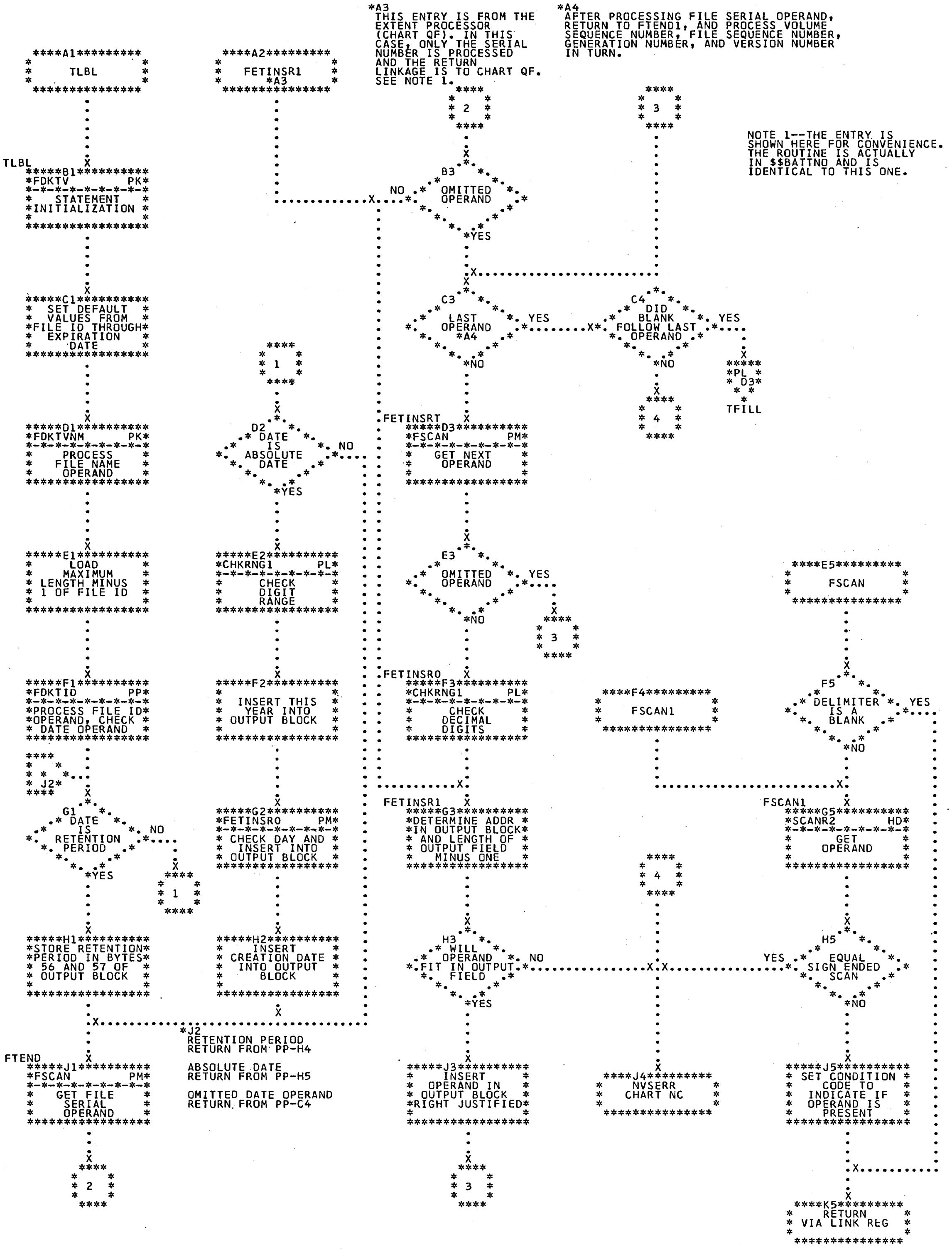

*A4
INCLUDES FILE SECURITY, BLOCK COUNT, AND SYSTEM CODE.
A2
TPLAB
1
TPLAB
B2
CONTINUATION INDICATED
NO
YES
B3
LONG STRING *A4
YES
NO
B4
CHKRNG
B5
CHKRNG1
*C1
VOL STATEMENT HAS BEEN PROCESSED.
C2
CONCAT PN
GET REST OF STATEMENT
C3
SHORT STRING
NO
YES
PM C4
CHKRNG1
C5
SET LIMIT (9) FOR CHECKING DECIMAL DIGITS
D2
CORRECT SEQUENCE *C1
NO
YES
TFILL
D3
INSERT DEFAULT VALUE '0000000/ DOS/TOS/360'
CHKRNG
D5
GET STRING ADDRESS AND NUMBER OF CHARACTERS
PK C3
2
OTSERR
E1
ERRRTN CHART NC *F1
E2
SCANR2 ND
GET FILE ID OPERAND
TPLEND
E3
SET JBCSW5 TO INDICATE TPLAB STATEMENT PROCESSED
CHKNXC
E5
NO CHARS LESS THAN 0
YES
NO
*F1
'1S10D STATEMENT OUT OF SEQUENCE'.
F2
FIRST CHAR AN APOSTROPHE
NO
YES
2
F3
RESET DATA LENGTH TO 80 AND STORE IN DTALGT
RNGTOP
F5
NO CHAR GREATER THAN MAX
YES
NO
G2
SET ADDRESS OF ENTRY IN LABEL STORAGE AREA
G3
SET FILE TYPE FIELD TO 'T' TO INDICATE TAPE
G4
NVSERR CHART NC
G5
END OF STRING
NO
YES
H2
SET FLAG BIT IN MTFLG2 FOR TPLAB
H3
CONTROL CHART NB
H5
RETURN VIA LINK REG
J2
MOVE CHARACTER STRING TO LABEL OUTPUT AREA AT MTFID
1

Chart PM. $$BATTNK - TLBL Statement Processor
Refer to Chart 13.


A1 TLBL
A2 FETINSR1 *A3
2
3
TLBL B1 FDKTV PK STATEMENT INITIALIZATION
B3 OMITTED OPERAND
NO
YES
C1 SET DEFAULT VALUES FROM FILE ID THROUGH EXPIRATION DATE
C3 LAST OPERAND *A4
YES
NO
C4 DID BLANK FOLLOW LAST OPERAND
YES
NO
PL D3 TFILL
4
1
D1 FDKTVNM PK PROCESS FILE NAME OPERAND
D2 DATE IS ABSOLUTE DATE
NO
YES
FETINSRT D3 FSCAN PM GET NEXT OPERAND
E1 LOAD MAXIMUM LENGTH MINUS 1 OF FILE ID
E2 CHKRNG1 PL CHECK DIGIT RANGE
E3 OMITTED OPERAND
YES
NO
3
E5 FSCAN
F1 FDKTID PP PROCESS FILE ID OPERAND, CHECK DATE OPERAND
F2 INSERT THIS YEAR INTO OUTPUT BLOCK
FETINSRO F3 CHKRNG1 PL CHECK DECIMAL DIGITS
F4 FSCAN1
F5 DELIMITER IS A BLANK
YES
NO
J2
G1 DATE IS RETENTION PERIOD
NO
YES
1
G2 FETINSRO PM CHECK DAY AND INSERT INTO OUTPUT BLOCK
FETINSR1 G3 DETERMINE ADDR IN OUTPUT BLOCK AND LENGTH OF OUTPUT FIELD MINUS ONE
FSCAN1 G5 SCANR2 HD GET OPERAND
4
H1 STORE RETENTION PERIOD IN BYTES 56 AND 57 OF OUTPUT BLOCK
H2 INSERT CREATION DATE INTO OUTPUT BLOCK
H3 WILL OPERAND FIT IN OUTPUT FIELD
NO
YES
H5 EQUAL SIGN ENDED SCAN
YES
NO
FTEND J1 FSCAN PM GET FILE SERIAL OPERAND
J3 INSERT OPERAND IN OUTPUT BLOCK RIGHT JUSTIFIED
J4 NVSERR CHART NC
J5 SET CONDITION CODE TO INDICATE IF OPERAND IS PRESENT
2
3
K5 RETURN VIA LINK REG

*A3
THIS ENTRY IS FROM THE EXTENT PROCESSOR (CHART QF). IN THIS CASE, ONLY THE SERIAL NUMBER IS PROCESSED AND THE RETURN LINKAGE IS TO CHART QF. SEE NOTE 1.

*A4
AFTER PROCESSING FILE SERIAL OPERAND, RETURN TO FTEND1, AND PROCESS VOLUME SEQUENCE NUMBER, FILE SEQUENCE NUMBER, GENERATION NUMBER, AND VERSION NUMBER IN TURN.

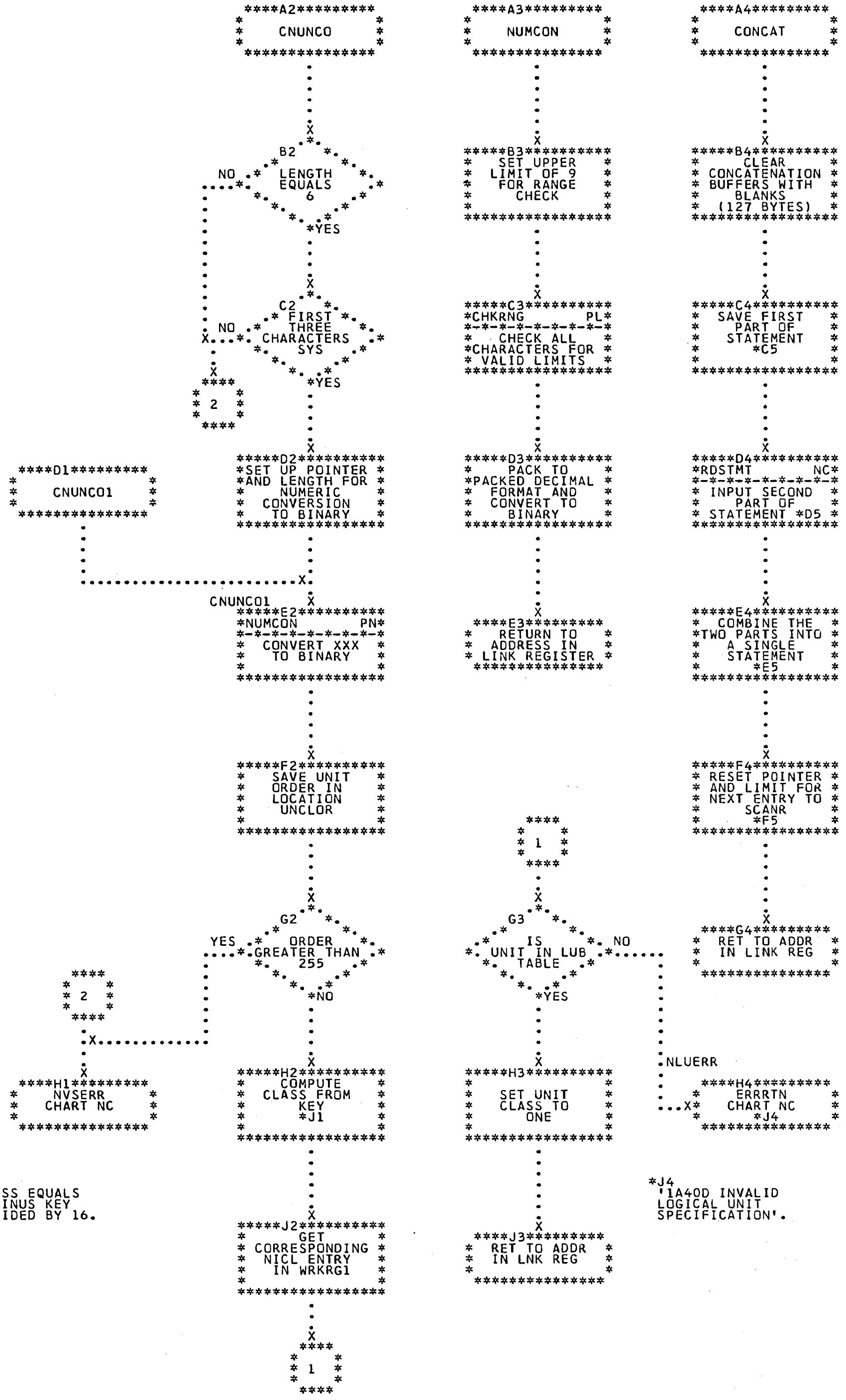
NOTE 1--THE ENTRY IS SHOWN HERE FOR CONVENIENCE. THE ROUTINE IS ACTUALLY IN $$BATTNO AND IS IDENTICAL TO THIS ONE.

*J2
RETENTION PERIOD RETURN FROM PP-H4

ABSOLUTE DATE RETURN FROM PP-H5

OMITTED DATE OPERAND RETURN FROM PP-C4

Chart PN. $$BATTNK - Check, Convert, and Concatenate Subroutines
Refer to Chart 13.


A2 CNUNCO
A3 NUMCON
A4 CONCAT
B2 LENGTH EQUALS 6
NO
YES
B3 SET UPPER LIMIT OF 9 FOR RANGE CHECK
B4 CLEAR CONCATENATION BUFFERS WITH BLANKS (127 BYTES)
C2 FIRST THREE CHARACTERS SYS
NO
YES
2
C3 CHKRNG PL CHECK ALL CHARACTERS FOR VALID LIMITS
C4 SAVE FIRST PART OF STATEMENT *C5
D1 CNUNCO1
D2 SET UP POINTER AND LENGTH FOR NUMERIC CONVERSION TO BINARY
D3 PACK TO PACKED DECIMAL FORMAT AND CONVERT TO BINARY
D4 RDSTMT NC INPUT SECOND PART OF STATEMENT *D5
CNUNCO1
E2 NUMCON PN CONVERT XXX TO BINARY
E3 RETURN TO ADDRESS IN LINK REGISTER
E4 COMBINE THE TWO PARTS INTO A SINGLE STATEMENT *E5
F2 SAVE UNIT ORDER IN LOCATION UNCLOR
F4 RESET POINTER AND LIMIT FOR NEXT ENTRY TO SCANR *F5
1
G2 ORDER GREATER THAN 255
YES
NO
G3 IS UNIT IN LUB TABLE
NO
YES
G4 RET TO ADDR IN LINK REG
2
NLUERR
H1 NVSERR CHART NC
H2 COMPUTE CLASS FROM KEY *J1
H3 SET UNIT CLASS TO ONE
H4 ERRRTN CHART NC *J4
J2 GET CORRESPONDING NICL ENTRY IN WRKRG1
J3 RET TO ADDR IN LNK REG
1

*C5
SAVE THE FIRST 70 BYTES OF THE I/O BUFFER, IN LOCATION BUFFER1 BY ORING THE TWO FIELDS. THIS WILL MAP ALL LOWERCASE CHARACTERS TO UPPERCASE.

*D5
THE SECOND PART OF THE STATEMENT IS NOW IN I/O BUFFER, AT LOCATION BUFFER.

*E5
COMBINE THE TWO PARTS BY ORING THE CONTENTS OF THE I/O BUFFER (STARTING AT BUFFER +15) WITH BUFFR2. BUFFR2 IS CONTIGUOUS WITH BUFFR1.

*F5
GET VALUES SAVED BY THE SCAN SUBROUTINE--
1. REMAINING BUFFER LENGTH.
2. ADDRESS OF OPERAND SUBTRACT THE ADDRESS FIELD OF THE CCW FROM THE STARTING ADDRESS OF THE OPERAND. ADD BUFFR1 ADDRESS TO THE DIFFERENCE. THE RESULTANT NEW OPERAND ADDRESS IS IN POINT1. ADD 56 TO THE REMAINING BUFFER FIELD. THE RESULTANT NEW BUFFER LENGTH IS IN POINT2. RESTORE POINT1 AND POINT2 IN LOCATION TMPAR1 WHERE IT IS REFERENCED BY THE SCAN SUBROUTINE.

*J1
CLASS EQUALS 4 MINUS KEY DIVIDED BY 16.

*J4
'1A40D INVALID LOGICAL UNIT SPECIFICATION'.

Chart PP. $$BATTNK - Process File ID and Date Operands
Refer to Chart 13.

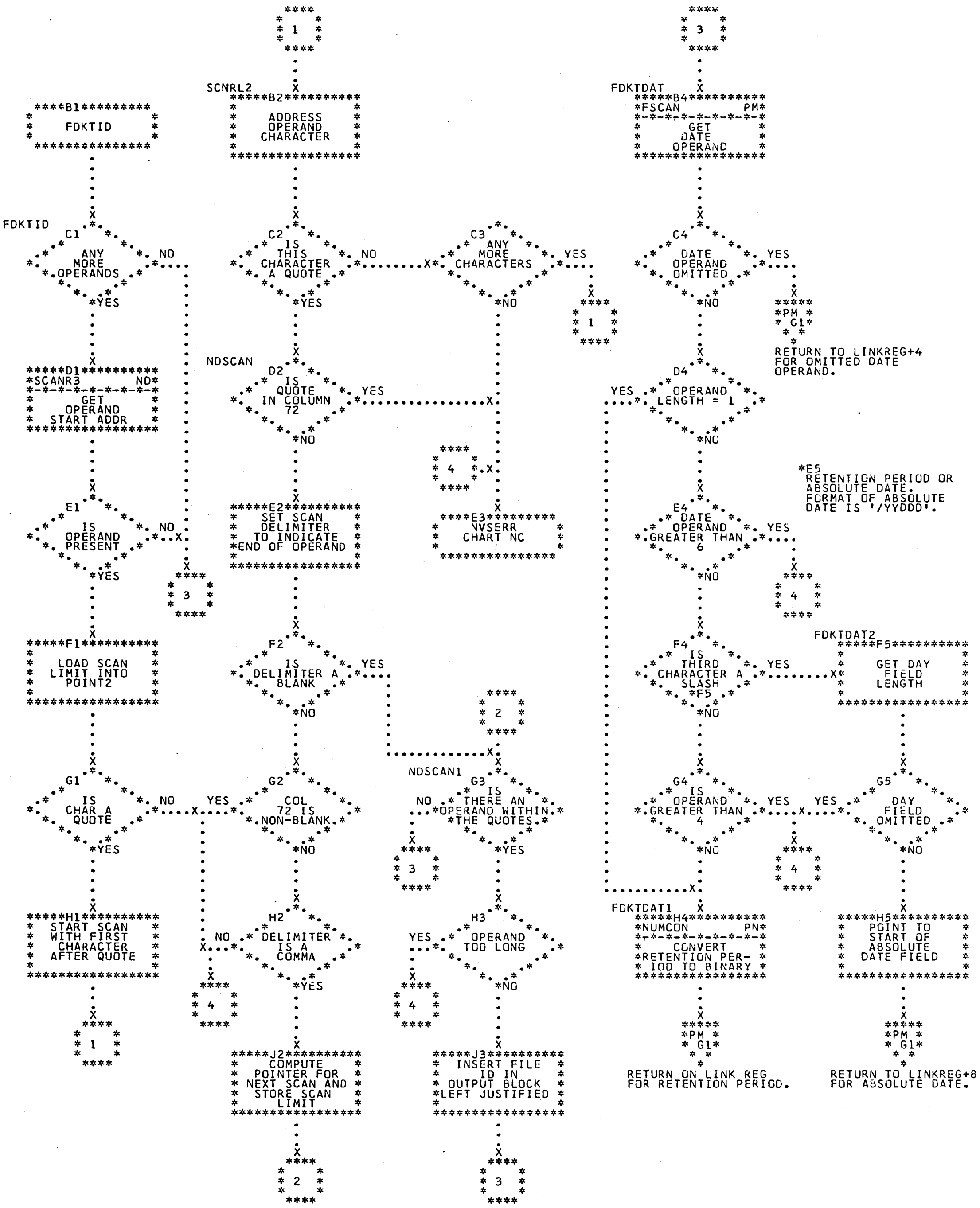
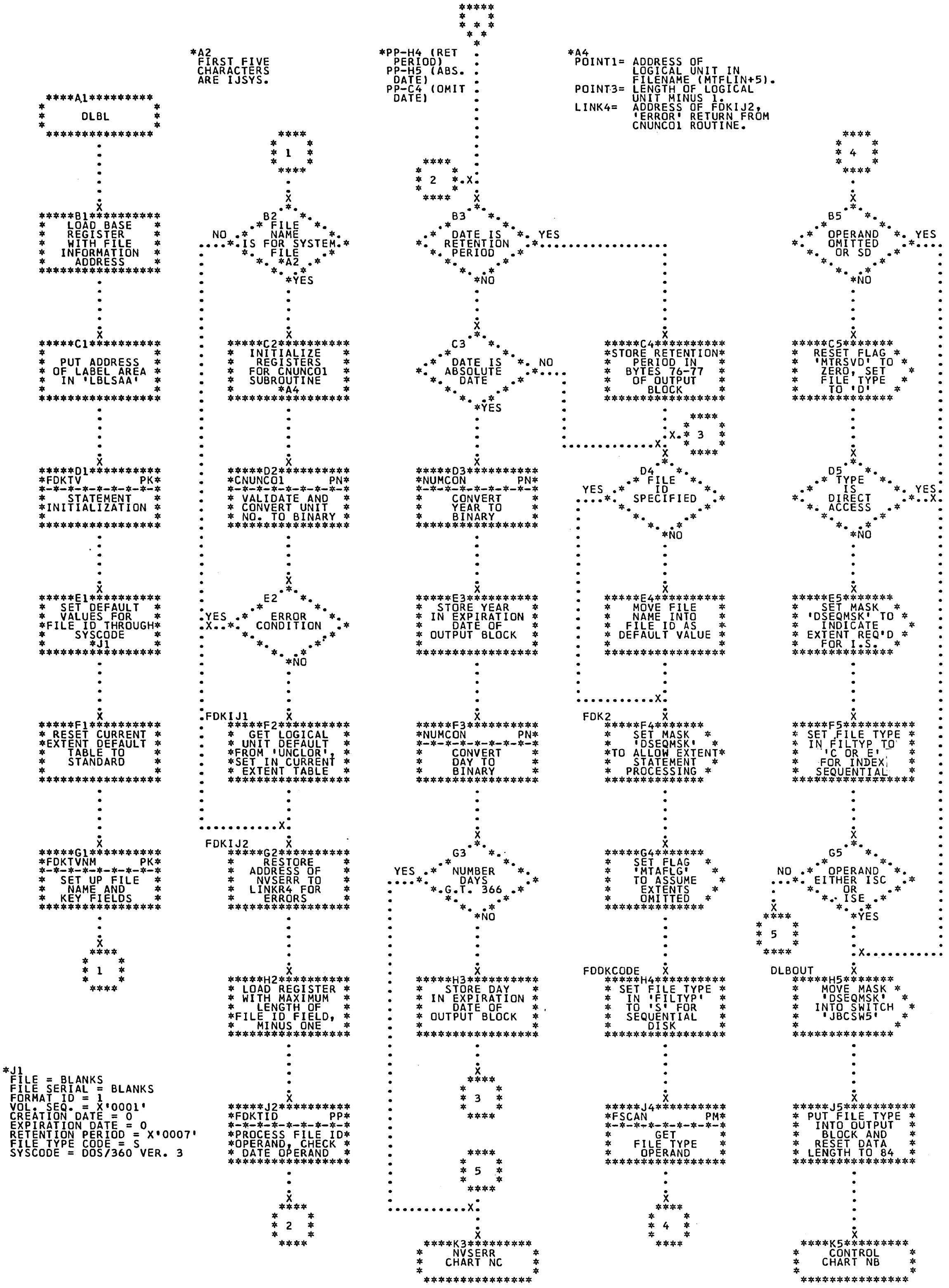
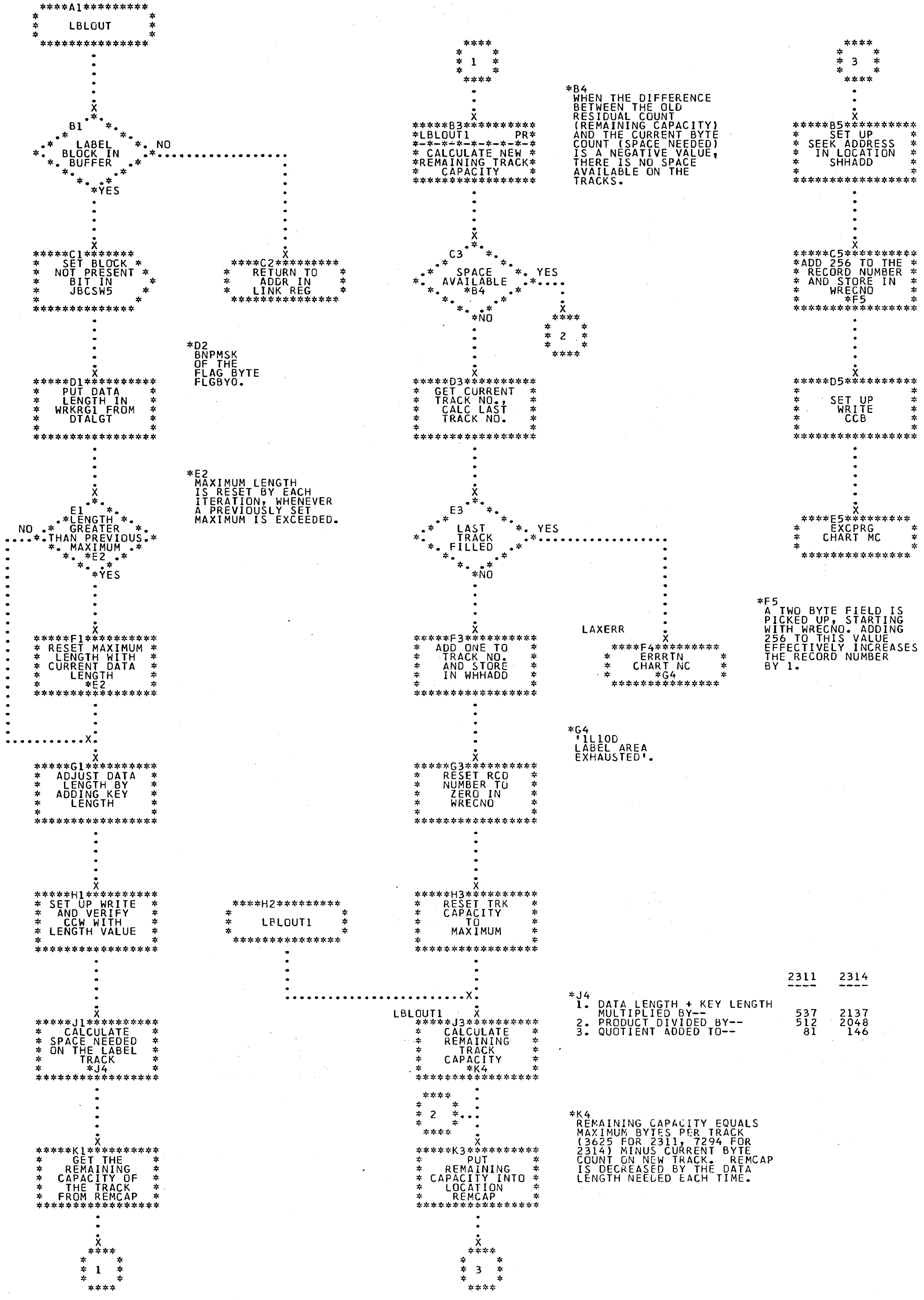

1
SCNRL2
B2
ADDRESS OPERAND CHARACTER
B1
FDKTID
FDKTID
C1
ANY MORE OPERANDS
NO
YES
C2
IS THIS CHARACTER A QUOTE
NO
YES
C3
ANY MORE CHARACTERS
YES
NO
1
D1
SCANR3 ND
GET OPERAND START ADDR
NDSCAN
D2
IS QUOTE IN COLUMN 72
YES
NO
4
E1
IS OPERAND PRESENT
NO
YES
3
E2
SET SCAN DELIMITER TO INDICATE END OF OPERAND
E3
NVSERR CHART NC
F1
LOAD SCAN LIMIT INTO POINT2
F2
IS DELIMITER A BLANK
YES
NO
2
G1
IS CHAR A QUOTE
NO
YES
G2
COL 72 IS NON-BLANK
YES
NO
NDSCAN1
G3
IS THERE AN OPERAND WITHIN THE QUOTES
NO
YES
3
H1
START SCAN WITH FIRST CHARACTER AFTER QUOTE
1
H2
DELIMITER IS A COMMA
NO
YES
4
H3
OPERAND TOO LONG
YES
NO
4
J2
COMPUTE POINTER FOR NEXT SCAN AND STORE SCAN LIMIT
2
J3
INSERT FILE ID IN OUTPUT BLOCK LEFT JUSTIFIED
3
3
FDKTDAT
B4
FSCAN PM
GET DATE OPERAND
C4
DATE OPERAND OMITTED
YES
NO
PM G1
RETURN TO LINKREG+4 FOR OMITTED DATE OPERAND.
D4
OPERAND LENGTH = 1
YES
NO
E5
RETENTION PERIOD OR ABSOLUTE DATE. FORMAT OF ABSOLUTE DATE IS '/YYDDD'.
E4
DATE OPERAND GREATER THAN 6
YES
NO
4
FDKTDAT2
F4
IS THIRD CHARACTER A SLASH F5
YES
NO
F5
GET DAY FIELD LENGTH
G4
IS OPERAND GREATER THAN 4
YES
NO
4
G5
DAY FIELD OMITTED
YES
NO
FDKTDAT1
H4
NUMCON PN
CONVERT RETENTION PER- IOD TO BINARY
H5
POINT TO START OF ABSOLUTE DATE FIELD
PM G1
RETURN ON LINK REG FOR RETENTION PERIOD.
PM G1
RETURN TO LINKREG+8 FOR ABSOLUTE DATE.

Chart PQ. $$BATTNK - DLBL Statement Processor
Refer to Chart 13.



*A2 FIRST FIVE CHARACTERS ARE IJSYS.
*PP-H4 (RET PERIOD) PP-H5 (ABS. DATE) PP-C4 (OMIT DATE)
*A4 POINT1= ADDRESS OF LOGICAL UNIT IN FILENAME (MTFLIN+5). POINT3= LENGTH OF LOGICAL UNIT MINUS 1. LINK4= ADDRESS OF FDKIJ2, 'ERROR' RETURN FROM CNUNC01 ROUTINE.
A1 DLBL
B1 LOAD BASE REGISTER WITH FILE INFORMATION ADDRESS
C1 PUT ADDRESS OF LABEL AREA IN 'LBLSAA'
D1 FDKTV PK STATEMENT INITIALIZATION
E1 SET DEFAULT VALUES FOR FILE ID THROUGH SYSCODE *J1
F1 RESET CURRENT EXTENT DEFAULT TABLE TO STANDARD
G1 FDKTVNM PK SET UP FILE NAME AND KEY FIELDS
1
1
B2 FILE NAME IS FOR SYSTEM FILE *A2
NO
YES
C2 INITIALIZE REGISTERS FOR CNUNC01 SUBROUTINE *A4
D2 CNUNC01 PN VALIDATE AND CONVERT UNIT NO. TO BINARY
E2 ERROR CONDITION
YES
NO
FDKIJ1
F2 GET LOGICAL UNIT DEFAULT FROM 'UNCLOR', SET IN CURRENT EXTENT TABLE
FDKIJ2
G2 RESTORE ADDRESS OF NVSERR TO LINKR4 FOR ERRORS
H2 LOAD REGISTER WITH MAXIMUM LENGTH OF FILE ID FIELD, MINUS ONE
J2 FDKTID PP PROCESS FILE ID OPERAND, CHECK DATE OPERAND
2
2
B3 DATE IS RETENTION PERIOD
YES
NO
C3 DATE IS ABSOLUTE DATE
NO
YES
D3 NUMCON PN CONVERT YEAR TO BINARY
E3 STORE YEAR IN EXPIRATION DATE OF OUTPUT BLOCK
F3 NUMCON PN CONVERT DAY TO BINARY
G3 NUMBER DAYS G.T. 366
YES
NO
H3 STORE DAY IN EXPIRATION DATE OF OUTPUT BLOCK
3
5
K3 NVSERR CHART NC
C4 STORE RETENTION PERIOD IN BYTES 76-77 OF OUTPUT BLOCK
3
D4 FILE ID SPECIFIED
YES
NO
E4 MOVE FILE NAME INTO FILE ID AS DEFAULT VALUE
FDK2
F4 SET MASK 'DSEQMSK' TO ALLOW EXTENT STATEMENT PROCESSING
G4 SET FLAG 'MTAFLG' TO ASSUME EXTENTS OMITTED
FDDKCODE
H4 SET FILE TYPE IN 'FILTYP' TO 'S' FOR SEQUENTIAL DISK
J4 FSCAN PM GET FILE TYPE OPERAND
4
4
B5 OPERAND OMITTED OR SD
YES
NO
C5 RESET FLAG 'MTRSVD' TO ZERO, SET FILE TYPE TO 'D'
D5 TYPE IS DIRECT ACCESS
YES
NO
E5 SET MASK 'DSEQMSK' TO INDICATE EXTENT REQ'D FOR I.S.
F5 SET FILE TYPE IN FILTYP TO 'C OR E' FOR INDEX SEQUENTIAL
G5 OPERAND EITHER ISC OR ISE
NO
YES
5
DLBOUT
H5 MOVE MASK 'DSEQMSK' INTO SWITCH 'JBCSW5'
J5 PUT FILE TYPE INTO OUTPUT BLOCK AND RESET DATA LENGTH TO 84
K5 CONTROL CHART NB
*J1
FILE = BLANKS
FILE SERIAL = BLANKS
FORMAT ID = 1
VOL. SEQ. = X'0001'
CREATION DATE = 0
EXPIRATION DATE = 0
RETENTION PERIOD = X'0007'
FILE TYPE CODE = S
SYSCODE = DOS/360 VER. 3

Chart PR. $$BATTNK - Output Label Data Subroutines
Refer to Chart 13.


A1
LBLOUT
B1
LABEL BLOCK IN BUFFER
NO
YES
C1
SET BLOCK NOT PRESENT BIT IN JBCSW5
C2
RETURN TO ADDR IN LINK REG
D1
PUT DATA LENGTH IN WRKRG1 FROM DTALGT
E1
LENGTH GREATER THAN PREVIOUS MAXIMUM
*E2
NO
YES
F1
RESET MAXIMUM LENGTH WITH CURRENT DATA LENGTH
*E2
G1
ADJUST DATA LENGTH BY ADDING KEY LENGTH
H1
SET UP WRITE AND VERIFY CCW WITH LENGTH VALUE
J1
CALCULATE SPACE NEEDED ON THE LABEL TRACK
*J4
K1
GET THE REMAINING CAPACITY OF THE TRACK FROM REMCAP
1
H2
LBLOUT1
1
B3
LBLOUT1 PR
CALCULATE NEW REMAINING TRACK CAPACITY
C3
SPACE AVAILABLE
*B4
YES
NO
2
D3
GET CURRENT TRACK NO., CALC LAST TRACK NO.
E3
LAST TRACK FILLED
YES
NO
F3
ADD ONE TO TRACK NO. AND STORE IN WHHADD
G3
RESET RCD NUMBER TO ZERO IN WRECNO
H3
RESET TRK CAPACITY TO MAXIMUM
LBLOUT1
J3
CALCULATE REMAINING TRACK CAPACITY
*K4
2
K3
PUT REMAINING CAPACITY INTO LOCATION REMCAP
3
LAXERR
F4
ERRRTN
CHART NC
*G4
3
B5
SET UP SEEK ADDRESS IN LOCATION SHHADD
C5
ADD 256 TO THE RECORD NUMBER AND STORE IN WRECNO
*F5
D5
SET UP WRITE CCB
E5
EXCPRG
CHART MC

*B4
WHEN THE DIFFERENCE BETWEEN THE OLD RESIDUAL COUNT (REMAINING CAPACITY) AND THE CURRENT BYTE COUNT (SPACE NEEDED) IS A NEGATIVE VALUE, THERE IS NO SPACE AVAILABLE ON THE TRACKS.

*D2
BNPMSK OF THE FLAG BYTE FLGBYO.

*E2
MAXIMUM LENGTH IS RESET BY EACH ITERATION, WHENEVER A PREVIOUSLY SET MAXIMUM IS EXCEEDED.

*F5
A TWO BYTE FIELD IS PICKED UP, STARTING WITH WRECNO. ADDING 256 TO THIS VALUE EFFECTIVELY INCREASES THE RECORD NUMBER BY 1.

*G4
'1L10D LABEL AREA EXHAUSTED'.

*J4

	2311	2314
1. DATA LENGTH + KEY LENGTH MULTIPLIED BY--	537	2137
2. PRODUCT DIVIDED BY--	512	2048
3. QUOTIENT ADDED TO--	81	146

*K4
REMAINING CAPACITY EQUALS MAXIMUM BYTES PER TRACK (3625 FOR 2311, 7294 FOR 2314) MINUS CURRENT BYTE COUNT ON NEW TRACK. REMCAP IS DECREASED BY THE DATA LENGTH NEEDED EACH TIME.

Chart PS. $$BATTNL - DLAB Statement Processor
Refer to Chart 14.

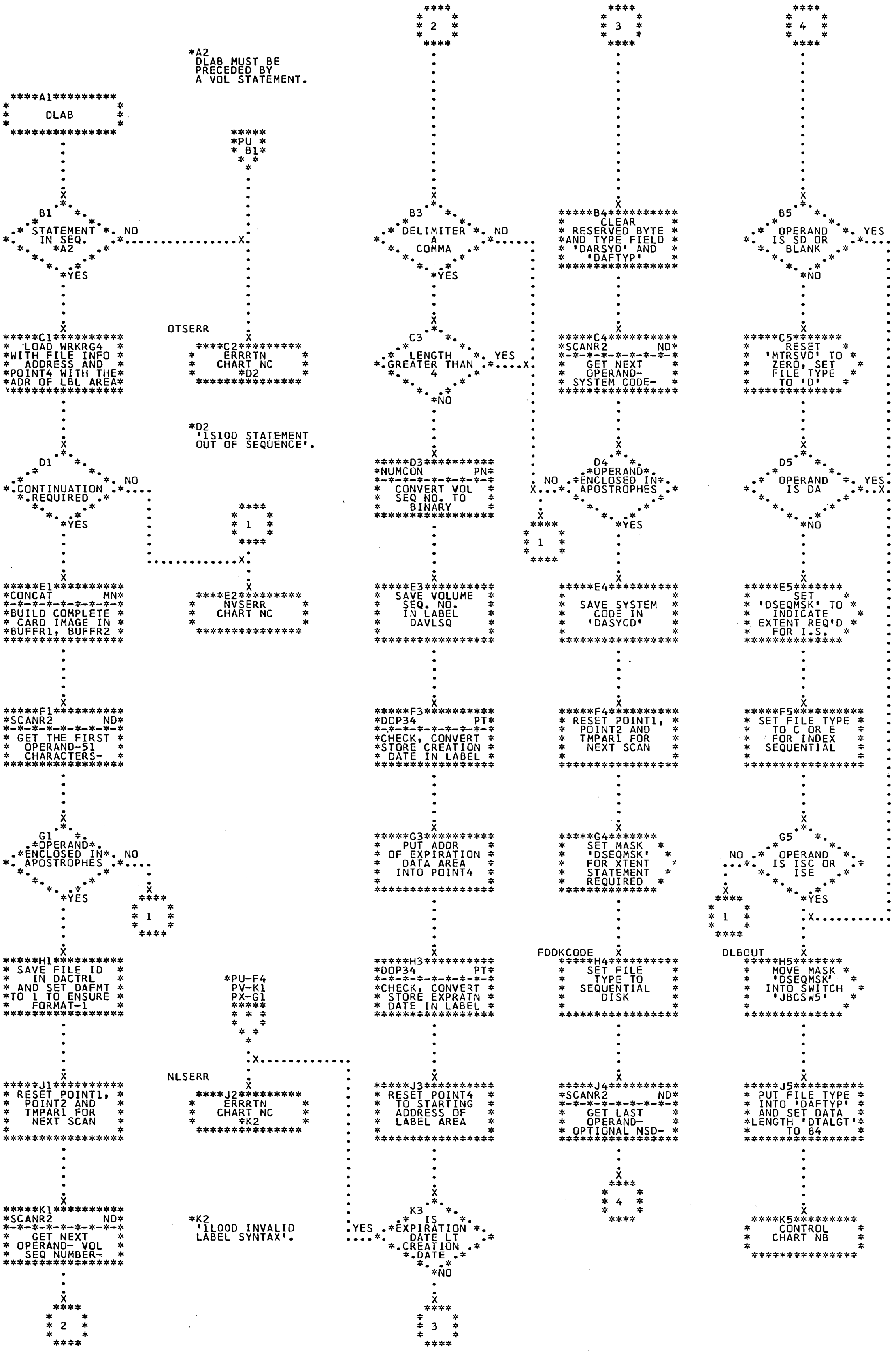

A1 DLAB
*A2 DLAB MUST BE PRECEDED BY A VOL STATEMENT.
B1 STATEMENT IN SEQ. *A2
NO
YES
PU B1
C1 LOAD WRKRG4 WITH FILE INFO ADDRESS AND POINT4 WITH THE ADR OF LBL AREA
OTSERR
C2 ERRRTN CHART NC *D2
*D2 'IS1OD STATEMENT OUT OF SEQUENCE'.
D1 CONTINUATION REQUIRED
NO
YES
1
E1 CONCAT MN BUILD COMPLETE CARD IMAGE IN BUFFR1, BUFFR2
E2 NVSERR CHART NC
F1 SCANR2 ND GET THE FIRST OPERAND-51 CHARACTERS-
G1 OPERAND ENCLOSED IN APOSTROPHES
NO
YES
1
H1 SAVE FILE ID IN DACTRL AND SET DAFMT TO 1 TO ENSURE FORMAT-1
*PU-F4 PV-K1 PX-G1
J1 RESET POINT1, POINT2 AND TMPAR1 FOR NEXT SCAN
NLSERR
J2 ERRRTN CHART NC *K2
K1 SCANR2 ND GET NEXT OPERAND- VOL SEQ NUMBER-
*K2 'ILOOD INVALID LABEL SYNTAX'.
2
2
B3 DELIMITER A COMMA
NO
YES
C3 LENGTH GREATER THAN 4
YES
NO
D3 NUMCON PN CONVERT VOL SEQ NO. TO BINARY
E3 SAVE VOLUME SEQ. NO. IN LABEL DAVLSQ
F3 DOP34 PT CHECK, CONVERT STORE CREATION DATE IN LABEL
G3 PUT ADDR OF EXPIRATION DATA AREA INTO POINT4
H3 DOP34 PT CHECK, CONVERT STORE EXPRATN DATE IN LABEL
J3 RESET POINT4 TO STARTING ADDRESS OF LABEL AREA
K3 IS EXPIRATION DATE LT CREATION DATE
YES
NO
3
3
B4 CLEAR RESERVED BYTE AND TYPE FIELD 'DARSVD' AND 'DAFTYP'
C4 SCANR2 ND GET NEXT OPERAND- SYSTEM CODE-
D4 OPERAND ENCLOSED IN APOSTROPHES
NO
YES
1
E4 SAVE SYSTEM CODE IN 'DASYCD'
F4 RESET POINT1, POINT2 AND TMPAR1 FOR NEXT SCAN
G4 SET MASK 'DSEQMSK' FOR XTENT STATEMENT REQUIRED
FDDKCODE
H4 SET FILE TYPE TO SEQUENTIAL DISK
J4 SCANR2 ND GET LAST OPERAND- OPTIONAL NSD-
4
4
B5 OPERAND IS SD OR BLANK
YES
NO
C5 RESET 'MTRSVD' TO ZERO, SET FILE TYPE TO 'D'
D5 OPERAND IS DA
YES
NO
E5 SET 'DSEQMSK' TO INDICATE EXTENT REQ'D FOR I.S.
F5 SET FILE TYPE TO C OR E FOR INDEX SEQUENTIAL
G5 OPERAND IS ISC OR ISE
NO
YES
1
DLBOUT
H5 MOVE MASK 'DSEQMSK' INTO SWITCH 'JBCSW5'
J5 PUT FILE TYPE INTO 'DAFTYP' AND SET DATA LENGTH 'DTALGT' TO 84
K5 CONTROL CHART NB

Chart PT. $$BATTNL - Extract Operand Routine
Refer to Chart 14.

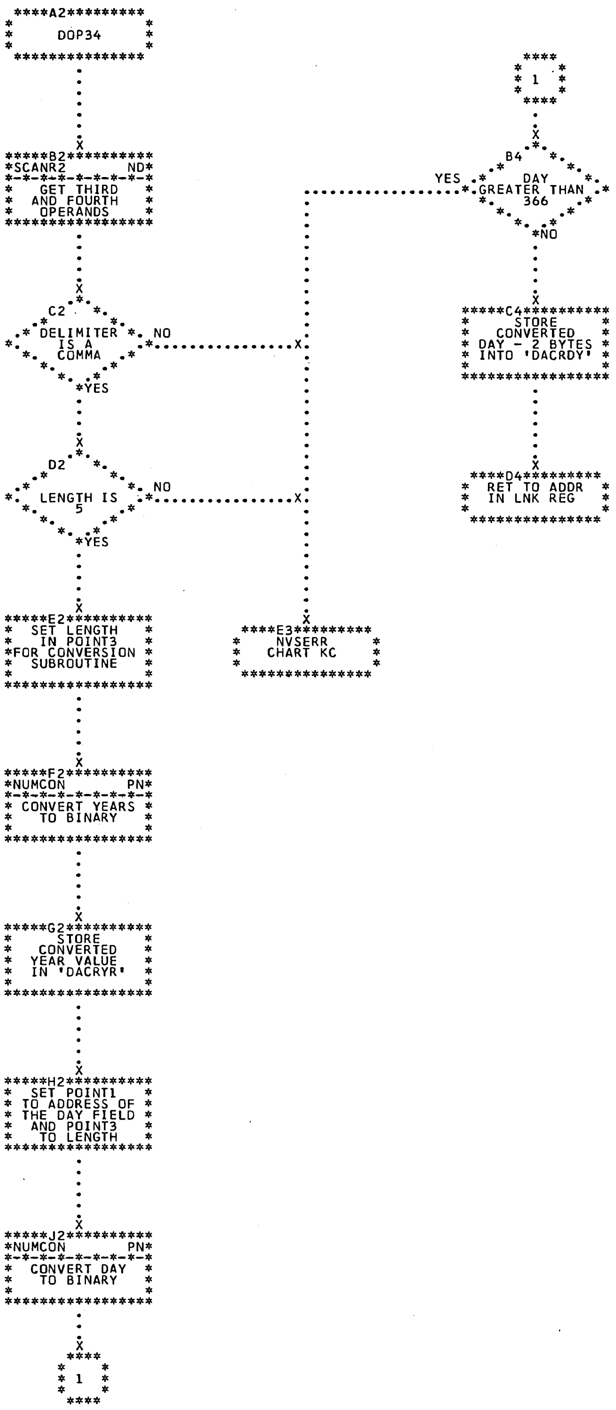
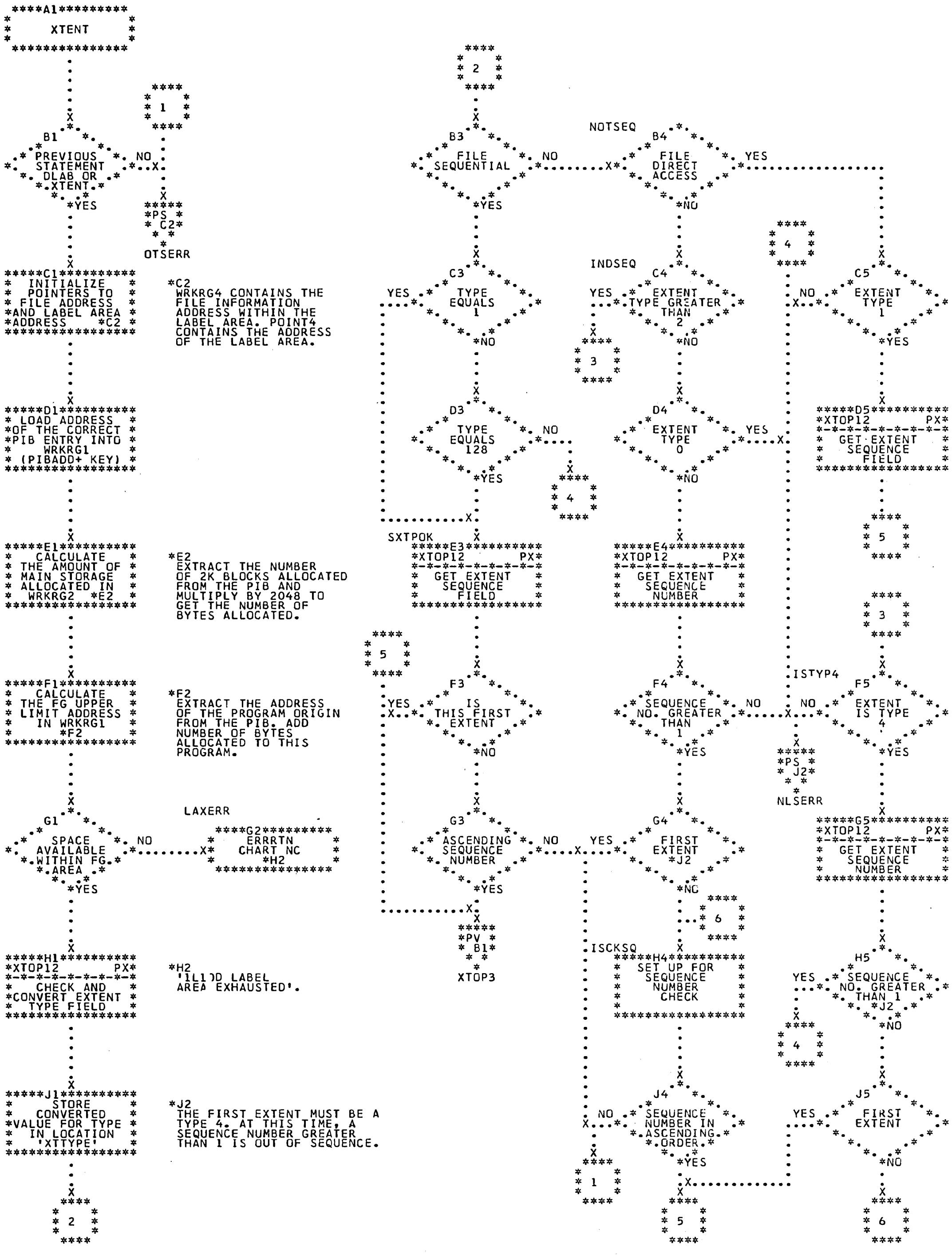

A2
DOP34
B2
SCANR2 ND
GET THIRD AND FOURTH OPERANDS
C2
DELIMITER IS A COMMA
NO
YES
D2
LENGTH IS 5
NO
YES
E2
SET LENGTH IN POINT3 FOR CONVERSION SUBROUTINE
E3
NVSERR CHART KC
F2
NUMCON PN
CONVERT YEARS TO BINARY
G2
STORE CONVERTED YEAR VALUE IN 'DACRYR'
H2
SET POINT1 TO ADDRESS OF THE DAY FIELD AND POINT3 TO LENGTH
J2
NUMCON PN
CONVERT DAY TO BINARY
1
1
B4
DAY GREATER THAN 366
YES
NO
C4
STORE CONVERTED DAY - 2 BYTES INTO 'DACRDY'
D4
RET TO ADDR IN LNK REG

Chart PU. $$BATTNL - XTENT Statement Processor (Part 1 of 3)
Refer to Chart 14.



A1
XTENT
1
B1
PREVIOUS STATEMENT DLAB OR XTENT.
NO
YES
PS C2
OTSERR
C1
INITIALIZE POINTERS TO FILE ADDRESS AND LABEL AREA ADDRESS *C2
D1
LOAD ADDRESS OF THE CORRECT PIB ENTRY INTO WRKRG1 (PIBADD+ KEY)
E1
CALCULATE THE AMOUNT OF MAIN STORAGE ALLOCATED IN WRKRG2 *E2
F1
CALCULATE THE FG UPPER LIMIT ADDRESS IN WRKRG1 *F2
G1
SPACE AVAILABLE WITHIN FG. AREA
NO
YES
LAXERR
G2
ERRRTN CHART NC *H2
H1
XTOP12 PX
CHECK AND CONVERT EXTENT TYPE FIELD
J1
STORE CONVERTED VALUE FOR TYPE IN LOCATION 'XTTYPE'
2
2
B3
FILE SEQUENTIAL
NO
YES
NOTSEQ
B4
FILE DIRECT ACCESS
YES
NO
C3
TYPE EQUALS 1
YES
NO
INDSEQ
C4
EXTENT TYPE GREATER THAN 2
YES
NO
3
4
C5
EXTENT TYPE 1
NO
YES
D3
TYPE EQUALS 128
NO
YES
4
D4
EXTENT TYPE 0
YES
NO
D5
XTOP12 PX
GET EXTENT SEQUENCE FIELD
5
SXTPOK
E3
XTOP12 PX
GET EXTENT SEQUENCE FIELD
E4
XTOP12 PX
GET EXTENT SEQUENCE NUMBER
3
5
F3
IS THIS FIRST EXTENT
YES
NO
F4
SEQUENCE NO. GREATER THAN 1
NO
YES
ISTYP4
F5
EXTENT IS TYPE 4
NO
YES
PS J2
NLSERR
G3
ASCENDING SEQUENCE NUMBER
NO
YES
G4
FIRST EXTENT *J2
YES
NO
6
G5
XTOP12 PX
GET EXTENT SEQUENCE NUMBER
PV B1
XTOP3
ISCKSQ
H4
SET UP FOR SEQUENCE NUMBER CHECK
H5
SEQUENCE NO. GREATER THAN 1 *J2
YES
NO
4
J4
SEQUENCE NUMBER IN ASCENDING ORDER
NO
YES
1
5
J5
FIRST EXTENT
YES
NO
6

*C2
WRKRG4 CONTAINS THE FILE INFORMATION ADDRESS WITHIN THE LABEL AREA. POINT4 CONTAINS THE ADDRESS OF THE LABEL AREA.

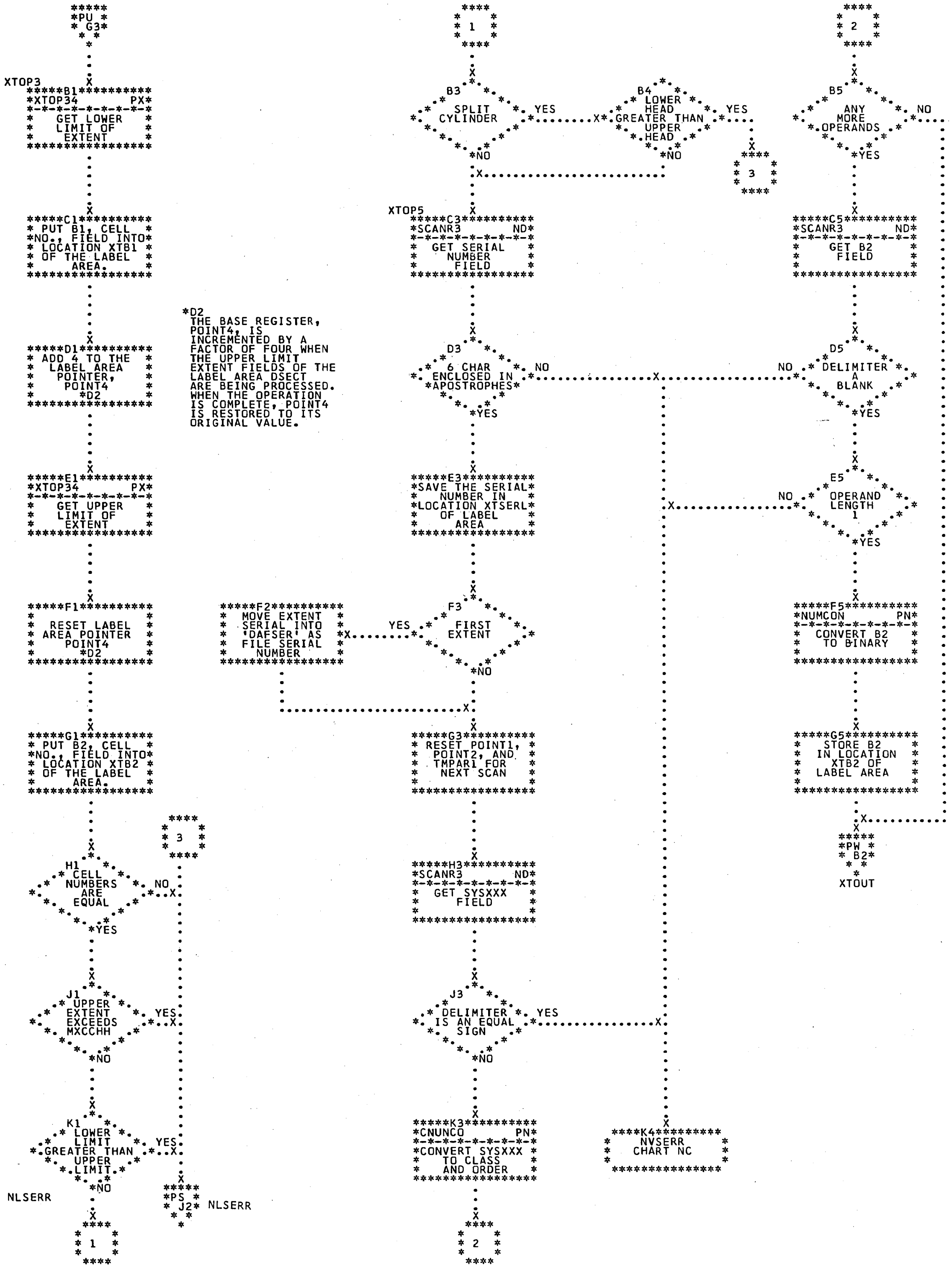
*E2
EXTRACT THE NUMBER OF 2K BLOCKS ALLOCATED FROM THE PIB AND MULTIPLY BY 2048 TO GET THE NUMBER OF BYTES ALLOCATED.

*F2
EXTRACT THE ADDRESS OF THE PROGRAM ORIGIN FROM THE PIB. ADD NUMBER OF BYTES ALLOCATED TO THIS PROGRAM.

*H2
'1L1)D LABEL AREA EXHAUSTED'.

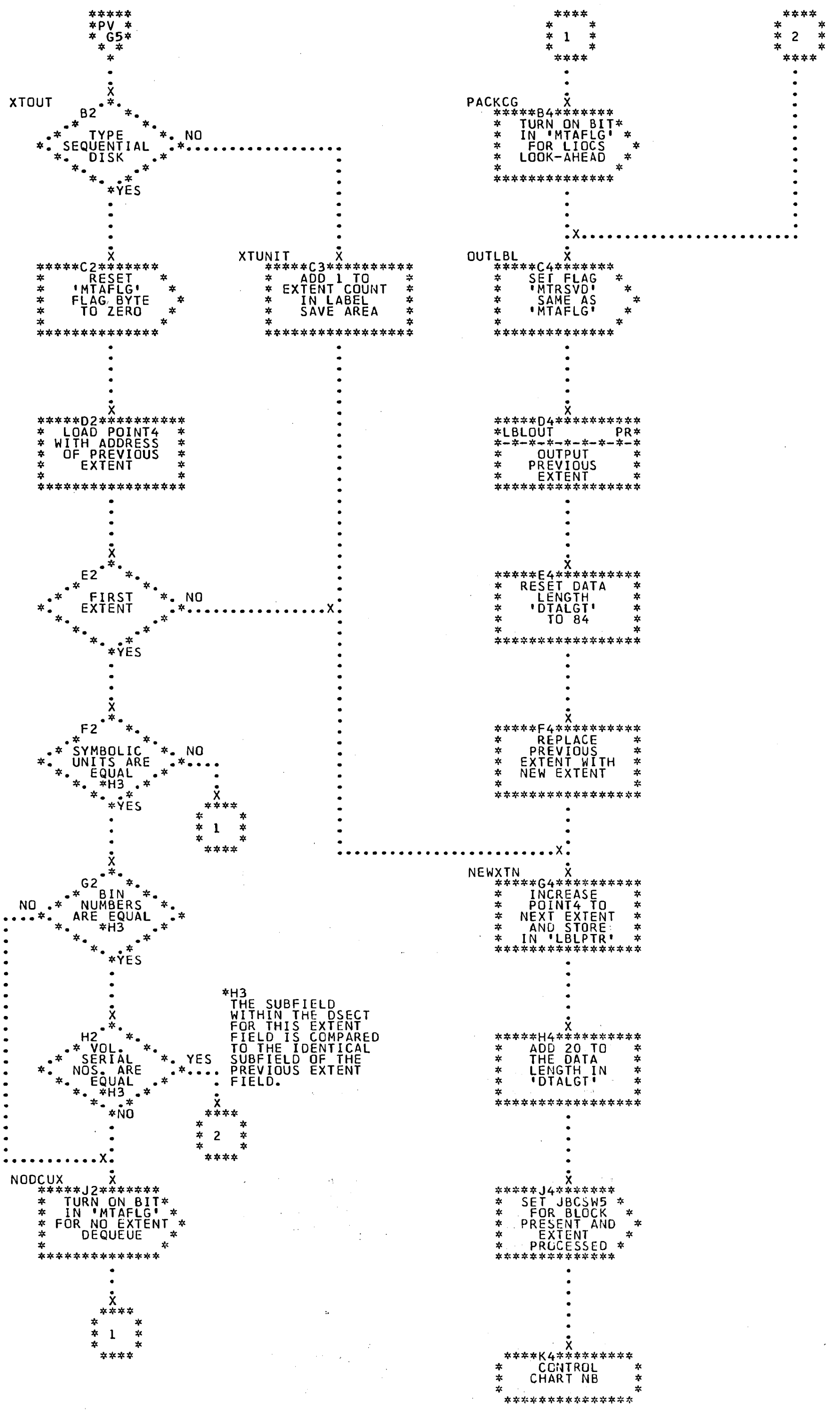
*J2
THE FIRST EXTENT MUST BE A TYPE 4. AT THIS TIME, A SEQUENCE NUMBER GREATER THAN 1 IS OUT OF SEQUENCE.

Chart PV. $$BATTNL - XTENT Statement Processor (Part 2 of 3)
Refer to Chart 14.



PU G3
XTOP3
B1 XTOP34 PX
GET LOWER LIMIT OF EXTENT
C1 PUT B1, CELL NO., FIELD INTO LOCATION XTB1 OF THE LABEL AREA.
D1 ADD 4 TO THE LABEL AREA POINTER, POINT4 *D2
*D2 THE BASE REGISTER, POINT4, IS INCREMENTED BY A FACTOR OF FOUR WHEN THE UPPER LIMIT EXTENT FIELDS OF THE LABEL AREA DSECT ARE BEING PROCESSED. WHEN THE OPERATION IS COMPLETE, POINT4 IS RESTORED TO ITS ORIGINAL VALUE.
E1 XTOP34 PX
GET UPPER LIMIT OF EXTENT
F1 RESET LABEL AREA POINTER POINT4 *D2
G1 PUT B2, CELL NO., FIELD INTO LOCATION XTB2 OF THE LABEL AREA.
3
H1 CELL NUMBERS ARE EQUAL
NO
YES
J1 UPPER EXTENT EXCEEDS MXCCHH
YES
NO
K1 LOWER LIMIT GREATER THAN UPPER LIMIT.
YES
NO
NLSERR
1
PS J2 NLSERR
1
B3 SPLIT CYLINDER
YES
NO
B4 LOWER HEAD GREATER THAN UPPER HEAD
YES
NO
3
XTOP5
C3 SCANR3 ND
GET SERIAL NUMBER FIELD
D3 6 CHAR ENCLOSED IN APOSTROPHES
NO
YES
E3 SAVE THE SERIAL NUMBER IN LOCATION XTSERL OF LABEL AREA
F2 MOVE EXTENT SERIAL INTO 'DAFSER' AS FILE SERIAL NUMBER
YES
F3 FIRST EXTENT
NO
G3 RESET POINT1, POINT2, AND TMPAR1 FOR NEXT SCAN
H3 SCANR3 ND
GET SYSXXX FIELD
J3 DELIMITER IS AN EQUAL SIGN
YES
NO
K3 CNUNCO PN
CONVERT SYSXXX TO CLASS AND ORDER
2
K4 NVSERR CHART NC
2
B5 ANY MORE OPERANDS
NO
YES
C5 SCANR3 ND
GET B2 FIELD
D5 DELIMITER A BLANK
NO
YES
E5 OPERAND LENGTH 1
NO
YES
F5 NUMCON PN
CONVERT B2 TO BINARY
G5 STORE B2 IN LOCATION XTB2 OF LABEL AREA
PW B2
XTOUT

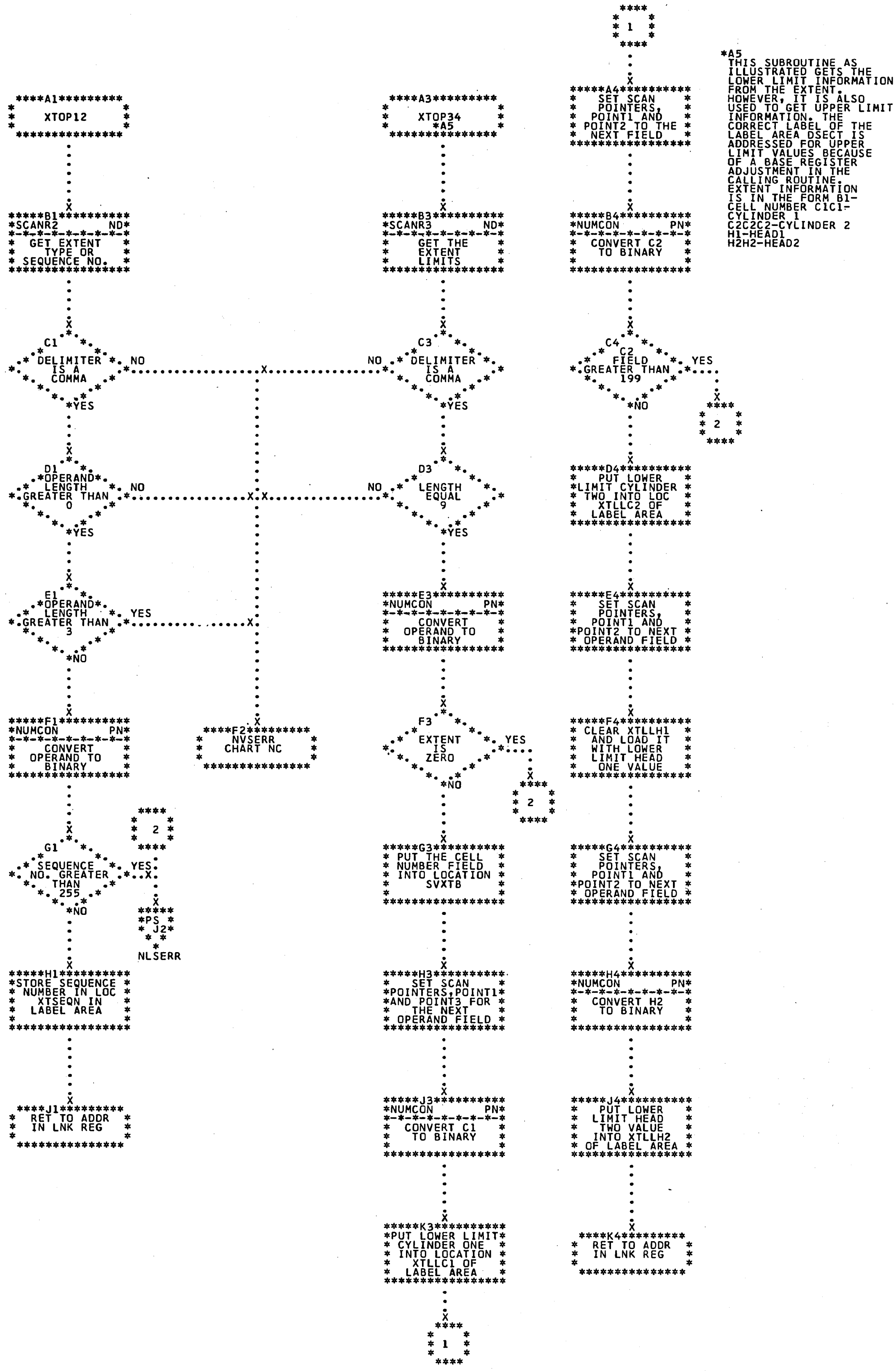
Chart PW. $$BATTNL - XTENT Statement Processor (Part 3 of 3)
Refer to Chart 14.



PV G5
XTOUT
B2
TYPE SEQUENTIAL DISK
NO
YES
C2
RESET 'MTAFLG' FLAG BYTE TO ZERO
D2
LOAD POINT4 WITH ADDRESS OF PREVIOUS EXTENT
E2
FIRST EXTENT
NO
YES
F2
SYMBOLIC UNITS ARE EQUAL *H3
NO
1
YES
G2
BIN NUMBERS ARE EQUAL *H3
NO
YES
H2
VOL. SERIAL NOS. ARE EQUAL *H3
YES
2
NO
NODCUX
J2
TURN ON BIT IN 'MTAFLG' FOR NO EXTENT DEQUEUE
1
XTUNIT
C3
ADD 1 TO EXTENT COUNT IN LABEL SAVE AREA
1
PACKCG
B4
TURN ON BIT IN 'MTAFLG' FOR LIOCS LOOK-AHEAD
2
OUTLBL
C4
SET FLAG 'MTRSVD' SAME AS 'MTAFLG'
D4
LBLOUT PR
OUTPUT PREVIOUS EXTENT
E4
RESET DATA LENGTH 'DTALGT' TO 84
F4
REPLACE PREVIOUS EXTENT WITH NEW EXTENT
NEWXTN
G4
INCREASE POINT4 TO NEXT EXTENT AND STORE IN 'LBLPTR'
H4
ADD 20 TO THE DATA LENGTH IN 'DTALGT'
J4
SET JBCSW5 FOR BLOCK PRESENT AND EXTENT PROCESSED
K4
CONTROL CHART NB

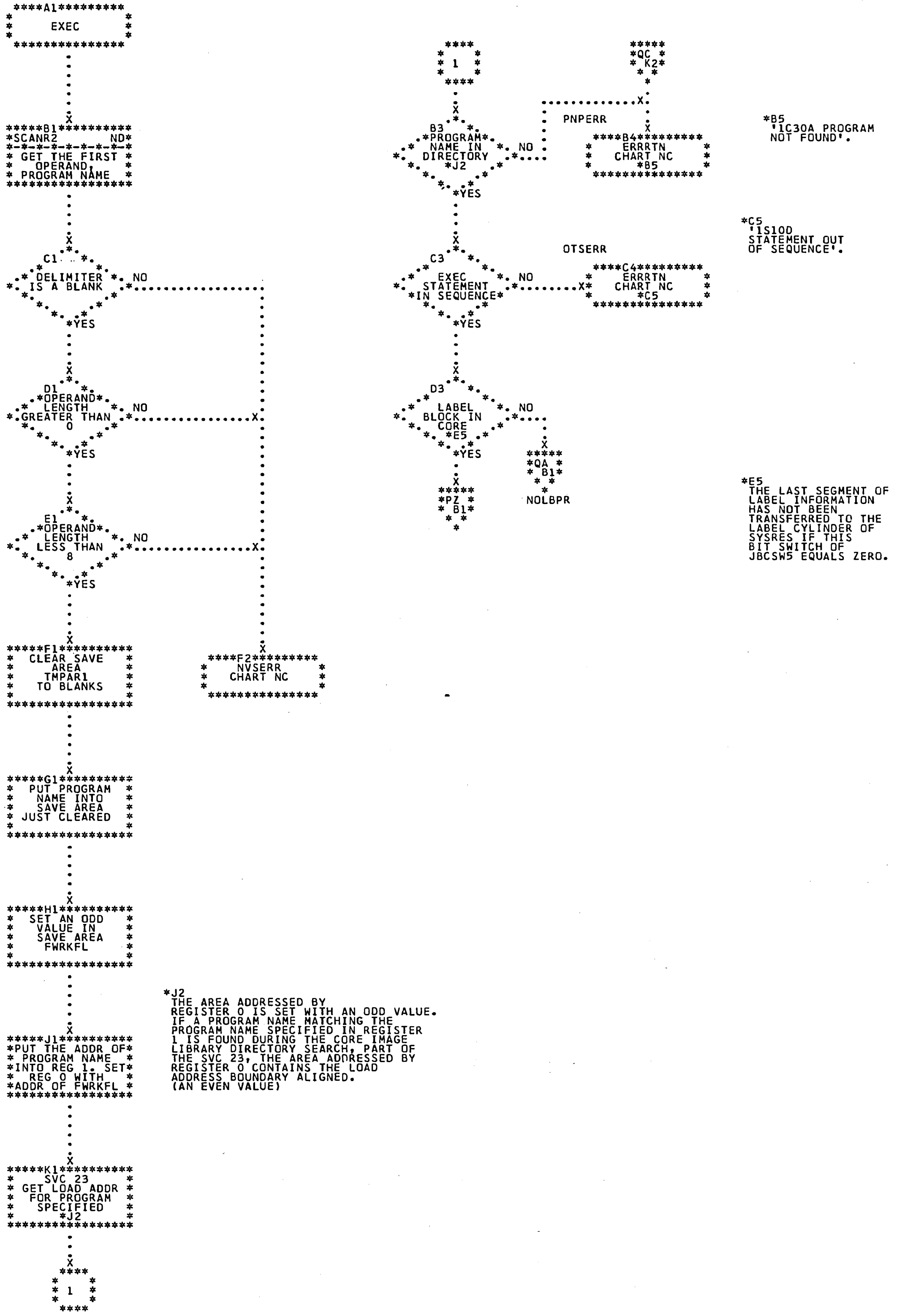
*H3
THE SUBFIELD WITHIN THE DSECT FOR THIS EXTENT FIELD IS COMPARED TO THE IDENTICAL SUBFIELD OF THE PREVIOUS EXTENT FIELD.

Chart PX. $$BATTNL - XTENT Processor Subroutines
Refer to Chart 14.


1
A4 SET SCAN POINTERS, POINT1 AND POINT2 TO THE NEXT FIELD
A1 XTOP12
A3 XTOP34 *A5
B1 SCANR2 ND GET EXTENT TYPE OR SEQUENCE NO.
B3 SCANR3 ND GET THE EXTENT LIMITS
B4 NUMCON PN CONVERT C2 TO BINARY
C1 DELIMITER IS A COMMA
NO
YES
C3 DELIMITER IS A COMMA
NO
YES
C4 C2 FIELD GREATER THAN 199
YES
NO
2
D1 OPERAND LENGTH GREATER THAN 0
NO
YES
D3 LENGTH EQUAL 9
NO
YES
D4 PUT LOWER LIMIT CYLINDER TWO INTO LOC XTLLC2 OF LABEL AREA
E1 OPERAND LENGTH GREATER THAN 3
YES
NO
E3 NUMCON PN CONVERT OPERAND TO BINARY
E4 SET SCAN POINTERS, POINT1 AND POINT2 TO NEXT OPERAND FIELD
F1 NUMCON PN CONVERT OPERAND TO BINARY
F2 NVSERR CHART NC
F3 EXTENT IS ZERO
YES
NO
2
F4 CLEAR XTLLH1 AND LOAD IT WITH LOWER LIMIT HEAD ONE VALUE
2
G1 SEQUENCE NO. GREATER THAN 255
YES
NO
PS J2
NLSERR
G3 PUT THE CELL NUMBER FIELD INTO LOCATION SVXTB
G4 SET SCAN POINTERS, POINT1 AND POINT2 TO NEXT OPERAND FIELD
H1 STORE SEQUENCE NUMBER IN LOC XTSEQN IN LABEL AREA
H3 SET SCAN POINTERS,POINT1 AND POINT3 FOR THE NEXT OPERAND FIELD
H4 NUMCON PN CONVERT H2 TO BINARY
J1 RET TO ADDR IN LNK REG
J3 NUMCON PN CONVERT C1 TO BINARY
J4 PUT LOWER LIMIT HEAD TWO VALUE INTO XTLLH2 OF LABEL AREA
K3 PUT LOWER LIMIT CYLINDER ONE INTO LOCATION XTLLC1 OF LABEL AREA
K4 RET TO ADDR IN LNK REG
1

*A5
THIS SUBROUTINE AS
ILLUSTRATED GETS THE
LOWER LIMIT INFORMATION
FROM THE EXTENT.
HOWEVER, IT IS ALSO
USED TO GET UPPER LIMIT
INFORMATION. THE
CORRECT LABEL OF THE
LABEL AREA DSECT IS
ADDRESSED FOR UPPER
LIMIT VALUES BECAUSE
OF A BASE REGISTER
ADJUSTMENT IN THE
CALLING ROUTINE.
EXTENT INFORMATION
IS IN THE FORM B1-
CELL NUMBER C1C1-
CYLINDER 1
C2C2C2-CYLINDER 2
H1-HEAD1
H2H2-HEAD2

Chart PY. $$BATTNM - EXEC Statement Processor
Refer to Chart 14.


A1
EXEC
B1
SCANR2 ND
GET THE FIRST OPERAND, PROGRAM NAME
C1
DELIMITER IS A BLANK
NO
YES
D1
OPERAND LENGTH GREATER THAN 0
NO
YES
E1
OPERAND LENGTH LESS THAN 8
NO
YES
F1
CLEAR SAVE AREA TMPAR1 TO BLANKS
F2
NVSERR CHART NC
G1
PUT PROGRAM NAME INTO SAVE AREA JUST CLEARED
H1
SET AN ODD VALUE IN SAVE AREA FWRKFL
J1
PUT THE ADDR OF PROGRAM NAME INTO REG 1. SET REG 0 WITH ADDR OF FWRKFL
K1
SVC 23 GET LOAD ADDR FOR PROGRAM SPECIFIED
*J2
1
1
QC K2
B3
PROGRAM NAME IN DIRECTORY
*J2
NO
YES
PNPERR
B4
ERRRTN CHART NC
*B5
C3
EXEC STATEMENT IN SEQUENCE
NO
YES
OTSERR
C4
ERRRTN CHART NC
*C5
D3
LABEL BLOCK IN CORE
*E5
NO
YES
QA B1
NOLBPR
PZ B1

*B5
'1C30A PROGRAM NOT FOUND'.

*C5
'1S10D STATEMENT OUT OF SEQUENCE'.

*E5
THE LAST SEGMENT OF LABEL INFORMATION HAS NOT BEEN TRANSFERRED TO THE LABEL CYLINDER OF SYSRES IF THIS BIT SWITCH OF JBCSW5 EQUALS ZERO.

*J2
THE AREA ADDRESSED BY REGISTER 0 IS SET WITH AN ODD VALUE. IF A PROGRAM NAME MATCHING THE PROGRAM NAME SPECIFIED IN REGISTER 1 IS FOUND DURING THE CORE IMAGE LIBRARY DIRECTORY SEARCH, PART OF THE SVC 23, THE AREA ADDRESSED BY REGISTER 0 CONTAINS THE LOAD ADDRESS BOUNDARY ALIGNED. (AN EVEN VALUE)

Chart PZ. $$BATTNM - Output Last Block of Label Data
Refer to Chart 14.

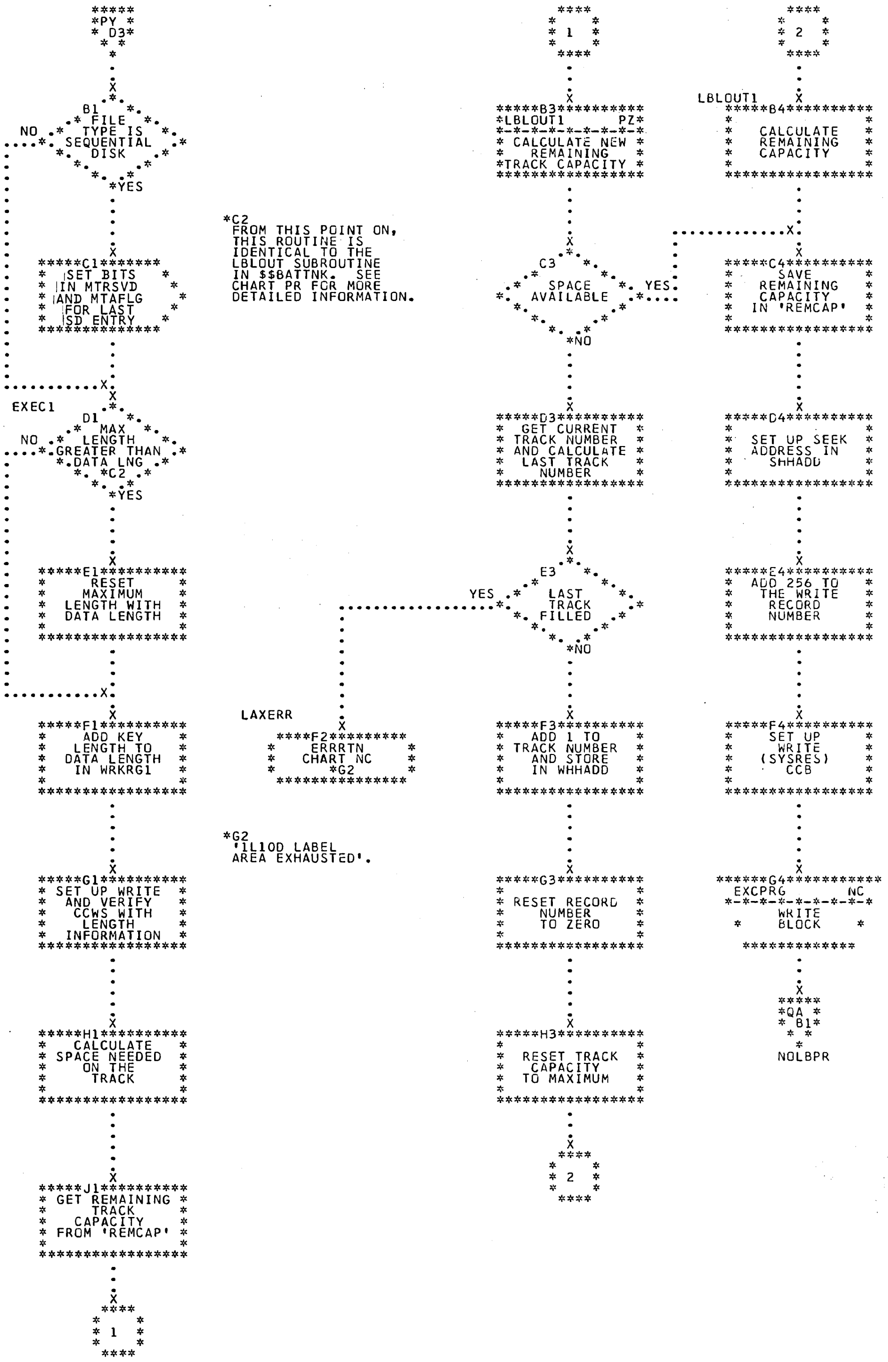
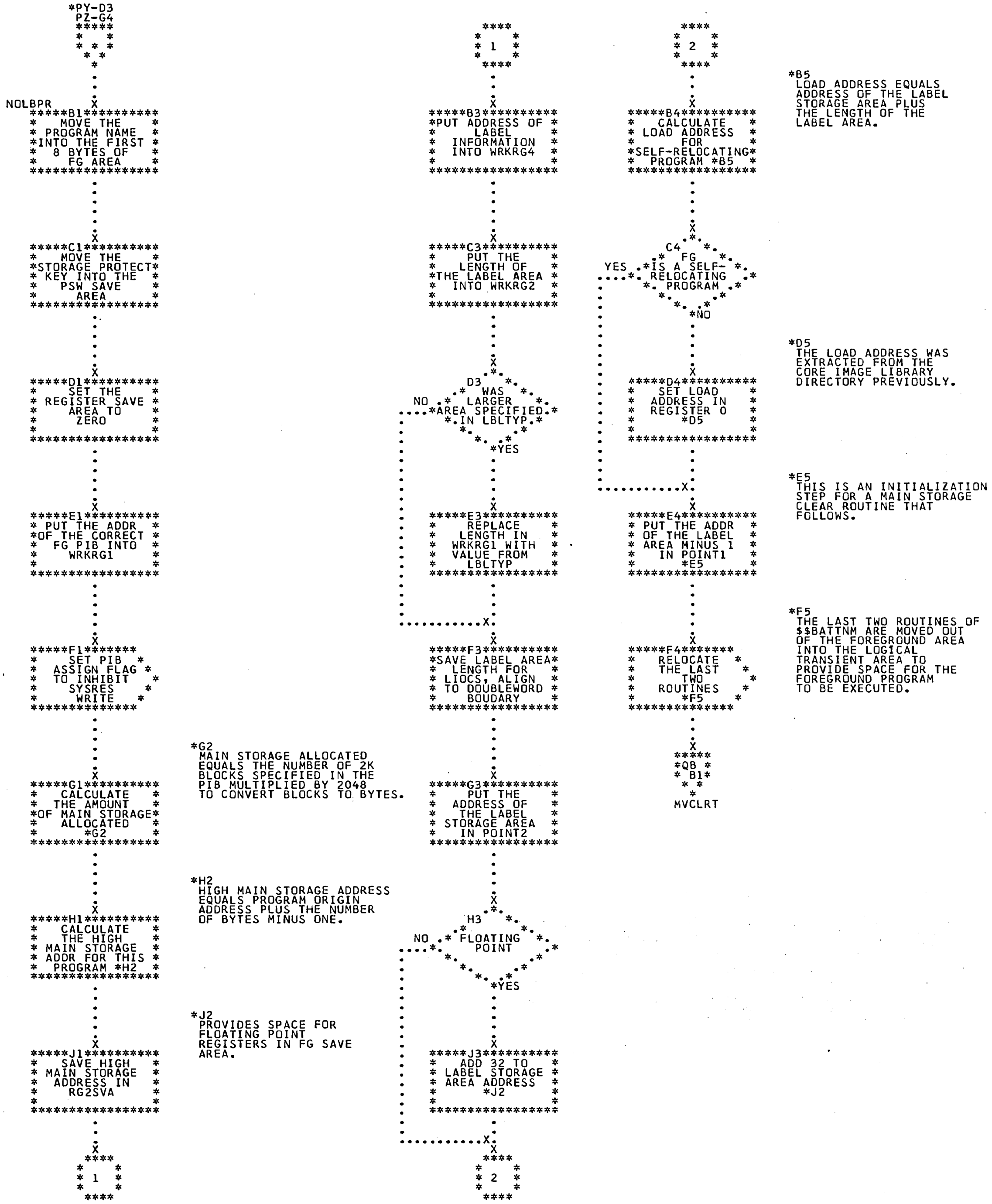

PY D3
B1 FILE TYPE IS SEQUENTIAL DISK
NO
YES
C1 SET BITS IN MTRSVD AND MTAFLG FOR LAST ISD ENTRY
EXEC1
D1 MAX LENGTH GREATER THAN DATA LNG *C2
NO
YES
E1 RESET MAXIMUM LENGTH WITH DATA LENGTH
F1 ADD KEY LENGTH TO DATA LENGTH IN WRKRG1
G1 SET UP WRITE AND VERIFY CCWS WITH LENGTH INFORMATION
H1 CALCULATE SPACE NEEDED ON THE TRACK
J1 GET REMAINING TRACK CAPACITY FROM 'REMCAP'
1
*C2
FROM THIS POINT ON, THIS ROUTINE IS IDENTICAL TO THE LBLOUT SUBROUTINE IN $$BATTNK. SEE CHART PR FOR MORE DETAILED INFORMATION.
LAXERR
F2 ERRRTN CHART NC *G2
*G2
'1L10D LABEL AREA EXHAUSTED'.
1
B3 LBLOUT1 PZ
CALCULATE NEW REMAINING TRACK CAPACITY
C3 SPACE AVAILABLE
YES
NO
D3 GET CURRENT TRACK NUMBER AND CALCULATE LAST TRACK NUMBER
E3 LAST TRACK FILLED
YES
NO
F3 ADD 1 TO TRACK NUMBER AND STORE IN WHHADD
G3 RESET RECORD NUMBER TO ZERO
H3 RESET TRACK CAPACITY TO MAXIMUM
2
2
LBLOUT1
B4 CALCULATE REMAINING CAPACITY
C4 SAVE REMAINING CAPACITY IN 'REMCAP'
D4 SET UP SEEK ADDRESS IN SHHADD
E4 ADD 256 TO THE WRITE RECORD NUMBER
F4 SET UP WRITE (SYSRES) CCB
G4 EXCPRG NC
WRITE BLOCK
QA B1
NOLBPR

Chart QA. $$BATTNM - Move Last Block Routine
Refer to Chart 14.


*PY-D3
PZ-G4
NOLBPR
B1
MOVE THE PROGRAM NAME INTO THE FIRST 8 BYTES OF FG AREA
C1
MOVE THE STORAGE PROTECT KEY INTO THE PSW SAVE AREA
D1
SET THE REGISTER SAVE AREA TO ZERO
E1
PUT THE ADDR OF THE CORRECT FG PIB INTO WRKRG1
F1
SET PIB ASSIGN FLAG TO INHIBIT SYSRES WRITE
G1
CALCULATE THE AMOUNT OF MAIN STORAGE ALLOCATED *G2
H1
CALCULATE THE HIGH MAIN STORAGE ADDR FOR THIS PROGRAM *H2
J1
SAVE HIGH MAIN STORAGE ADDRESS IN RG2SVA
1
1
B3
PUT ADDRESS OF LABEL INFORMATION INTO WRKRG4
C3
PUT THE LENGTH OF THE LABEL AREA INTO WRKRG2
D3
WAS LARGER AREA SPECIFIED IN LBLTYP
NO
YES
E3
REPLACE LENGTH IN WRKRG1 WITH VALUE FROM LBLTYP
F3
SAVE LABEL AREA LENGTH FOR LIOCS, ALIGN TO DOUBLEWORD BOUDARY
G3
PUT THE ADDRESS OF THE LABEL STORAGE AREA IN POINT2
H3
FLOATING POINT
NO
YES
J3
ADD 32 TO LABEL STORAGE AREA ADDRESS *J2
2
2
B4
CALCULATE LOAD ADDRESS FOR SELF-RELOCATING PROGRAM *B5
C4
FG IS A SELF-RELOCATING PROGRAM
YES
NO
D4
SET LOAD ADDRESS IN REGISTER 0 *D5
E4
PUT THE ADDR OF THE LABEL AREA MINUS 1 IN POINT1 *E5
F4
RELOCATE THE LAST TWO ROUTINES *F5
QB B1
MVCLRT

*B5
LOAD ADDRESS EQUALS ADDRESS OF THE LABEL STORAGE AREA PLUS THE LENGTH OF THE LABEL AREA.

*D5
THE LOAD ADDRESS WAS EXTRACTED FROM THE CORE IMAGE LIBRARY DIRECTORY PREVIOUSLY.

*E5
THIS IS AN INITIALIZATION STEP FOR A MAIN STORAGE CLEAR ROUTINE THAT FOLLOWS.

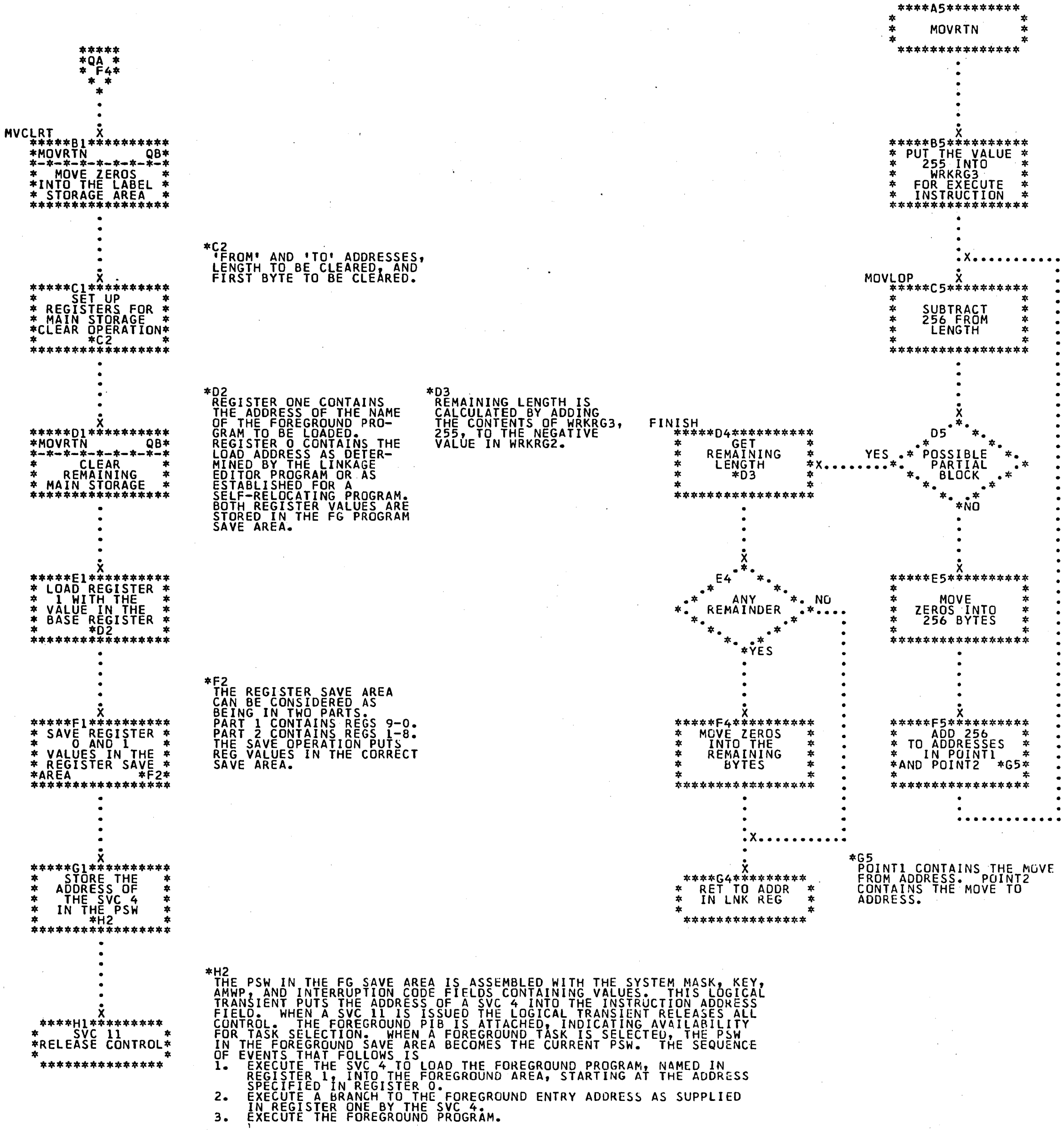
*F5
THE LAST TWO ROUTINES OF $$BATTNM ARE MOVED OUT OF THE FOREGROUND AREA INTO THE LOGICAL TRANSIENT AREA TO PROVIDE SPACE FOR THE FOREGROUND PROGRAM TO BE EXECUTED.

*G2
MAIN STORAGE ALLOCATED EQUALS THE NUMBER OF 2K BLOCKS SPECIFIED IN THE PIB MULTIPLIED BY 2048 TO CONVERT BLOCKS TO BYTES.

*H2
HIGH MAIN STORAGE ADDRESS EQUALS PROGRAM ORIGIN ADDRESS PLUS THE NUMBER OF BYTES MINUS ONE.

*J2
PROVIDES SPACE FOR FLOATING POINT REGISTERS IN FG SAVE AREA.

Chart QB. $$BATTNM - Move Subroutine and Initialize For FG Program Load Routine Refer to Chart 14.


QA F4
MVCLRT
B1 MOVRTN QB
MOVE ZEROS INTO THE LABEL STORAGE AREA
C1
SET UP REGISTERS FOR MAIN STORAGE CLEAR OPERATION *C2
D1 MOVRTN QB
CLEAR REMAINING MAIN STORAGE
E1
LOAD REGISTER 1 WITH THE VALUE IN THE BASE REGISTER *D2
F1
SAVE REGISTER 0 AND 1 VALUES IN THE REGISTER SAVE AREA *F2
G1
STORE THE ADDRESS OF THE SVC 4 IN THE PSW *H2
H1
SVC 11 RELEASE CONTROL
A5
MOVRTN
B5
PUT THE VALUE 255 INTO WRKRG3 FOR EXECUTE INSTRUCTION
MOVLOP
C5
SUBTRACT 256 FROM LENGTH
D5
POSSIBLE PARTIAL BLOCK
YES
NO
FINISH
D4
GET REMAINING LENGTH *D3
E4
ANY REMAINDER
NO
YES
E5
MOVE ZEROS INTO 256 BYTES
F4
MOVE ZEROS INTO THE REMAINING BYTES
F5
ADD 256 TO ADDRESSES IN POINT1 AND POINT2 *G5
G4
RET TO ADDR IN LNK REG

*C2
'FROM' AND 'TO' ADDRESSES, LENGTH TO BE CLEARED, AND FIRST BYTE TO BE CLEARED.

*D2
REGISTER ONE CONTAINS THE ADDRESS OF THE NAME OF THE FOREGROUND PROGRAM TO BE LOADED. REGISTER 0 CONTAINS THE LOAD ADDRESS AS DETERMINED BY THE LINKAGE EDITOR PROGRAM OR AS ESTABLISHED FOR A SELF-RELOCATING PROGRAM. BOTH REGISTER VALUES ARE STORED IN THE FG PROGRAM SAVE AREA.

*D3
REMAINING LENGTH IS CALCULATED BY ADDING THE CONTENTS OF WRKRG3, 255, TO THE NEGATIVE VALUE IN WRKRG2.

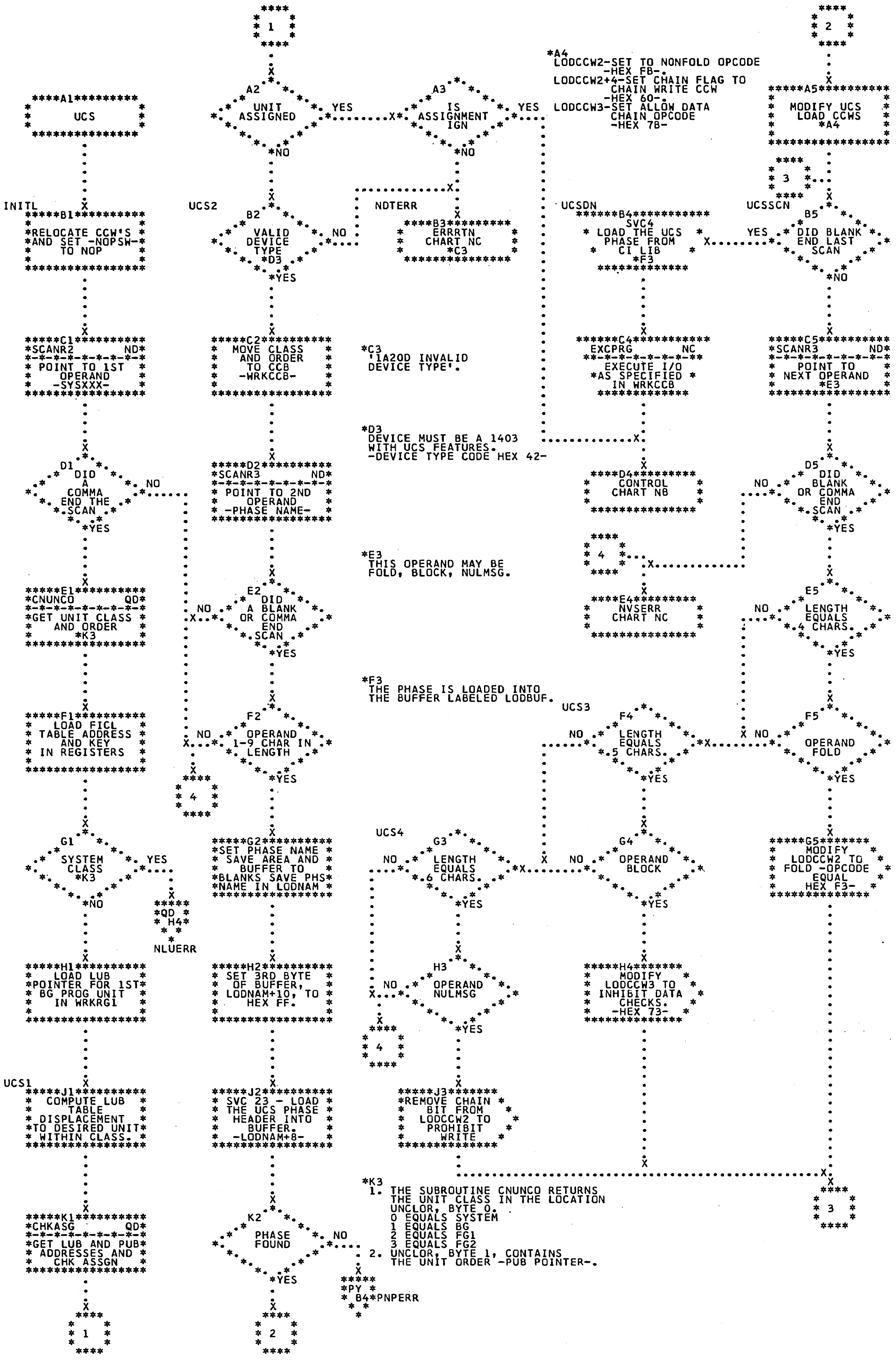
*F2
THE REGISTER SAVE AREA CAN BE CONSIDERED AS BEING IN TWO PARTS. PART 1 CONTAINS REGS 9-0. PART 2 CONTAINS REGS 1-8. THE SAVE OPERATION PUTS REG VALUES IN THE CORRECT SAVE AREA.

*G5
POINT1 CONTAINS THE MOVE FROM ADDRESS. POINT2 CONTAINS THE MOVE TO ADDRESS.

*H2
THE PSW IN THE FG SAVE AREA IS ASSEMBLED WITH THE SYSTEM MASK, KEY, AMWP, AND INTERRUPTION CODE FIELDS CONTAINING VALUES. THIS LOGICAL TRANSIENT PUTS THE ADDRESS OF A SVC 4 INTO THE INSTRUCTION ADDRESS FIELD. WHEN A SVC 11 IS ISSUED THE LOGICAL TRANSIENT RELEASES ALL CONTROL. THE FOREGROUND PIB IS ATTACHED, INDICATING AVAILABILITY FOR TASK SELECTION. WHEN A FOREGROUND TASK IS SELECTED, THE PSW IN THE FOREGROUND SAVE AREA BECOMES THE CURRENT PSW. THE SEQUENCE OF EVENTS THAT FOLLOWS IS

1. EXECUTE THE SVC 4 TO LOAD THE FOREGROUND PROGRAM, NAMED IN REGISTER 1, INTO THE FOREGROUND AREA, STARTING AT THE ADDRESS SPECIFIED IN REGISTER 0.
2. EXECUTE A BRANCH TO THE FOREGROUND ENTRY ADDRESS AS SUPPLIED IN REGISTER ONE BY THE SVC 4.
3. EXECUTE THE FOREGROUND PROGRAM.

Chart QC. $$BATTNM - UCS Statement Processor
Refer to Chart 14.


1
A1 UCS
INITL B1 RELOCATE CCW'S AND SET -NOPSW- TO NOP
C1 SCANR2 ND POINT TO 1ST OPERAND -SYSXXX-
D1 DID A COMMA END THE SCAN
NO
YES
E1 CNUNCO QD GET UNIT CLASS AND ORDER *K3
F1 LOAD FICL TABLE ADDRESS AND KEY IN REGISTERS
G1 SYSTEM CLASS *K3
YES
NO
QD H4
NLUERR
H1 LOAD LUB POINTER FOR 1ST BG PROG UNIT IN WRKRG1
UCS1 J1 COMPUTE LUB TABLE DISPLACEMENT TO DESIRED UNIT WITHIN CLASS.
K1 CHKASG QD GET LUB AND PUB ADDRESSES AND CHK ASSGN
1
A2 UNIT ASSIGNED
YES
NO
UCS2 B2 VALID DEVICE TYPE *D3
NO
YES
C2 MOVE CLASS AND ORDER TO CCB -WRKCCB-
D2 SCANR3 ND POINT TO 2ND OPERAND -PHASE NAME-
E2 DID A BLANK OR COMMA END SCAN
NO
YES
F2 OPERAND 1-9 CHAR IN LENGTH
NO
YES
4
G2 SET PHASE NAME SAVE AREA AND BUFFER TO BLANKS SAVE PHS NAME IN LODNAM
H2 SET 3RD BYTE OF BUFFER, LODNAM+10, TO HEX FF.
J2 SVC 23 - LOAD THE UCS PHASE HEADER INTO BUFFER. -LODNAM+8-
K2 PHASE FOUND
NO
YES
PY B4 PNPERR
2
A3 IS ASSIGNMENT IGN
YES
NO
NOTERR B3 ERRRTN CHART NC *C3
UCS4 G3 LENGTH EQUALS 6 CHARS.
NO
YES
H3 OPERAND NULMSG
NO
YES
4
J3 REMOVE CHAIN BIT FROM LODCCW2 TO PROHIBIT WRITE
UCSDN B4 SVC4 LOAD THE UCS PHASE FROM CI LIB *F3
C4 EXCPRG NC EXECUTE I/O AS SPECIFIED IN WRKCCB
D4 CONTROL CHART NB
4
E4 NVSERR CHART NC
UCS3 F4 LENGTH EQUALS 5 CHARS.
NO
YES
G4 OPERAND BLOCK
NO
YES
H4 MODIFY LODCCW3 TO INHIBIT DATA CHECKS. -HEX 73-
2
A5 MODIFY UCS LOAD CCWS *A4
3
UCSSCN B5 DID BLANK END LAST SCAN
YES
NO
C5 SCANR3 ND POINT TO NEXT OPERAND *E3
D5 DID BLANK OR COMMA END SCAN
NO
YES
E5 LENGTH EQUALS 4 CHARS.
NO
YES
F5 OPERAND FOLD
NO
YES
G5 MODIFY LODCCW2 TO FOLD -OPCODE EQUAL HEX F3-
3

*A4
LODCCW2-SET TO NONFOLD OPCODE -HEX FB-.
LODCCW2+4-SET CHAIN FLAG TO CHAIN WRITE CCW -HEX 60-.
LODCCW3-SET ALLOW DATA CHAIN OPCODE -HEX 7B-

*C3
'1A20D INVALID DEVICE TYPE'.

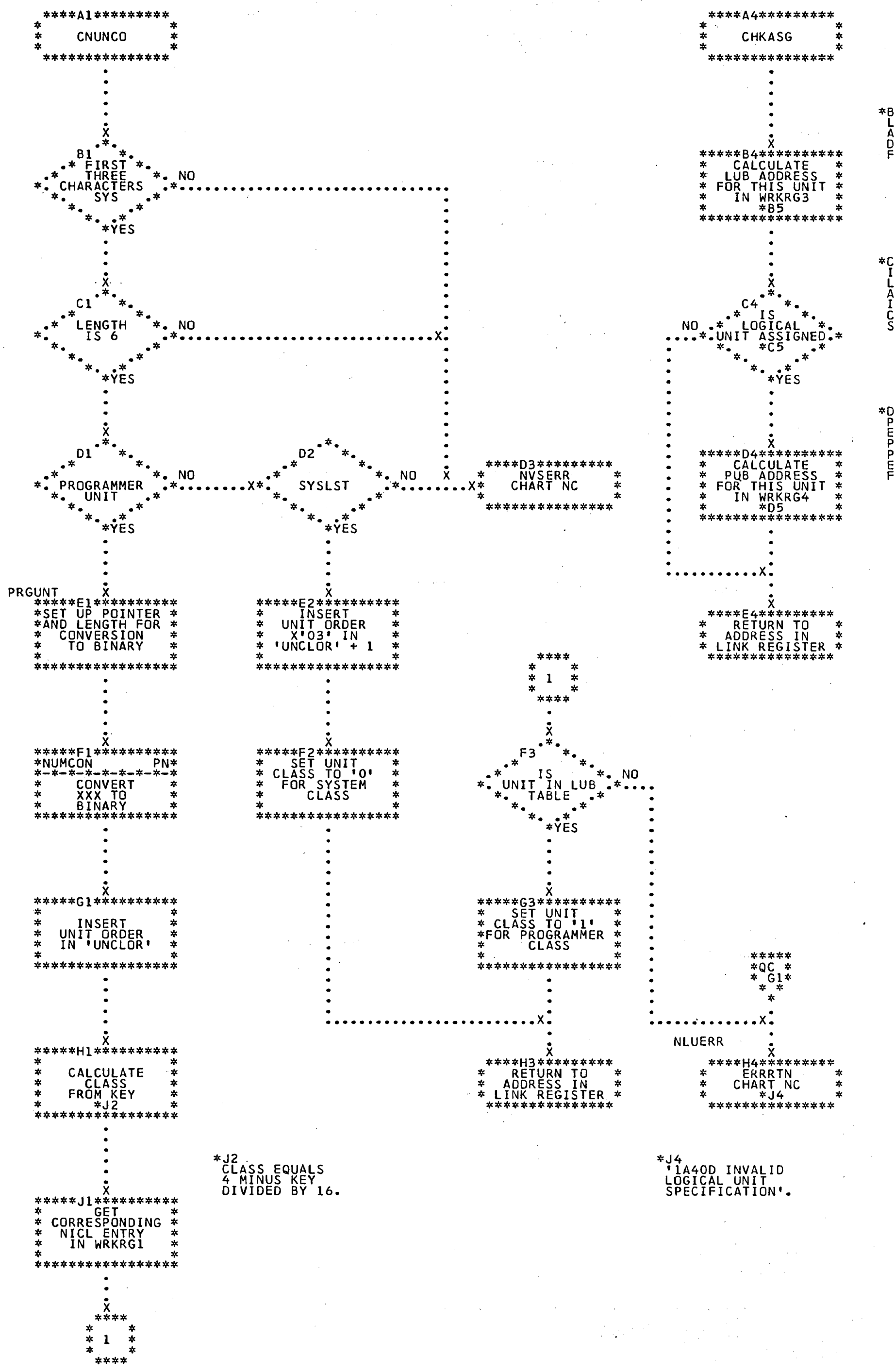
*D3
DEVICE MUST BE A 1403 WITH UCS FEATURES. -DEVICE TYPE CODE HEX 42-

*E3
THIS OPERAND MAY BE FOLD, BLOCK, NULMSG.

*F3
THE PHASE IS LOADED INTO THE BUFFER LABELED LODBUF.

*K3
1. THE SUBROUTINE CNUNCO RETURNS THE UNIT CLASS IN THE LOCATION UNCLOR, BYTE 0.
0 EQUALS SYSTEM
1 EQUALS BG
2 EQUALS FG1
3 EQUALS FG2
2. UNCLOR, BYTE 1, CONTAINS THE UNIT ORDER -PUB POINTER-.

Chart QD. $$BATTNM - UCS Subroutines
Refer to Chart 14.


A1
CNUNCO
B1
FIRST THREE CHARACTERS SYS
NO
YES
C1
LENGTH IS 6
NO
YES
D1
PROGRAMMER UNIT
NO
YES
D2
SYSLST
NO
YES
D3
NVSERR
CHART NC
PRGUNT
E1
SET UP POINTER AND LENGTH FOR CONVERSION TO BINARY
E2
INSERT UNIT ORDER X'03' IN 'UNCLOR' + 1
F1
NUMCON PN
CONVERT XXX TO BINARY
F2
SET UNIT CLASS TO '0' FOR SYSTEM CLASS
G1
INSERT UNIT ORDER IN 'UNCLOR'
H1
CALCULATE CLASS FROM KEY
*J2
J1
GET CORRESPONDING NICL ENTRY IN WRKRG1
1
1
F3
IS UNIT IN LUB TABLE
NO
YES
G3
SET UNIT CLASS TO '1' FOR PROGRAMMER CLASS
QC
G1
H3
RETURN TO ADDRESS IN LINK REGISTER
NLUERR
H4
ERRRTN
CHART NC
*J4
A4
CHKASG
B4
CALCULATE LUB ADDRESS FOR THIS UNIT IN WRKRG3
*B5
C4
IS LOGICAL UNIT ASSIGNED
*C5
NO
YES
D4
CALCULATE PUB ADDRESS FOR THIS UNIT IN WRKRG4
*D5
E4
RETURN TO ADDRESS IN LINK REGISTER

*B5
LUB TABLE ADDRESS PLUS DISPLACEMENT FOR THIS UNIT.

*C5
IN BYTE 0 OF THE LUB, X'FE'=IGNORE AND X'FF'=UNASSIGNED. IN EITHER CASE, THE CONDITION CODE IS SET TO 1.

*D5
PUB POINTER X 8 EQUALS DISPLACEMENT. PUB TABLE ADDRESS PLUS DISPLACEMENT EQUALS PUB ADDRESS FOR THIS UNIT.

*J2
CLASS EQUALS 4 MINUS KEY DIVIDED BY 16.

*J4
'1A40D INVALID LOGICAL UNIT SPECIFICATION'.

Chart QE. $$BATTNN - TIMER Statement Processor
Refer to Chart 12.

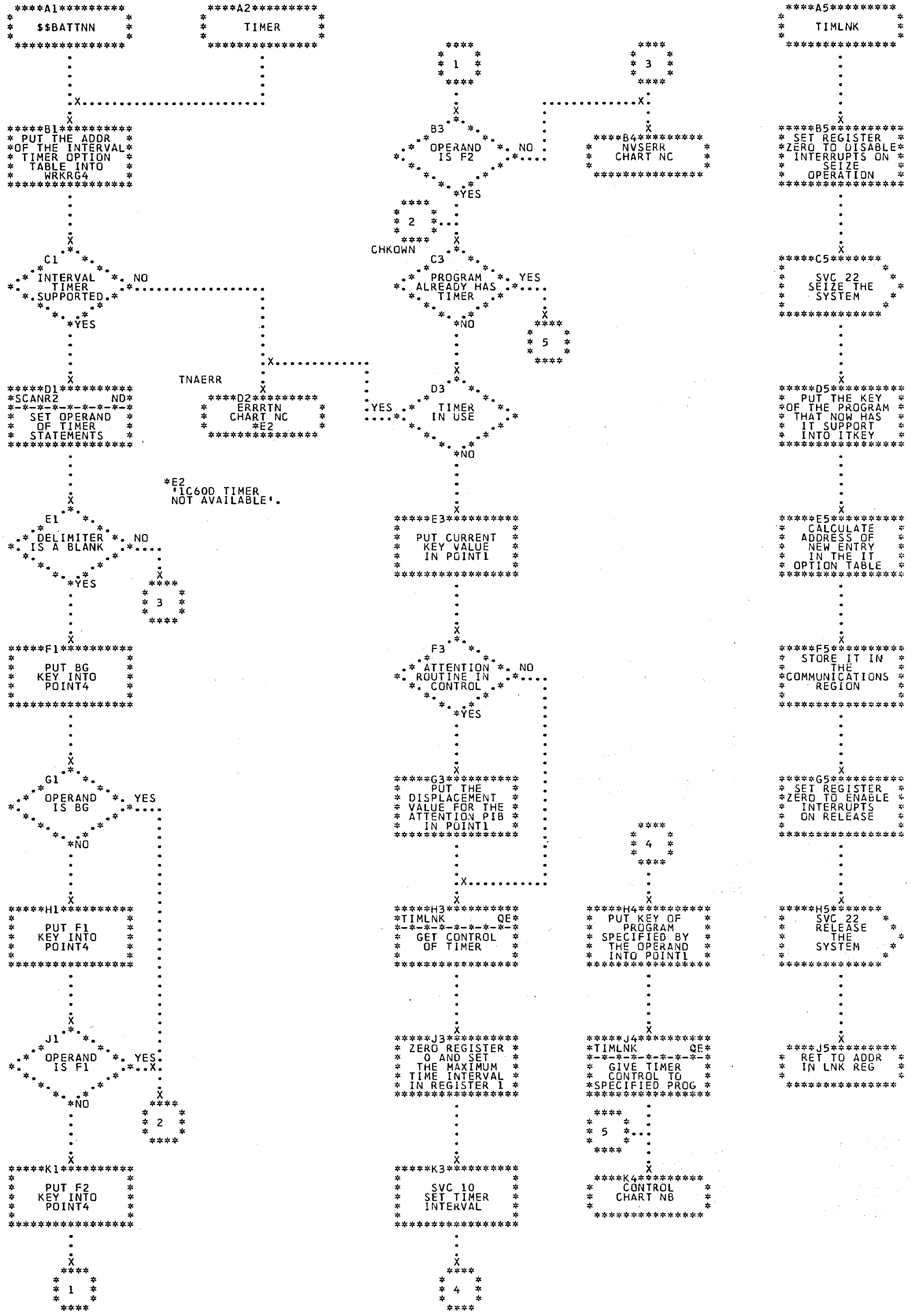
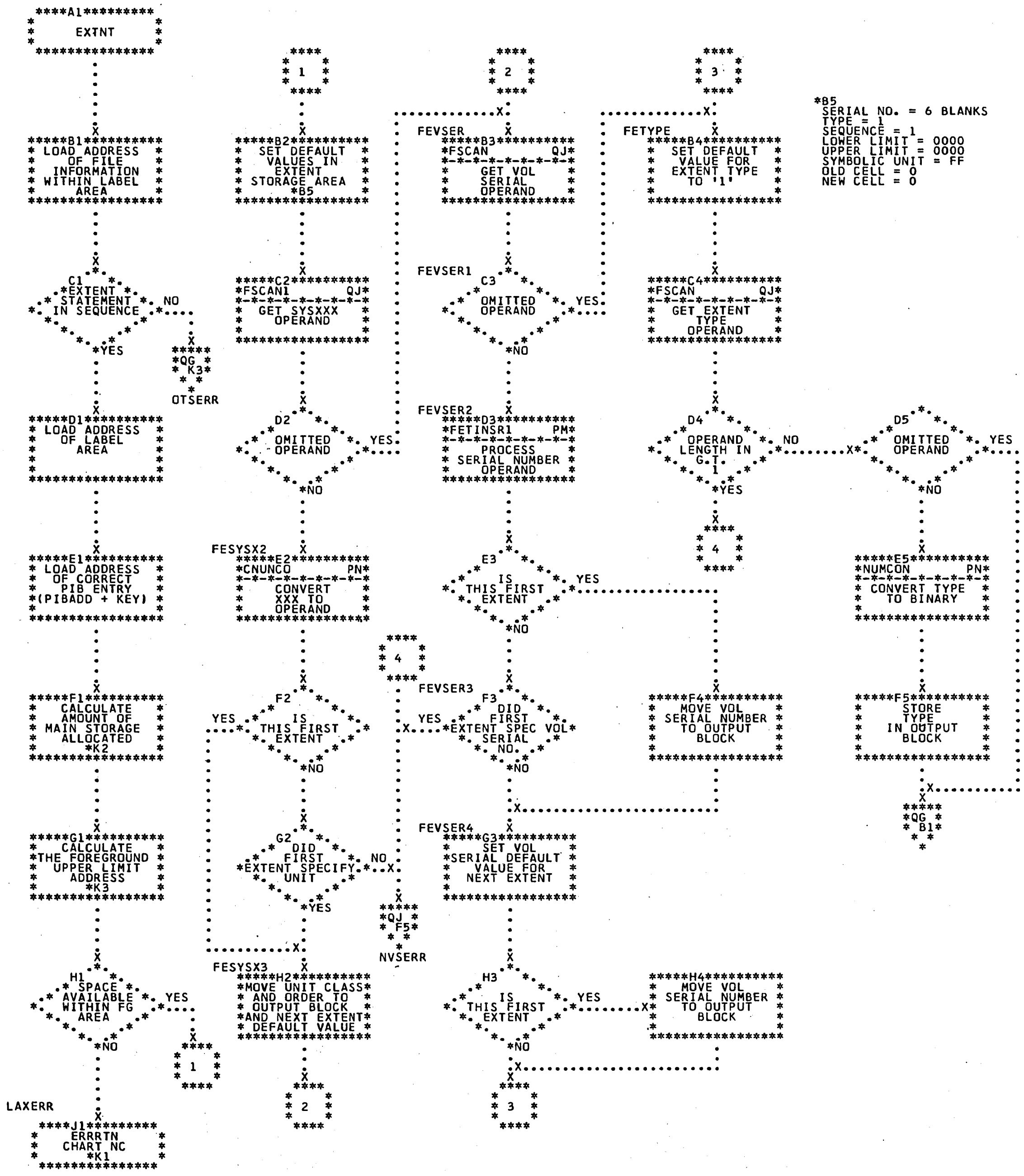

A1 $$BATTNN
A2 TIMER
A5 TIMLNK
1
3
B1 PUT THE ADDR OF THE INTERVAL TIMER OPTION TABLE INTO WRKRG4
B3 OPERAND IS F2
NO
YES
B4 NVSERR CHART NC
B5 SET REGISTER ZERO TO DISABLE INTERRUPTS ON SEIZE OPERATION
2
CHKOWN
C1 INTERVAL TIMER SUPPORTED
NO
YES
C3 PROGRAM ALREADY HAS TIMER
YES
NO
5
C5 SVC 22 SEIZE THE SYSTEM
TNAERR
D1 SCANR2 ND SET OPERAND OF TIMER STATEMENTS
D2 ERRRTN CHART NC *E2
D3 TIMER IN USE
YES
NO
D5 PUT THE KEY OF THE PROGRAM THAT NOW HAS IT SUPPORT INTO ITKEY
*E2 '1C60D TIMER NOT AVAILABLE'.
E1 DELIMITER IS A BLANK
NO
YES
3
E3 PUT CURRENT KEY VALUE IN POINT1
E5 CALCULATE ADDRESS OF NEW ENTRY IN THE IT OPTION TABLE
F1 PUT BG KEY INTO POINT4
F3 ATTENTION ROUTINE IN CONTROL
NO
YES
F5 STORE IT IN THE COMMUNICATIONS REGION
G1 OPERAND IS BG
YES
NO
G3 PUT THE DISPLACEMENT VALUE FOR THE ATTENTION PIB IN POINT1
4
G5 SET REGISTER ZERO TO ENABLE INTERRUPTS ON RELEASE
H1 PUT F1 KEY INTO POINT4
H3 TIMLNK QE GET CONTROL OF TIMER
H4 PUT KEY OF PROGRAM SPECIFIED BY THE OPERAND INTO POINT1
H5 SVC 22 RELEASE THE SYSTEM
J1 OPERAND IS F1
YES
NO
2
J3 ZERO REGISTER 0 AND SET THE MAXIMUM TIME INTERVAL IN REGISTER 1
J4 TIMLNK QE GIVE TIMER CONTROL TO SPECIFIED PROG
J5 RET TO ADDR IN LNK REG
5
K1 PUT F2 KEY INTO POINT4
K3 SVC 10 SET TIMER INTERVAL
K4 CONTROL CHART NB
1
4

Chart QF. $$BATTNO - EXTENT Statement Processor (Part 1 of 3)
Refer to Chart 14.


A1
EXTNT
1
2
3
B1
LOAD ADDRESS OF FILE INFORMATION WITHIN LABEL AREA
B2
SET DEFAULT VALUES IN EXTENT STORAGE AREA
*B5
FEVSER
B3
FSCAN QJ
GET VOL SERIAL OPERAND
FETYPE
B4
SET DEFAULT VALUE FOR EXTENT TYPE TO '1'
C1
EXTENT STATEMENT IN SEQUENCE
NO
YES
QG K3
OTSERR
C2
FSCAN1 QJ
GET SYSXXX OPERAND
FEVSER1
C3
OMITTED OPERAND
YES
NO
C4
FSCAN QJ
GET EXTENT TYPE OPERAND
D1
LOAD ADDRESS OF LABEL AREA
D2
OMITTED OPERAND
YES
NO
FEVSER2
D3
FETINSR1 PM
PROCESS SERIAL NUMBER OPERAND
D4
OPERAND LENGTH IN G.T. 1
NO
YES
4
D5
OMITTED OPERAND
YES
NO
E1
LOAD ADDRESS OF CORRECT PIB ENTRY (PIBADD + KEY)
FESYSX2
E2
CNUNCO PN
CONVERT XXX TO OPERAND
E3
IS THIS FIRST EXTENT
YES
NO
E5
NUMCON PN
CONVERT TYPE TO BINARY
4
F1
CALCULATE AMOUNT OF MAIN STORAGE ALLOCATED
*K2
F2
IS THIS FIRST EXTENT
YES
NO
FEVSER3
F3
DID FIRST EXTENT SPEC VOL SERIAL NO.
YES
NO
F4
MOVE VOL SERIAL NUMBER TO OUTPUT BLOCK
F5
STORE TYPE IN OUTPUT BLOCK
QG B1
G1
CALCULATE THE FOREGROUND UPPER LIMIT ADDRESS
*K3
G2
DID FIRST EXTENT SPECIFY UNIT
NO
YES
QJ F5
NVSERR
FEVSER4
G3
SET VOL SERIAL DEFAULT VALUE FOR NEXT EXTENT
H1
SPACE AVAILABLE WITHIN FG AREA
YES
NO
1
FESYSX3
H2
MOVE UNIT CLASS AND ORDER TO OUTPUT BLOCK AND NEXT EXTENT DEFAULT VALUE
2
H3
IS THIS FIRST EXTENT
YES
NO
H4
MOVE VOL SERIAL NUMBER TO OUTPUT BLOCK
3
LAXERR
J1
ERRRTN
CHART NC
*K1

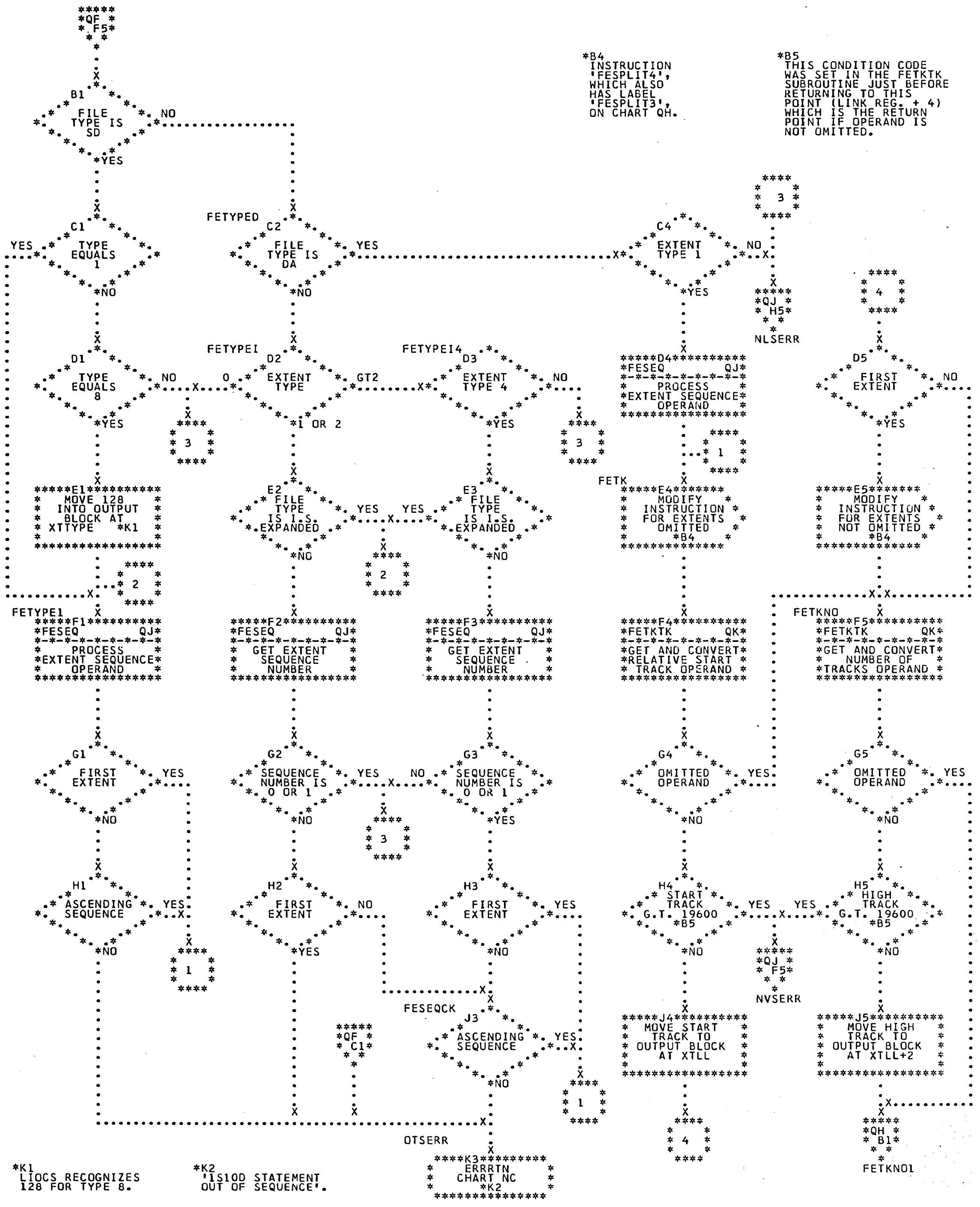
*B5
SERIAL NO. = 6 BLANKS
TYPE = 1
SEQUENCE = 1
LOWER LIMIT = 0000
UPPER LIMIT = 0000
SYMBOLIC UNIT = FF
OLD CELL = 0
NEW CELL = 0

*K1
'IL10D LABEL
AREA EXHAUSTED'.

*K2
EXTRACT THE NUMBER OF 2K BLOCKS ALLOCATED FROM THE PIB AND MULTIPLY BY 2048 TO GET NUMBER OF BYTES ALLOCATED.

*K3
EXTRACT THE ADDRESS OF THE PROGRAM ORIGIN FROM THE PIB. ADD NUMBER OF BYTES ALLOCATED TO THIS PROGRAM.

Chart QG. $$BATTNO - EXTENT Statement Processor (Part 2 of 3)
Refer to Chart 14.


QF F5
B1 FILE TYPE IS SD
NO
YES
FETYPED
C1 TYPE EQUALS 1
YES
NO
C2 FILE TYPE IS DA
YES
NO
C4 EXTENT TYPE 1
NO
YES
3
QJ H5
NLSERR
4
FETYPEI
D1 TYPE EQUALS 8
NO
YES
3
0
D2 EXTENT TYPE
GT2
1 OR 2
FETYPEI4
D3 EXTENT TYPE 4
NO
YES
3
D4 FESEQ QJ PROCESS EXTENT SEQUENCE OPERAND
1
D5 FIRST EXTENT
NO
YES
E1 MOVE 128 INTO OUTPUT BLOCK AT XTTYPE *K1
2
E2 FILE TYPE IS I.S. EXPANDED
YES
NO
2
E3 FILE TYPE IS I.S. EXPANDED
YES
NO
FETK
E4 MODIFY INSTRUCTION FOR EXTENTS OMITTED *B4
E5 MODIFY INSTRUCTION FOR EXTENTS NOT OMITTED *B4
FETYPE1
F1 FESEQ QJ PROCESS EXTENT SEQUENCE OPERAND
F2 FESEQ QJ GET EXTENT SEQUENCE NUMBER
F3 FESEQ QJ GET EXTENT SEQUENCE NUMBER
F4 FETKTK QK GET AND CONVERT RELATIVE START TRACK OPERAND
FETKNO
F5 FETKTK QK GET AND CONVERT NUMBER OF TRACKS OPERAND
G1 FIRST EXTENT
YES
NO
G2 SEQUENCE NUMBER IS 0 OR 1
YES
NO
3
G3 SEQUENCE NUMBER IS 0 OR 1
NO
YES
G4 OMITTED OPERAND
YES
NO
G5 OMITTED OPERAND
YES
NO
H1 ASCENDING SEQUENCE
YES
NO
1
H2 FIRST EXTENT
NO
YES
H3 FIRST EXTENT
YES
NO
H4 START TRACK G.T. 19600 *B5
YES
NO
QJ F5
NVSERR
H5 HIGH TRACK G.T. 19600 *B5
YES
NO
FESEQCK
J3 ASCENDING SEQUENCE
YES
NO
1
QF C1
J4 MOVE START TRACK TO OUTPUT BLOCK AT XTLL
4
J5 MOVE HIGH TRACK TO OUTPUT BLOCK AT XTLL+2
QH B1
FETKNO1
OTSERR
K3 ERRRTN CHART NC *K2

*B4
INSTRUCTION 'FESPLIT4', WHICH ALSO HAS LABEL 'FESPLIT3', ON CHART QH.

*B5
THIS CONDITION CODE WAS SET IN THE FETKTK SUBROUTINE JUST BEFORE RETURNING TO THIS POINT (LINK REG. + 4) WHICH IS THE RETURN POINT IF OPERAND IS NOT OMITTED.

*K1
LIOCS RECOGNIZES 128 FOR TYPE 8.

*K2
'1S1OD STATEMENT OUT OF SEQUENCE'.

Chart QH. $$BATTNO - EXTENT Statement Processor (Part 3 of 3)
Refer to Chart 14.

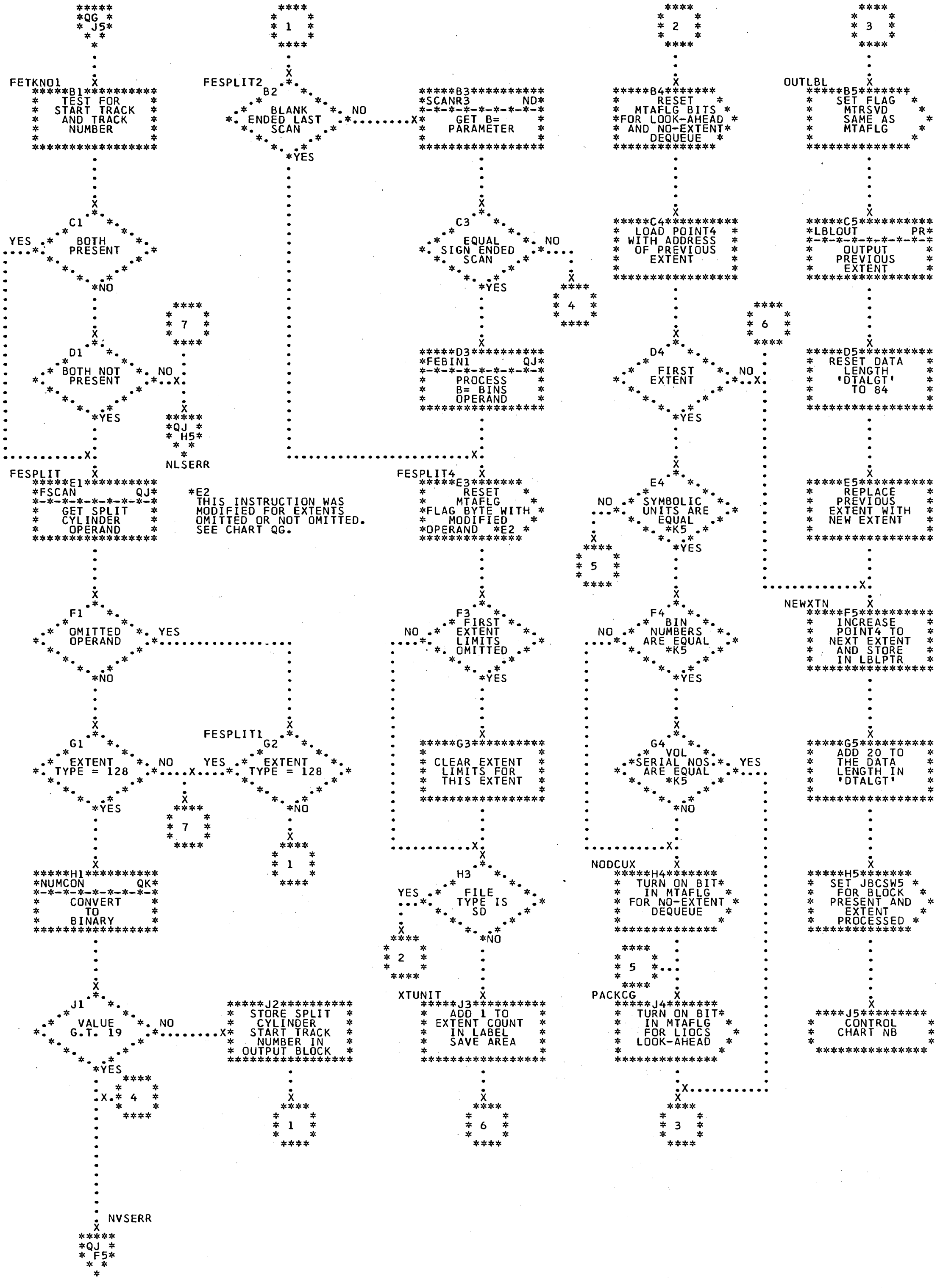

QG J5
FETKNO1
B1 TEST FOR START TRACK AND TRACK NUMBER
C1 BOTH PRESENT
YES
NO
D1 BOTH NOT PRESENT
NO
YES
7
QJ H5
NLSERR
FESPLIT
E1 FSCAN QJ GET SPLIT CYLINDER OPERAND
*E2 THIS INSTRUCTION WAS MODIFIED FOR EXTENTS OMITTED OR NOT OMITTED. SEE CHART QG.
F1 OMITTED OPERAND
YES
NO
G1 EXTENT TYPE = 128
NO
YES
7
H1 NUMCON QK CONVERT TO BINARY
J1 VALUE G.T. 19
NO
YES
4
NVSERR
QJ F5
1
FESPLIT2
B2 BLANK ENDED LAST SCAN
NO
YES
FESPLIT1
G2 EXTENT TYPE = 128
YES
NO
1
J2 STORE SPLIT CYLINDER START TRACK NUMBER IN OUTPUT BLOCK
1
B3 SCANR3 ND GET B= PARAMETER
C3 EQUAL SIGN ENDED SCAN
NO
4
YES
D3 FEBIN1 QJ PROCESS B= BINS OPERAND
FESPLIT4
E3 RESET MTAFLG FLAG BYTE WITH MODIFIED OPERAND *E2
F3 FIRST EXTENT LIMITS OMITTED
NO
YES
G3 CLEAR EXTENT LIMITS FOR THIS EXTENT
H3 FILE TYPE IS SD
YES
2
NO
XTUNIT
J3 ADD 1 TO EXTENT COUNT IN LABEL SAVE AREA
6
2
B4 RESET MTAFLG BITS FOR LOOK-AHEAD AND NO-EXTENT DEQUEUE
C4 LOAD POINT4 WITH ADDRESS OF PREVIOUS EXTENT
D4 FIRST EXTENT
NO
YES
6
E4 SYMBOLIC UNITS ARE EQUAL K5
NO
5
YES
F4 BIN NUMBERS ARE EQUAL K5
NO
YES
G4 VOL SERIAL NOS. ARE EQUAL K5
YES
NO
NODCUX
H4 TURN ON BIT IN MTAFLG FOR NO-EXTENT DEQUEUE
5
PACKCG
J4 TURN ON BIT IN MTAFLG FOR LIOCS LOOK-AHEAD
3
3
OUTLBL
B5 SET FLAG MTRSVD SAME AS MTAFLG
C5 LBLOUT PR OUTPUT PREVIOUS EXTENT
D5 RESET DATA LENGTH 'DTALGT' TO 84
E5 REPLACE PREVIOUS EXTENT WITH NEW EXTENT
NEWXTN
F5 INCREASE POINT4 TO NEXT EXTENT AND STORE IN LBLPTR
G5 ADD 20 TO THE DATA LENGTH IN 'DTALGT'
H5 SET JBCSW5 FOR BLOCK PRESENT AND EXTENT PROCESSED
J5 CONTROL CHART NB

Chart QJ. $$BATTNO - EXTENT Processor Subroutines
Refer to Chart 14.

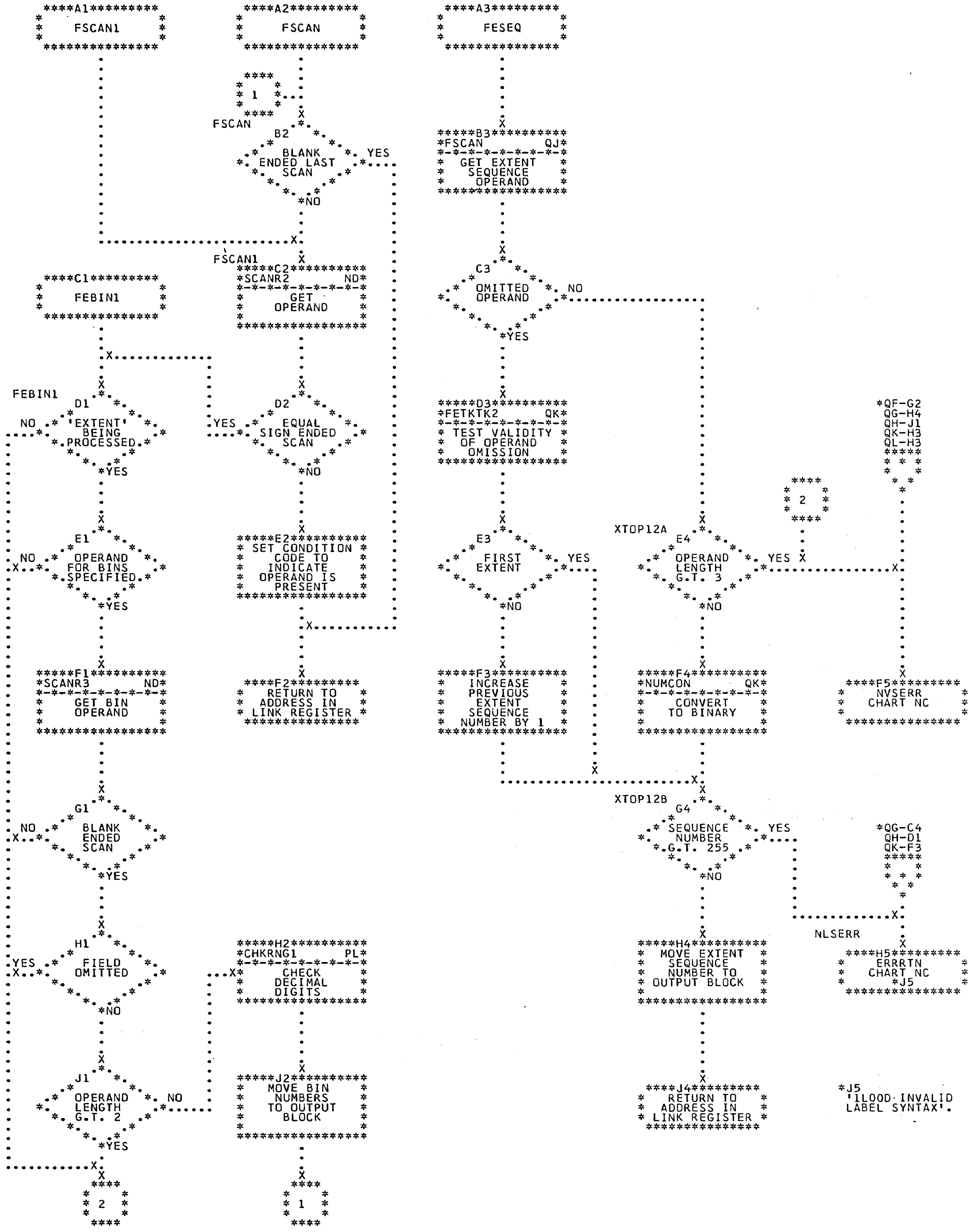

A1 FSCAN1
A2 FSCAN
A3 FESEQ
1
FSCAN B2 BLANK ENDED LAST SCAN
YES
NO
B3 FSCAN QJ GET EXTENT SEQUENCE OPERAND
FSCAN1 C2 SCANR2 ND GET OPERAND
C1 FEBIN1
C3 OMITTED OPERAND
NO
YES
FEBIN1 D1 'EXTENT' BEING PROCESSED
NO
YES
D2 EQUAL SIGN ENDED SCAN
YES
NO
D3 FETKTK2 QK TEST VALIDITY OF OPERAND OMISSION
*QF-G2 QG-H4 QH-J1 QK-H3 QL-H3
2
E1 OPERAND FOR BINS SPECIFIED
NO
YES
E2 SET CONDITION CODE TO INDICATE OPERAND IS PRESENT
E3 FIRST EXTENT
YES
NO
XTOP12A E4 OPERAND LENGTH G.T. 3
YES
NO
F1 SCANR3 ND GET BIN OPERAND
F2 RETURN TO ADDRESS IN LINK REGISTER
F3 INCREASE PREVIOUS EXTENT SEQUENCE NUMBER BY 1
F4 NUMCON QK CONVERT TO BINARY
F5 NVSERR CHART NC
G1 BLANK ENDED SCAN
NO
YES
XTOP12B G4 SEQUENCE NUMBER G.T. 255
YES
NO
*QG-C4 QH-D1 QK-F3
NLSERR
H1 FIELD OMITTED
YES
NO
H2 CHKRNG1 PL CHECK DECIMAL DIGITS
H4 MOVE EXTENT SEQUENCE NUMBER TO OUTPUT BLOCK
H5 ERRRTN CHART NC *J5
J1 OPERAND LENGTH G.T. 2
NO
YES
J2 MOVE BIN NUMBERS TO OUTPUT BLOCK
J4 RETURN TO ADDRESS IN LINK REGISTER
*J5 '1LOOD INVALID LABEL SYNTAX'.
2
1

Chart QK. $$BATTNO - Process Track Operands
Refer to Chart 14.

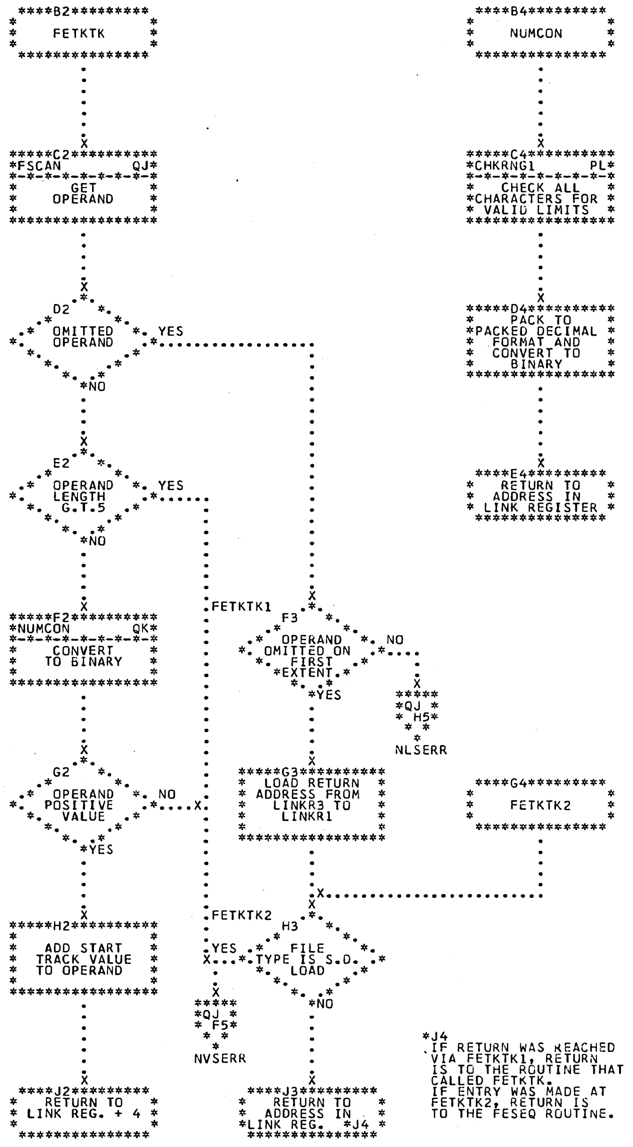

B2
FETKTK
C2
FSCAN QJ
GET OPERAND
D2
OMITTED OPERAND
YES
NO
E2
OPERAND LENGTH G.T.5
YES
NO
F2
NUMCON QK
CONVERT TO BINARY
G2
OPERAND POSITIVE VALUE
NO
YES
H2
ADD START TRACK VALUE TO OPERAND
J2
RETURN TO LINK REG. + 4
FETKTK1
F3
OPERAND OMITTED ON FIRST EXTENT
NO
YES
QJ H5
NLSERR
G3
LOAD RETURN ADDRESS FROM LINKR3 TO LINKR1
FETKTK2
H3
FILE TYPE IS S.D. LOAD
YES
NO
QJ F5
NVSERR
J3
RETURN TO ADDRESS IN LINK REG. *J4
B4
NUMCON
C4
CHKRNG1 PL
CHECK ALL CHARACTERS FOR VALID LIMITS
D4
PACK TO PACKED DECIMAL FORMAT AND CONVERT TO BINARY
E4
RETURN TO ADDRESS IN LINK REGISTER
G4
FETKTK2
*J4
IF RETURN WAS REACHED VIA FETKTK1, RETURN IS TO THE ROUTINE THAT CALLED FETKTK.
IF ENTRY WAS MADE AT FETKTK2, RETURN IS TO THE FESEQ ROUTINE.

Chart QL. $$BATTNO - LBLTYP Statement Processor
Refer to Chart 14.

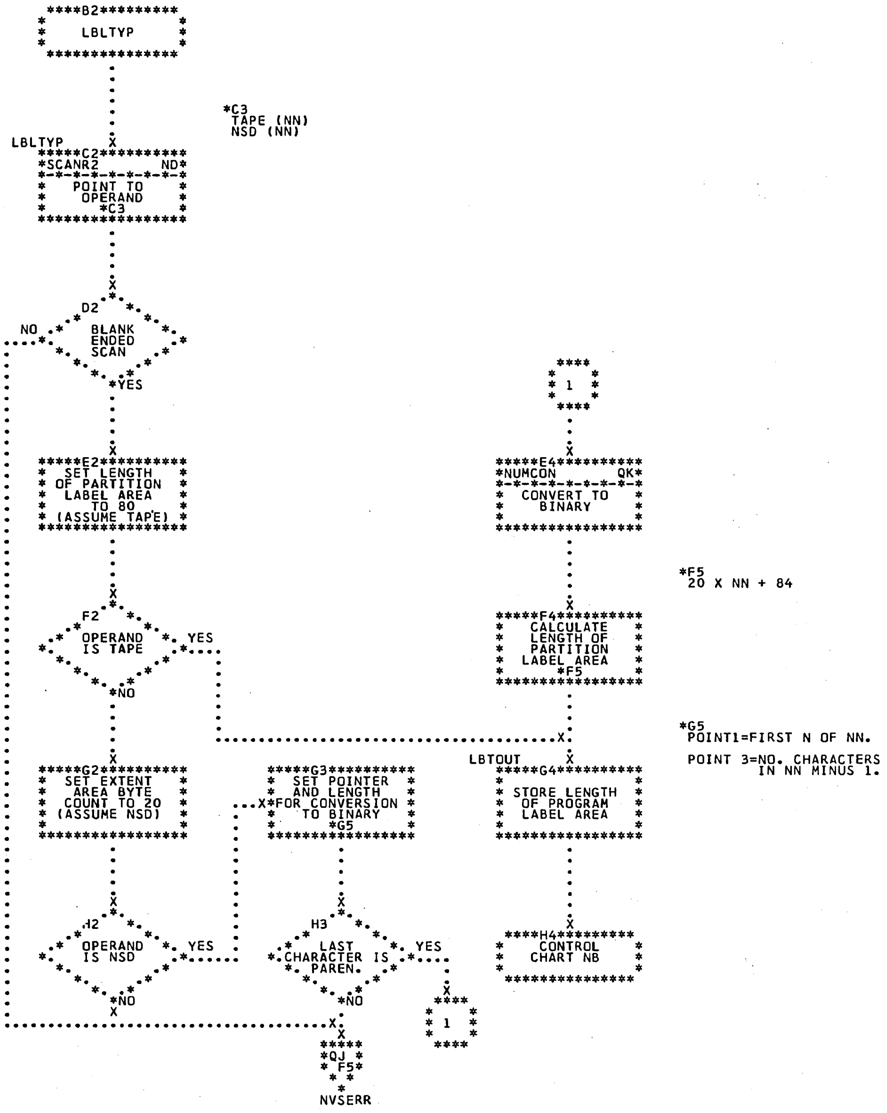
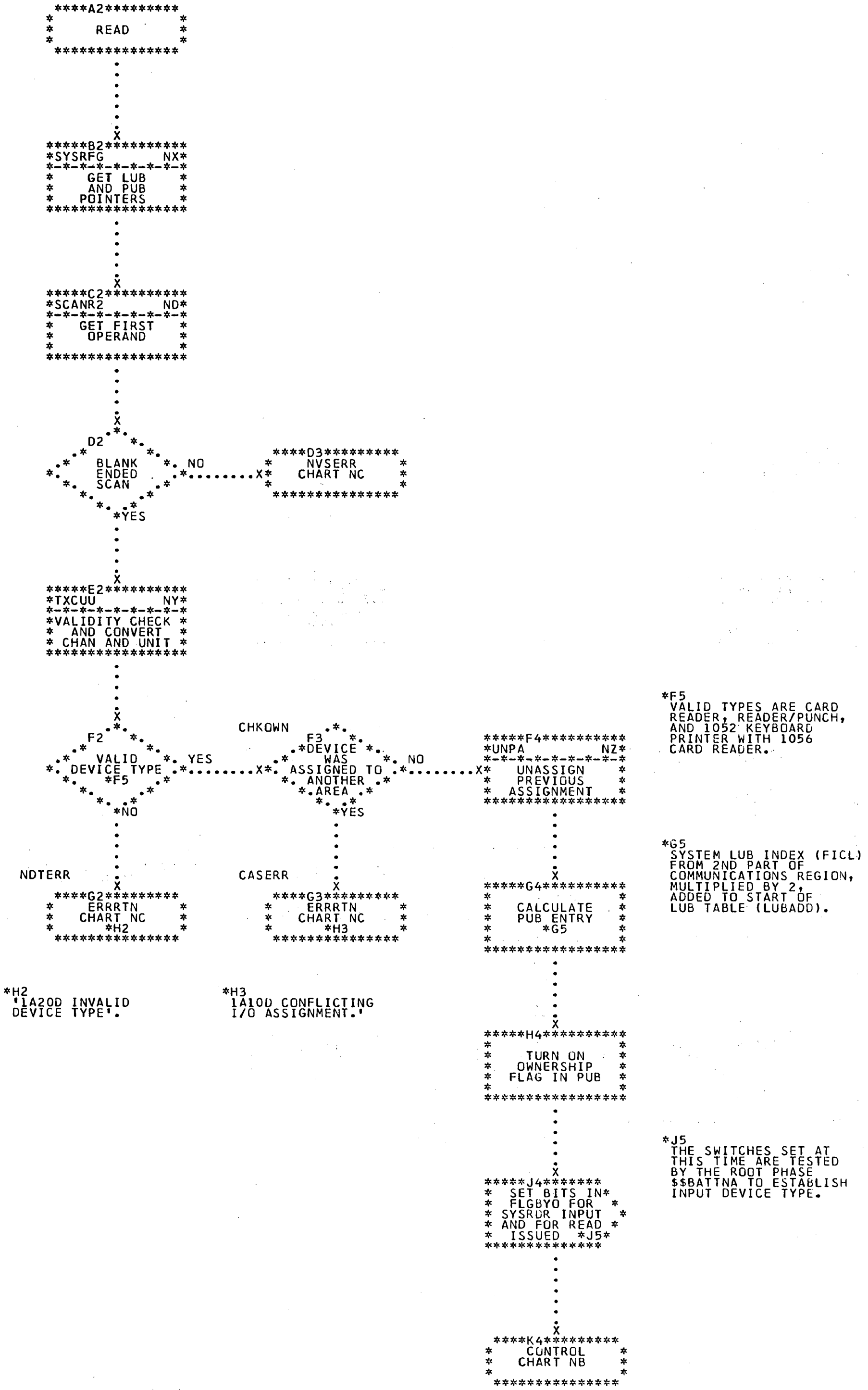


Chart QM. $$BATTNP - READ Statement Processor
Refer to Chart 14.


A2 READ
B2 SYSRFG NX GET LUB AND PUB POINTERS
C2 SCANR2 ND GET FIRST OPERAND
D2 BLANK ENDED SCAN
NO
D3 NVSERR CHART NC
YES
E2 TXCUU NY VALIDITY CHECK AND CONVERT CHAN AND UNIT
F2 VALID DEVICE TYPE *F5
YES
CHKOWN
F3 DEVICE WAS ASSIGNED TO ANOTHER AREA
NO
F4 UNPA NZ UNASSIGN PREVIOUS ASSIGNMENT
NO
NDTERR
G2 ERRRTN CHART NC *H2
YES
CASERR
G3 ERRRTN CHART NC *H3
G4 CALCULATE PUB ENTRY *G5
H4 TURN ON OWNERSHIP FLAG IN PUB
J4 SET BITS IN FLGBYO FOR SYSRDR INPUT AND FOR READ ISSUED *J5
K4 CONTROL CHART NB

*F5
VALID TYPES ARE CARD READER, READER/PUNCH, AND 1052 KEYBOARD PRINTER WITH 1056 CARD READER.

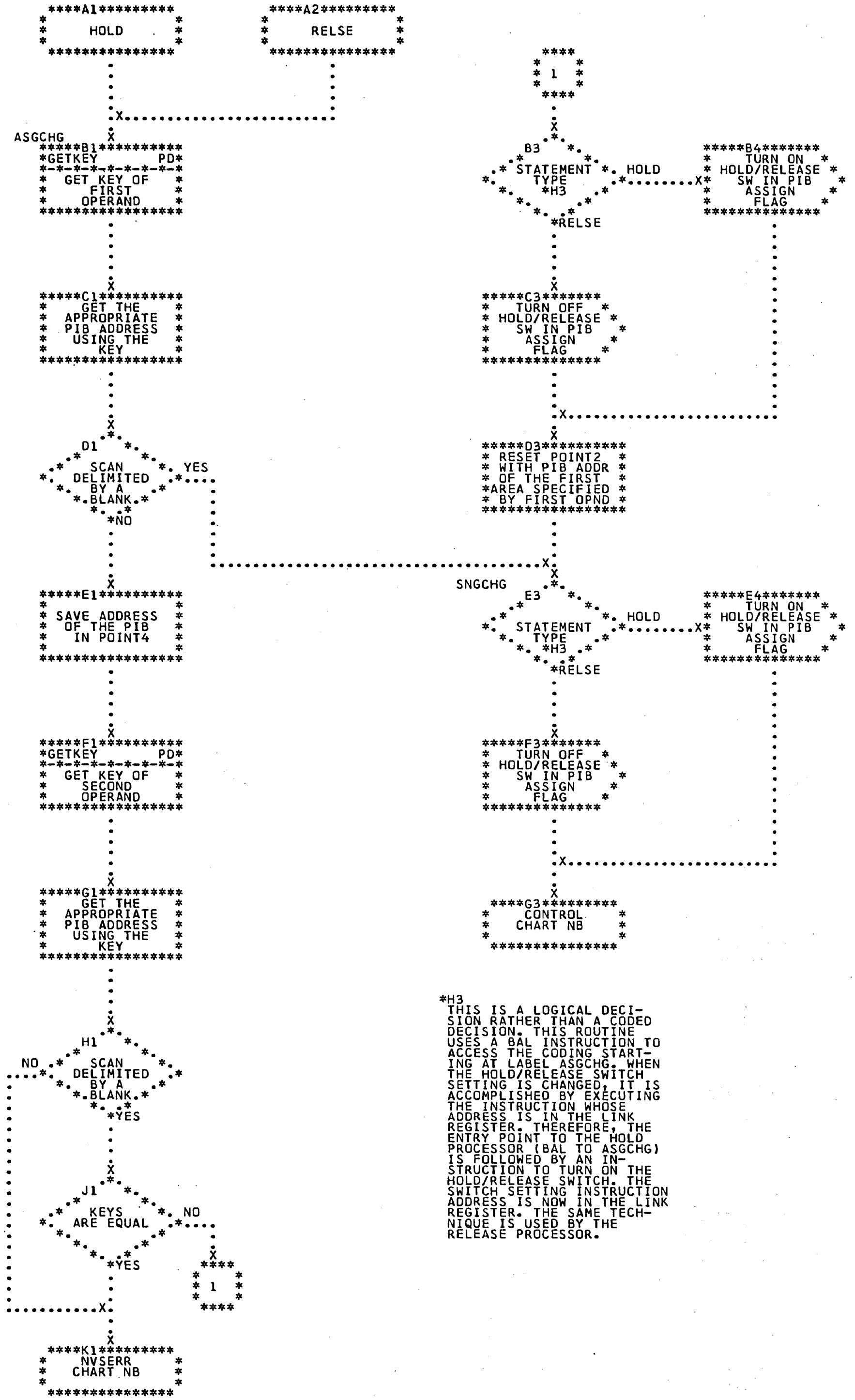
*G5
SYSTEM LUB INDEX (FICL) FROM 2ND PART OF COMMUNICATIONS REGION, MULTIPLIED BY 2, ADDED TO START OF LUB TABLE (LUBADD).

*H2
'1A20D INVALID DEVICE TYPE'.

*H3
1A10D CONFLICTING I/O ASSIGNMENT.'

*J5
THE SWITCHES SET AT THIS TIME ARE TESTED BY THE ROOT PHASE $$BATTNA TO ESTABLISH INPUT DEVICE TYPE.

Chart QN. $$BATTNP - HOLD or RELSE Statement Processor
Refer to Chart 14.


A1
HOLD
A2
RELSE
1
ASGCHG
B1
GETKEY PD
GET KEY OF FIRST OPERAND
B3
STATEMENT TYPE *H3
HOLD
RELSE
B4
TURN ON HOLD/RELEASE SW IN PIB ASSIGN FLAG
C1
GET THE APPROPRIATE PIB ADDRESS USING THE KEY
C3
TURN OFF HOLD/RELEASE SW IN PIB ASSIGN FLAG
D1
SCAN DELIMITED BY A BLANK
YES
NO
D3
RESET POINT2 WITH PIB ADDR OF THE FIRST AREA SPECIFIED BY FIRST OPND
SNGCHG
E1
SAVE ADDRESS OF THE PIB IN POINT4
E3
STATEMENT TYPE *H3
HOLD
RELSE
E4
TURN ON HOLD/RELEASE SW IN PIB ASSIGN FLAG
F1
GETKEY PD
GET KEY OF SECOND OPERAND
F3
TURN OFF HOLD/RELEASE SW IN PIB ASSIGN FLAG
G1
GET THE APPROPRIATE PIB ADDRESS USING THE KEY
G3
CONTROL CHART NB
H1
SCAN DELIMITED BY A BLANK
NO
YES
J1
KEYS ARE EQUAL
NO
YES
1
K1
NVSERR CHART NB

*H3
THIS IS A LOGICAL DECISION RATHER THAN A CODED DECISION. THIS ROUTINE USES A BAL INSTRUCTION TO ACCESS THE CODING STARTING AT LABEL ASGCHG. WHEN THE HOLD/RELEASE SWITCH SETTING IS CHANGED, IT IS ACCOMPLISHED BY EXECUTING THE INSTRUCTION WHOSE ADDRESS IS IN THE LINK REGISTER. THEREFORE, THE ENTRY POINT TO THE HOLD PROCESSOR (BAL TO ASGCHG) IS FOLLOWED BY AN INSTRUCTION TO TURN ON THE HOLD/RELEASE SWITCH. THE SWITCH SETTING INSTRUCTION ADDRESS IS NOW IN THE LINK REGISTER. THE SAME TECHNIQUE IS USED BY THE RELEASE PROCESSOR.

Chart RA. $$BEOJ - Terminated Program I/O Handling
Refer to Chart 15.

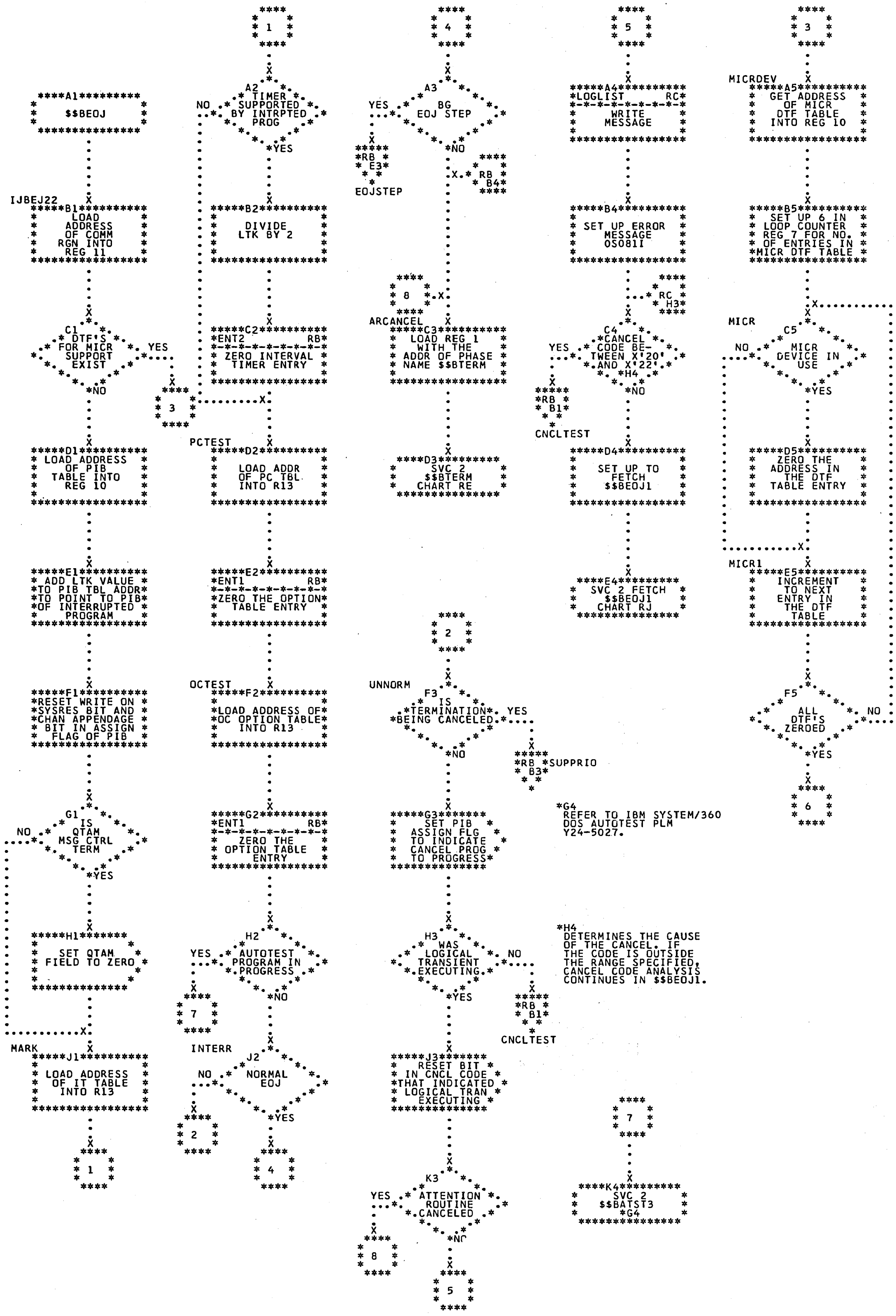

1
4
5
3
A1
$$BEOJ
IJBEJ22
B1
LOAD ADDRESS OF COMM RGN INTO REG 11
C1
DTF'S FOR MICR SUPPORT EXIST
YES
NO
3
D1
LOAD ADDRESS OF PIB TABLE INTO REG 10
E1
ADD LTK VALUE TO PIB TBL ADDR TO POINT TO PIB OF INTERRUPTED PROGRAM
F1
RESET WRITE ON SYSRES BIT AND CHAN APPENDAGE BIT IN ASSIGN FLAG OF PIB
G1
IS QTAM MSG CTRL TERM
NO
YES
H1
SET QTAM FIELD TO ZERO
MARK
J1
LOAD ADDRESS OF IT TABLE INTO R13
1
A2
TIMER SUPPORTED BY INTRPTED PROG
NO
YES
B2
DIVIDE LTK BY 2
C2
ENT2 RB
ZERO INTERVAL TIMER ENTRY
PCTEST
D2
LOAD ADDR OF PC TBL INTO R13
E2
ENT1 RB
ZERO THE OPTION TABLE ENTRY
OCTEST
F2
LOAD ADDRESS OF OC OPTION TABLE INTO R13
G2
ENT1 RB
ZERO THE OPTION TABLE ENTRY
H2
AUTOTEST PROGRAM IN PROGRESS
YES
NO
7
INTERR
J2
NORMAL EOJ
NO
YES
2
4
A3
BG EOJ STEP
YES
NO
RB E3
EOJSTEP
RB B4
8
ARCANCEL
C3
LOAD REG 1 WITH THE ADDR OF PHASE NAME $$BTERM
D3
SVC 2 $$BTERM CHART RE
2
UNNORM
F3
IS TERMINATION BEING CANCELED
YES
NO
RB B3
SUPPRIO
G3
SET PIB ASSIGN FLG TO INDICATE CANCEL PROG TO PROGRESS
H3
WAS LOGICAL TRANSIENT EXECUTING
NO
YES
RB B1
CNCLTEST
J3
RESET BIT IN CNCL CODE THAT INDICATED LOGICAL TRAN EXECUTING
K3
ATTENTION ROUTINE CANCELED
YES
NO
8
5
A4
LOGLIST RC
WRITE MESSAGE
B4
SET UP ERROR MESSAGE 0S081I
RC H3
C4
CANCEL CODE BETWEEN X'20' AND X'22'
H4
YES
NO
RB B1
CNCLTEST
D4
SET UP TO FETCH $$BEOJ1
E4
SVC 2 FETCH $$BEOJ1 CHART RJ
*G4 REFER TO IBM SYSTEM/360 DOS AUTOTEST PLM Y24-5027.
*H4 DETERMINES THE CAUSE OF THE CANCEL. IF THE CODE IS OUTSIDE THE RANGE SPECIFIED, CANCEL CODE ANALYSIS CONTINUES IN $$BEOJ1.
7
K4
SVC 2 $$BATST3 *G4
MICRDEV
A5
GET ADDRESS OF MICR DTF TABLE INTO REG 10
B5
SET UP 6 IN LOOP COUNTER REG 7 FOR NO. OF ENTRIES IN MICR DTF TABLE
MICR
C5
MICR DEVICE IN USE
NO
YES
D5
ZERO THE ADDRESS IN THE DTF TABLE ENTRY
MICR1
E5
INCREMENT TO NEXT ENTRY IN THE DTF TABLE
F5
ALL DTF'S ZEROED
NO
YES
6

Chart RB. $$BEOJ - EOJ Processing Routine and $$BEOJ Subroutines
Refer to Chart 15.

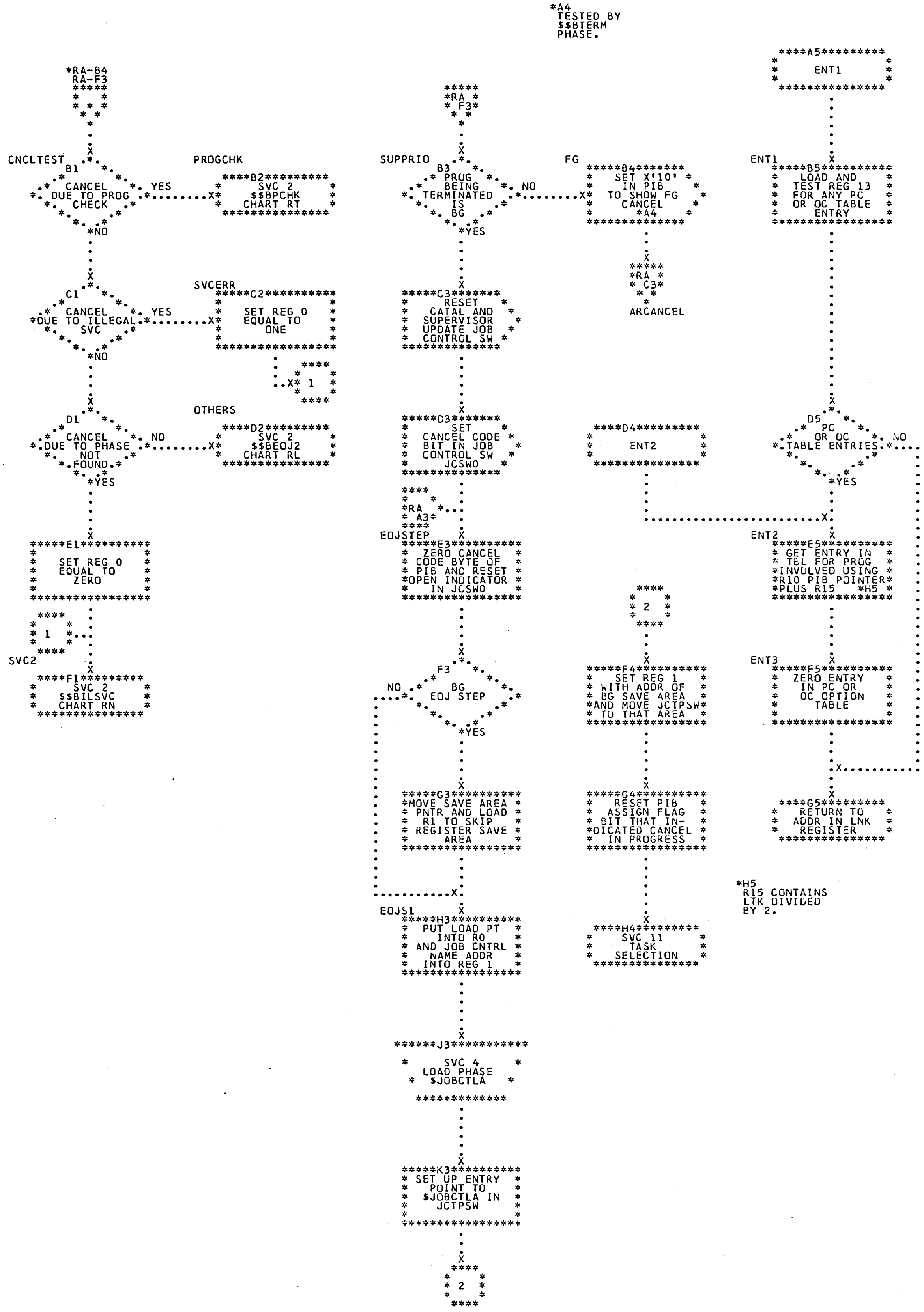

*A4 TESTED BY $$BTERM PHASE.
A5 ENT1
RA-B4 RA-F3
RA F3
CNCLTEST B1 CANCEL DUE TO PROG CHECK
YES
NO
PROGCHK B2 SVC 2 $$BPCHK CHART RT
SUPPRIO B3 PROG BEING TERMINATED IS BG
NO
YES
FG B4 SET X'10' IN PIB TO SHOW FG CANCEL *A4
ENT1 B5 LOAD AND TEST REG 13 FOR ANY PC OR OC TABLE ENTRY
C1 CANCEL DUE TO ILLEGAL SVC
SVCERR C2 SET REG 0 EQUAL TO ONE
1
C3 RESET CATAL AND SUPERVISOR UPDATE JOB CONTROL SW
RA C3 ARCANCEL
D1 CANCEL DUE TO PHASE NOT FOUND
OTHERS D2 SVC 2 $$BEOJ2 CHART RL
D3 SET CANCEL CODE BIT IN JOB CONTROL SW JCSWO
D4 ENT2
D5 PC OR OC TABLE ENTRIES
RA A3
EOJSTEP E3 ZERO CANCEL CODE BYTE OF PIB AND RESET OPEN INDICATOR IN JCSWO
E1 SET REG 0 EQUAL TO ZERO
ENT2 E5 GET ENTRY IN TEL FOR PROG INVOLVED USING R10 PIB POINTER PLUS R15 *H5
2
1
SVC2 F1 SVC 2 $$BILSVC CHART RN
F3 BG EOJ STEP
F4 SET REG 1 WITH ADDR OF BG SAVE AREA AND MOVE JCTPSW TO THAT AREA
ENT3 F5 ZERO ENTRY IN PC OR OC OPTION TABLE
G3 MOVE SAVE AREA PNTR AND LOAD R1 TO SKIP REGISTER SAVE AREA
G4 RESET PIB ASSIGN FLAG BIT THAT INDICATED CANCEL IN PROGRESS
G5 RETURN TO ADDR IN LNK REGISTER
*H5 R15 CONTAINS LTK DIVIDED BY 2.
EOJS1 H3 PUT LOAD PT INTO R0 AND JOB CNTRL NAME ADDR INTO REG 1
H4 SVC 11 TASK SELECTION
J3 SVC 4 LOAD PHASE $JOBCTLA
K3 SET UP ENTRY POINT TO $JOBCTLA IN JCTPSW
2

Chart RC. $$BEOJ - Message Output Subroutine
Refer to Chart 15.

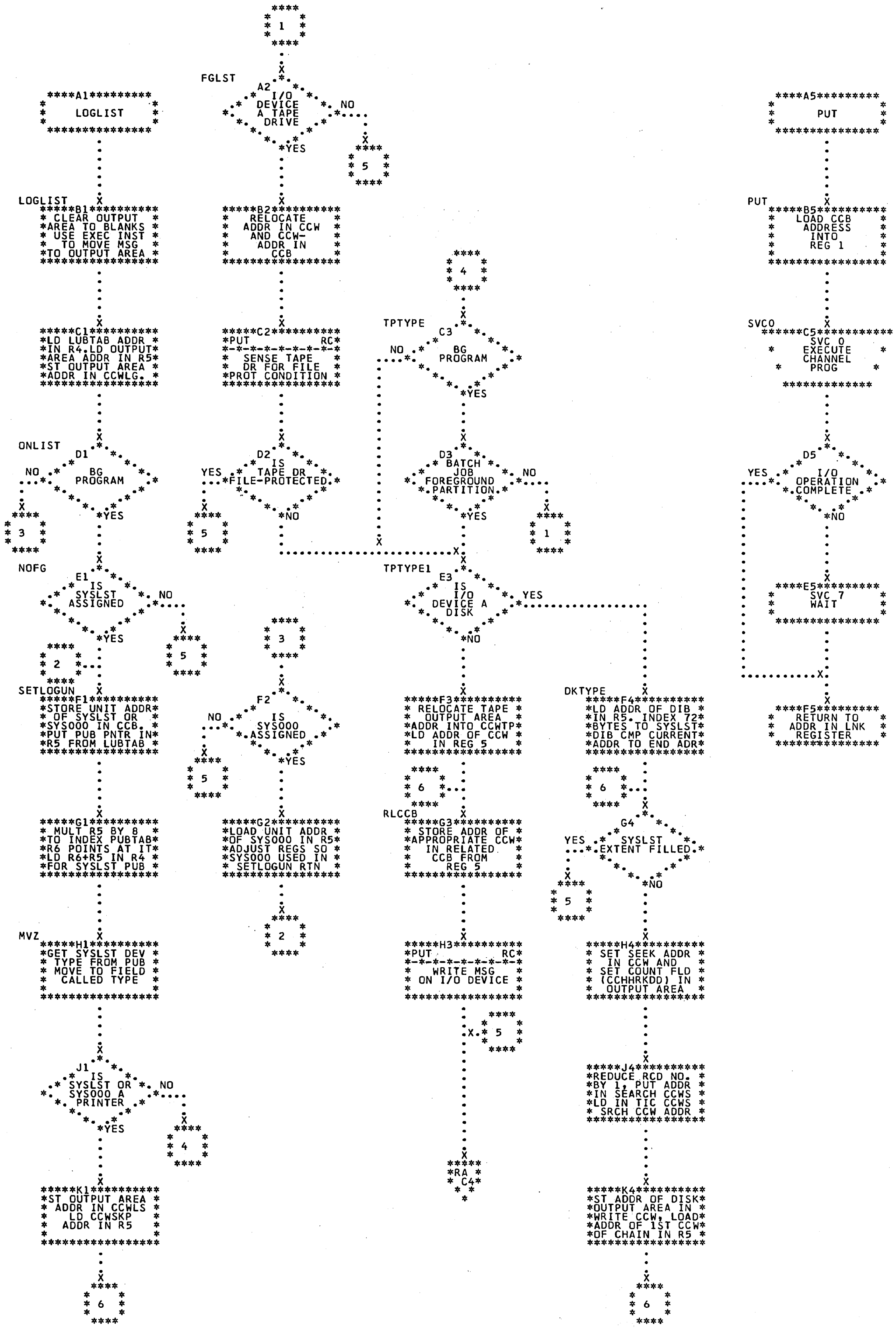

1
FGLST
A2
I/O DEVICE A TAPE DRIVE
NO
5
YES
A1
LOGLIST
LOGLIST
B1
CLEAR OUTPUT AREA TO BLANKS USE EXEC INST TO MOVE MSG TO OUTPUT AREA
B2
RELOCATE ADDR IN CCW AND CCW-ADDR IN CCB
C1
LD LUBTAB ADDR IN R4.LD OUTPUT AREA ADDR IN R5 ST OUTPUT AREA ADDR IN CCWLG.
C2
PUT RC
SENSE TAPE DR FOR FILE PROT CONDITION
ONLIST
D1
BG PROGRAM
NO
3
YES
D2
IS TAPE DR FILE-PROTECTED
YES
5
NO
NOFG
E1
IS SYSLST ASSIGNED
NO
5
YES
2
SETLOGUN
F1
STORE UNIT ADDR OF SYSLST OR SYS000 IN CCB. PUT PUB PNTR IN R5 FROM LUBTAB
3
F2
IS SYS000 ASSIGNED
NO
5
YES
G1
MULT R5 BY 8 TO INDEX PUBTAB R6 POINTS AT IT LD R6+R5 IN R4 FOR SYSLST PUB
G2
LOAD UNIT ADDR OF SYS000 IN R5 ADJUST REGS SO SYS000 USED IN SETLOGUN RTN
2
MVZ
H1
GET SYSLST DEV TYPE FROM PUB MOVE TO FIELD CALLED TYPE
J1
IS SYSLST OR SYS000 A PRINTER
NO
4
YES
K1
ST OUTPUT AREA ADDR IN CCWLS LD CCWSKP ADDR IN R5
6
4
TPTYPE
C3
BG PROGRAM
NO
YES
D3
BATCH JOB FOREGROUND PARTITION
NO
1
YES
TPTYPE1
E3
IS I/O DEVICE A DISK
YES
NO
F3
RELOCATE TAPE OUTPUT AREA ADDR INTO CCWTP LD ADDR OF CCW IN REG 5
6
RLCCB
G3
STORE ADDR OF APPROPRIATE CCW IN RELATED CCB FROM REG 5
H3
PUT RC
WRITE MSG ON I/O DEVICE
5
RA C4
DKTYPE
F4
LD ADDR OF DIB IN R5. INDEX 72 BYTES TO SYSLST DIB CMP CURRENT ADDR TO END ADR
6
G4
SYSLST EXTENT FILLED
YES
5
NO
H4
SET SEEK ADDR IN CCW AND SET COUNT FLD (CCHHRKDD) IN OUTPUT AREA
J4
REDUCE RCD NO. BY 1, PUT ADDR IN SEARCH CCWS LD IN TIC CCWS SRCH CCW ADDR
K4
ST ADDR OF DISK OUTPUT AREA IN WRITE CCW, LOAD ADDR OF 1ST CCW OF CHAIN IN R5
6
A5
PUT
PUT
B5
LOAD CCB ADDRESS INTO REG 1
SVC0
C5
SVC 0 EXECUTE CHANNEL PROG
D5
I/O OPERATION COMPLETE
YES
NO
E5
SVC 7 WAIT
F5
RETURN TO ADDR IN LNK REGISTER

Chart RD. $$BEOJ3 - Quiesce I/O Phase
Refer to Chart 15.

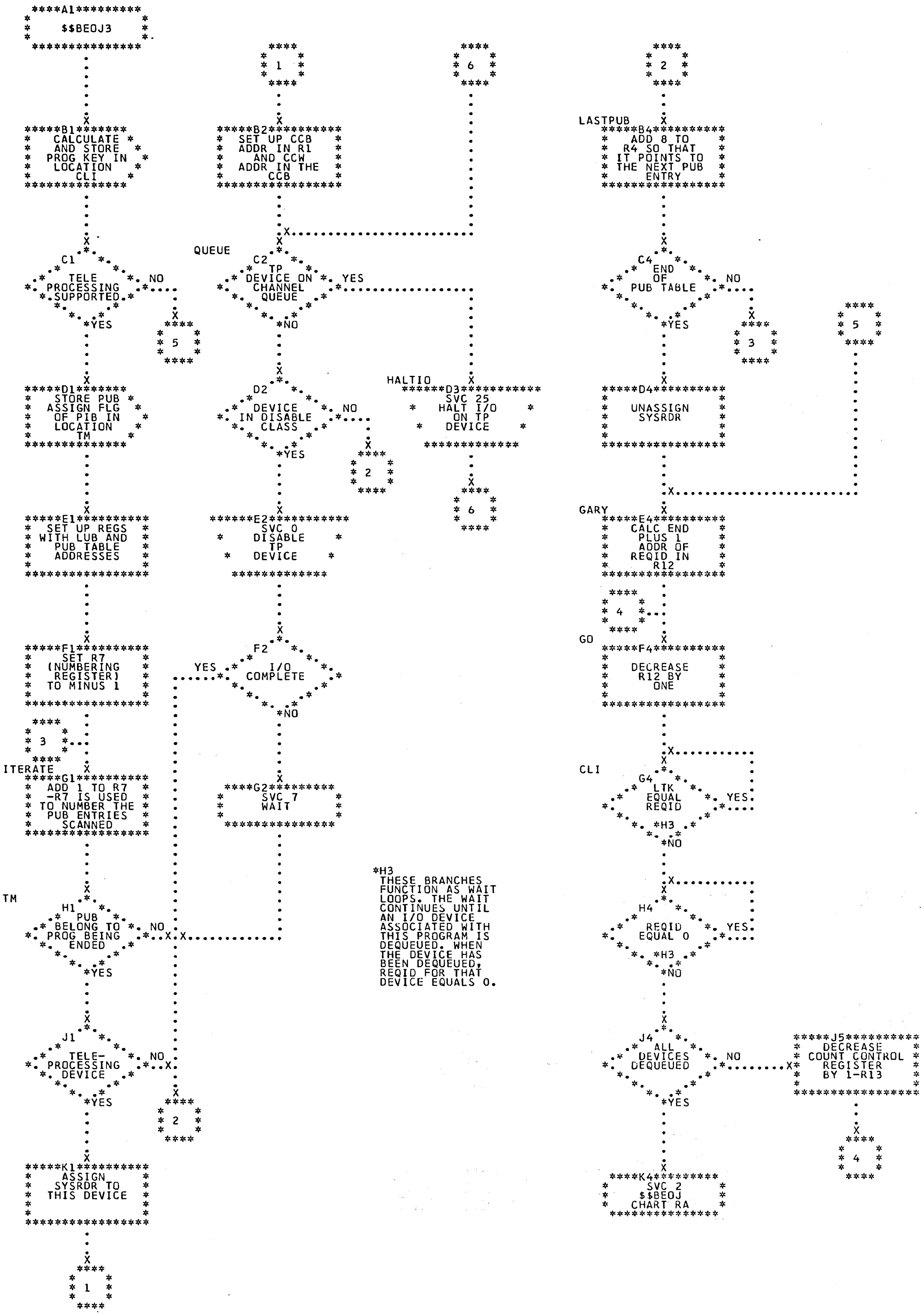
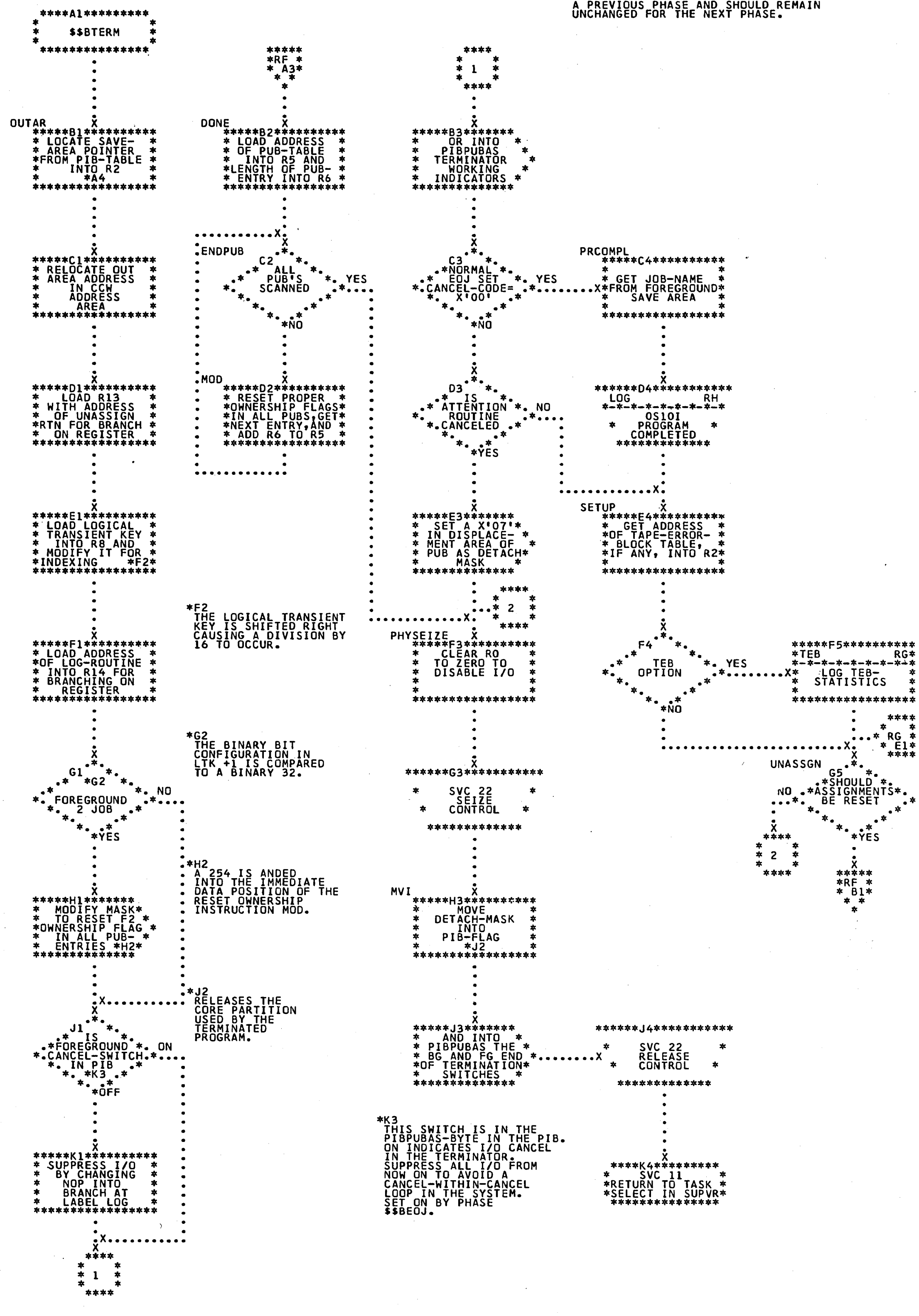

A1
$$BEOJ3
B1
CALCULATE AND STORE PROG KEY IN LOCATION CLI
C1
TELE PROCESSING SUPPORTED
NO
YES
5
D1
STORE PUB ASSIGN FLG OF PIB IN LOCATION TM
E1
SET UP REGS WITH LUB AND PUB TABLE ADDRESSES
F1
SET R7 (NUMBERING REGISTER) TO MINUS 1
3
ITERATE
G1
ADD 1 TO R7 -R7 IS USED TO NUMBER THE PUB ENTRIES SCANNED
TM
H1
PUB BELONG TO PROG BEING ENDED
NO
YES
J1
TELE-PROCESSING DEVICE
NO
YES
2
K1
ASSIGN SYSRDR TO THIS DEVICE
1
1
B2
SET UP CCB ADDR IN R1 AND CCW ADDR IN THE CCB
QUEUE
C2
TP DEVICE ON CHANNEL QUEUE
YES
NO
D2
DEVICE IN DISABLE CLASS
NO
YES
2
E2
SVC 0 DISABLE TP DEVICE
F2
I/O COMPLETE
YES
NO
G2
SVC 7 WAIT
6
HALTIO
D3
SVC 25 HALT I/O ON TP DEVICE
6
*H3
THESE BRANCHES FUNCTION AS WAIT LOOPS. THE WAIT CONTINUES UNTIL AN I/O DEVICE ASSOCIATED WITH THIS PROGRAM IS DEQUEUED. WHEN THE DEVICE HAS BEEN DEQUEUED, REQID FOR THAT DEVICE EQUALS 0.
2
LASTPUB
B4
ADD 8 TO R4 SO THAT IT POINTS TO THE NEXT PUB ENTRY
C4
END OF PUB TABLE
NO
YES
3
D4
UNASSIGN SYSRDR
GARY
E4
CALC END PLUS 1 ADDR OF REQID IN R12
4
GO
F4
DECREASE R12 BY ONE
CLI
G4
LTK EQUAL REQID
YES
*H3
NO
H4
REQID EQUAL 0
YES
*H3
NO
J4
ALL DEVICES DEQUEUED
NO
YES
K4
SVC 2 $$BEOJ CHART RA
5
J5
DECREASE COUNT CONTROL REGISTER BY 1-R13
4

Chart RE. $$BTERM - Reset Foreground PUB Ownership and Detach Attention Routine Refer to Chart 15.


A1
$$BTERM
OUTAR
B1
LOCATE SAVE-AREA POINTER FROM PIB-TABLE INTO R2 *A4
C1
RELOCATE OUT AREA ADDRESS IN CCW ADDRESS AREA
D1
LOAD R13 WITH ADDRESS OF UNASSIGN RTN FOR BRANCH ON REGISTER
E1
LOAD LOGICAL TRANSIENT KEY INTO R8 AND MODIFY IT FOR INDEXING *F2
F1
LOAD ADDRESS OF LOG-ROUTINE INTO R14 FOR BRANCHING ON REGISTER
G1
*G2
FOREGROUND 2 JOB
NO
YES
H1
MODIFY MASK TO RESET F2 OWNERSHIP FLAG IN ALL PUB-ENTRIES *H2
J1
IS FOREGROUND CANCEL-SWITCH IN PIB *K3
ON
OFF
K1
SUPPRESS I/O BY CHANGING NOP INTO BRANCH AT LABEL LOG
1
RF A3
DONE
B2
LOAD ADDRESS OF PUB-TABLE INTO R5 AND LENGTH OF PUB-ENTRY INTO R6
ENDPUB
C2
ALL PUB'S SCANNED
YES
NO
MOD
D2
RESET PROPER OWNERSHIP FLAGS IN ALL PUBS,GET NEXT ENTRY,AND ADD R6 TO R5
1
B3
OR INTO PIBPUBAS TERMINATOR WORKING INDICATORS
C3
NORMAL EOJ SET CANCEL-CODE= X'00'
YES
NO
D3
IS ATTENTION ROUTINE CANCELED
NO
YES
E3
SET A X'07' IN DISPLACE-MENT AREA OF PUB AS DETACH MASK
2
PHYSEIZE
F3
CLEAR R0 TO ZERO TO DISABLE I/O
G3
SVC 22 SEIZE CONTROL
MVI
H3
MOVE DETACH-MASK INTO PIB-FLAG *J2
J3
AND INTO PIBPUBAS THE BG AND FG END OF TERMINATION SWITCHES
PRCOMPL
C4
GET JOB-NAME FROM FOREGROUND SAVE AREA
D4
LOG RH
OS10I PROGRAM COMPLETED
SETUP
E4
GET ADDRESS OF TAPE-ERROR-BLOCK TABLE, IF ANY, INTO R2
F4
TEB OPTION
YES
NO
F5
TEB RG
LOG TEB-STATISTICS
RG E1
UNASSGN
G5
SHOULD ASSIGNMENTS BE RESET
NO
YES
2
RF B1
J4
SVC 22 RELEASE CONTROL
K4
SVC 11 RETURN TO TASK SELECT IN SUPVR

*A4
R10 AND R11 ARE USED AS BASE REGISTERS FOR THE PIB-TABLE AND COMMUNICATICATION REGION RESPECTIVELY AND WERE SET BY A PREVIOUS PHASE AND SHOULD REMAIN UNCHANGED FOR THE NEXT PHASE.

*F2
THE LOGICAL TRANSIENT KEY IS SHIFTED RIGHT CAUSING A DIVISION BY 16 TO OCCUR.

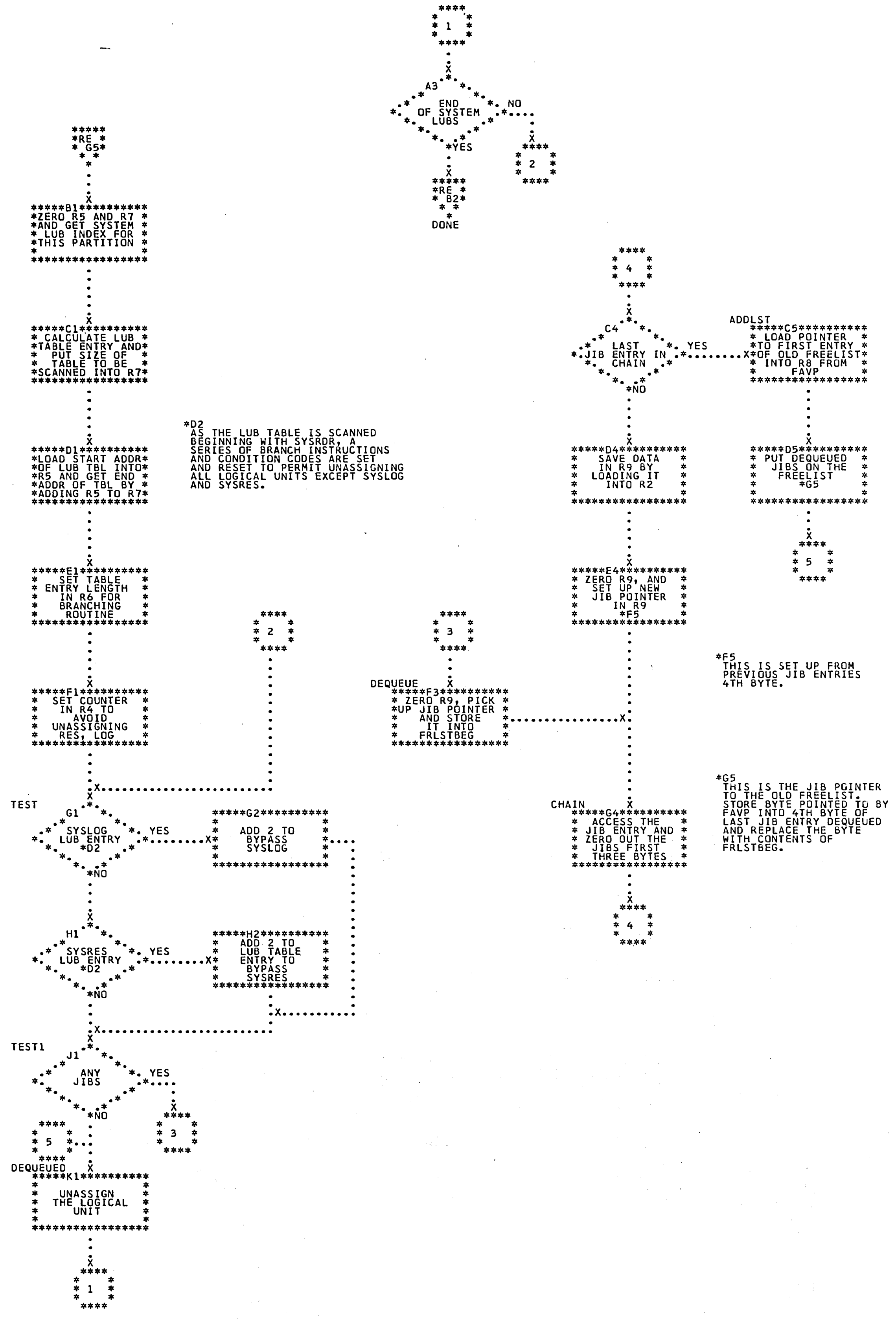
*G2
THE BINARY BIT CONFIGURATION IN LTK +1 IS COMPARED TO A BINARY 32.

*H2
A 254 IS ANDED INTO THE IMMEDIATE DATA POSITION OF THE RESET OWNERSHIP INSTRUCTION MOD.

*J2
RELEASES THE CORE PARTITION USED BY THE TERMINATED PROGRAM.

*K3
THIS SWITCH IS IN THE PIBPUBAS-BYTE IN THE PIB. ON INDICATES I/O CANCEL IN THE TERMINATOR. SUPPRESS ALL I/O FROM NOW ON TO AVOID A CANCEL-WITHIN-CANCEL LOOP IN THE SYSTEM. SET ON BY PHASE $$BEOJ.

Chart RF. $$BTERM - Reset JIBs for I/O Device of Terminated Program Refer to Chart 15.


1
A3
END OF SYSTEM LUBS
NO
YES
2
RE B2
DONE
RE G5
B1
ZERO R5 AND R7 AND GET SYSTEM LUB INDEX FOR THIS PARTITION
C1
CALCULATE LUB TABLE ENTRY AND PUT SIZE OF TABLE TO BE SCANNED INTO R7
D1
LOAD START ADDR OF LUB TBL INTO R5 AND GET END ADDR OF TBL BY ADDING R5 TO R7
E1
SET TABLE ENTRY LENGTH IN R6 FOR BRANCHING ROUTINE
F1
SET COUNTER IN R4 TO AVOID UNASSIGNING RES, LOG
2
TEST
G1
SYSLOG LUB ENTRY *D2
YES
NO
G2
ADD 2 TO BYPASS SYSLOG
H1
SYSRES LUB ENTRY *D2
YES
NO
H2
ADD 2 TO LUB TABLE ENTRY TO BYPASS SYSRES
TEST1
J1
ANY JIBS
YES
NO
3
5
DEQUEUED
K1
UNASSIGN THE LOGICAL UNIT
1
3
DEQUEUE
F3
ZERO R9, PICK UP JIB POINTER AND STORE IT INTO FRLSTBEG
4
C4
LAST JIB ENTRY IN CHAIN
YES
NO
ADDLST
C5
LOAD POINTER TO FIRST ENTRY OF OLD FREELIST INTO R8 FROM FAVP
D4
SAVE DATA IN R9 BY LOADING IT INTO R2
D5
PUT DEQUEUED JIBS ON THE FREELIST *G5
5
E4
ZERO R9, AND SET UP NEW JIB POINTER IN R9 *F5
CHAIN
G4
ACCESS THE JIB ENTRY AND ZERO OUT THE JIBS FIRST THREE BYTES
4

*D2
AS THE LUB TABLE IS SCANNED BEGINNING WITH SYSRDR, A SERIES OF BRANCH INSTRUCTIONS AND CONDITION CODES ARE SET AND RESET TO PERMIT UNASSIGNING ALL LOGICAL UNITS EXCEPT SYSLOG AND SYSRES.

*F5
THIS IS SET UP FROM PREVIOUS JIB ENTRIES 4TH BYTE.

*G5
THIS IS THE JIB POINTER TO THE OLD FREELIST. STORE BYTE POINTED TO BY FAVP INTO 4TH BYTE OF LAST JIB ENTRY DEQUEUED AND REPLACE THE BYTE WITH CONTENTS OF FRLSTBEG.

Chart RG. $$BTERM - Get TEB Statistics and Reset TEBs
Refer to Chart 15.

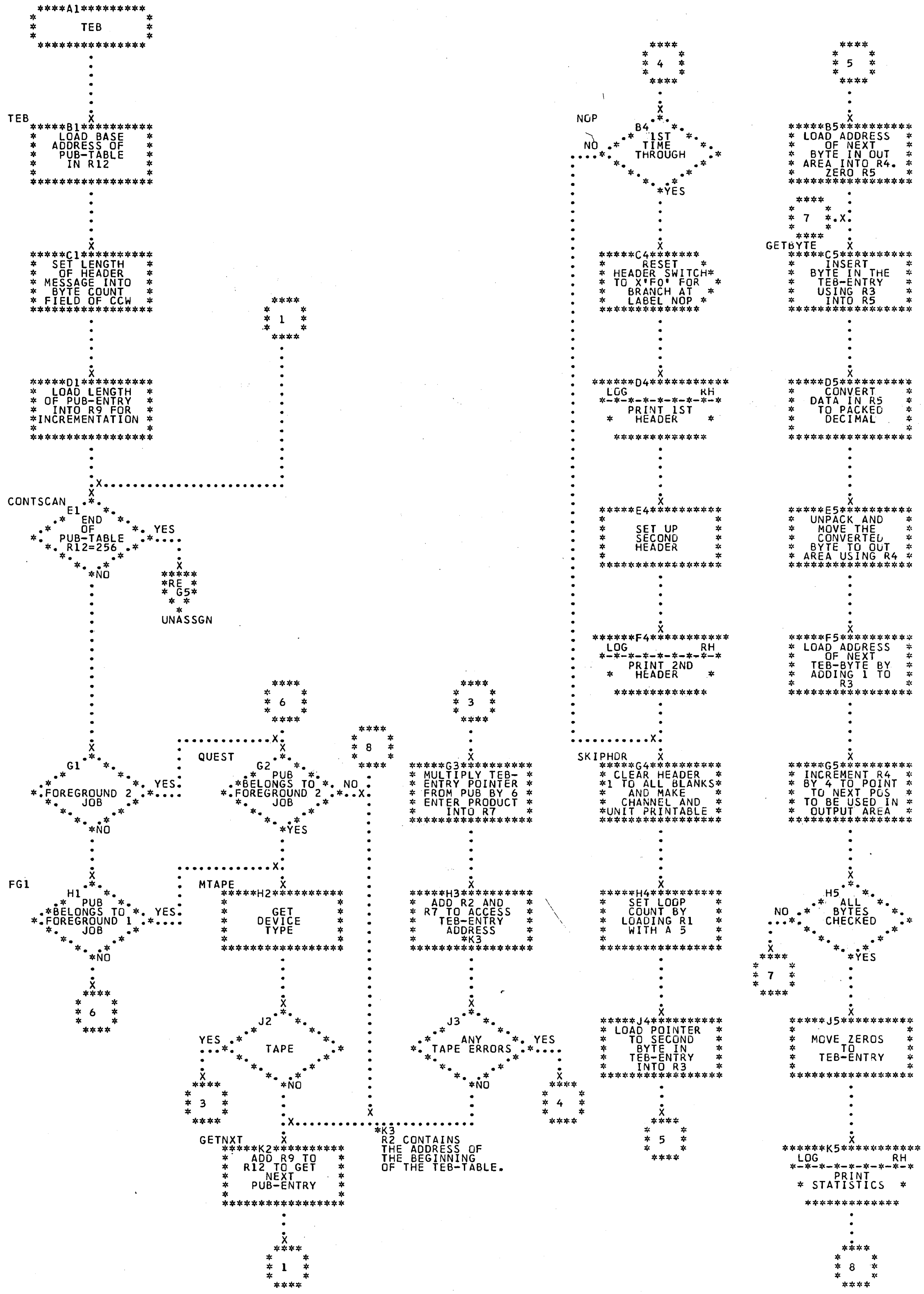

A1
TEB
TEB
B1
LOAD BASE ADDRESS OF PUB-TABLE IN R12
C1
SET LENGTH OF HEADER MESSAGE INTO BYTE COUNT FIELD OF CCW
1
D1
LOAD LENGTH OF PUB-ENTRY INTO R9 FOR INCREMENTATION
CONTSCAN
E1
END OF PUB-TABLE R12=256
YES
NO
RE G5
UNASSGN
6
3
8
G1
FOREGROUND 2 JOB
YES
NO
QUEST
G2
PUB BELONGS TO FOREGROUND 2 JOB
NO
YES
G3
MULTIPLY TEB-ENTRY POINTER FROM PUB BY 6 ENTER PRODUCT INTO R7
FG1
H1
PUB BELONGS TO FOREGROUND 1 JOB
YES
NO
6
MTAPE
H2
GET DEVICE TYPE
H3
ADD R2 AND R7 TO ACCESS TEB-ENTRY ADDRESS
K3
J2
TAPE
YES
NO
3
J3
ANY TAPE ERRORS
YES
NO
4
GETNXT
K2
ADD R9 TO R12 TO GET NEXT PUB-ENTRY
1
K3
R2 CONTAINS THE ADDRESS OF THE BEGINNING OF THE TEB-TABLE.
4
NOP
B4
1ST TIME THROUGH
NO
YES
C4
RESET HEADER SWITCH TO X'F0' FOR BRANCH AT LABEL NOP
D4
LOG RH
PRINT 1ST HEADER
E4
SET UP SECOND HEADER
F4
LOG RH
PRINT 2ND HEADER
SKIPHDR
G4
CLEAR HEADER 1 TO ALL BLANKS AND MAKE CHANNEL AND UNIT PRINTABLE
H4
SET LOOP COUNT BY LOADING R1 WITH A 5
J4
LOAD POINTER TO SECOND BYTE IN TEB-ENTRY INTO R3
5
5
B5
LOAD ADDRESS OF NEXT BYTE IN OUT AREA INTO R4. ZERO R5
7
GETBYTE
C5
INSERT BYTE IN THE TEB-ENTRY USING R3 INTO R5
D5
CONVERT DATA IN R5 TO PACKED DECIMAL
E5
UNPACK AND MOVE THE CONVERTED BYTE TO OUT AREA USING R4
F5
LOAD ADDRESS OF NEXT TEB-BYTE BY ADDING 1 TO R3
G5
INCREMENT R4 BY 4 TO POINT TO NEXT POS TO BE USED IN OUTPUT AREA
H5
ALL BYTES CHECKED
NO
YES
7
J5
MOVE ZEROS TO TEB-ENTRY
K5
LOG RH
PRINT STATISTICS
8

Chart RH. $$BTERM - Print Message and TEB Statistics Subroutine
Refer to Chart 15.

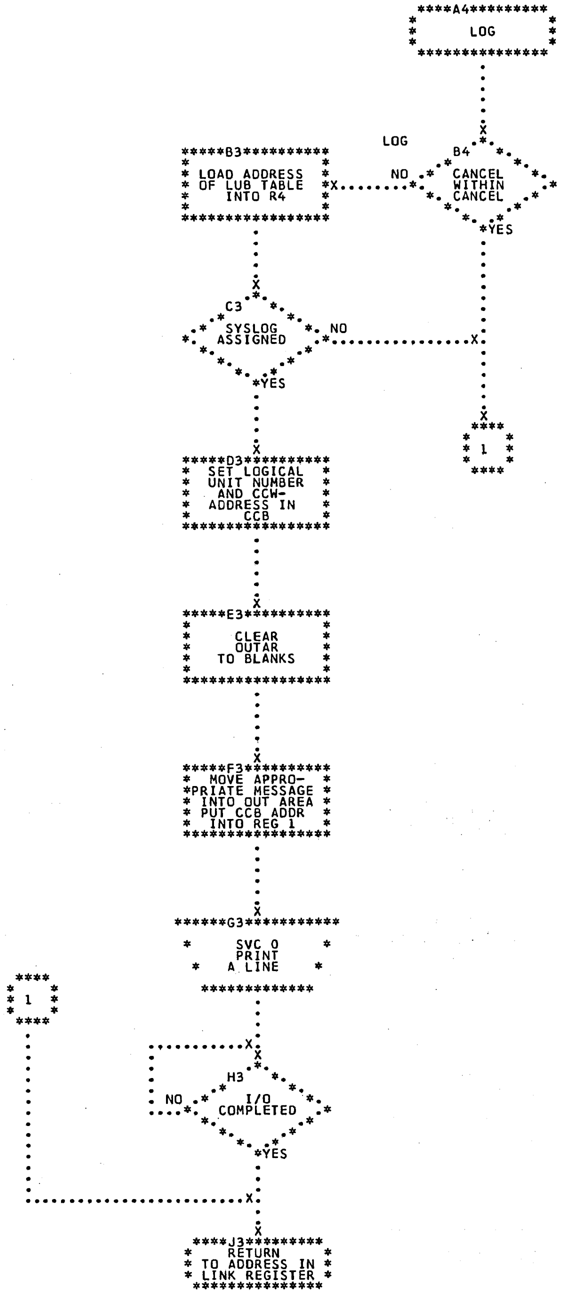

A4
LOG
LOG
B4
CANCEL WITHIN CANCEL
NO
YES
B3
LOAD ADDRESS OF LUB TABLE INTO R4
C3
SYSLOG ASSIGNED
NO
YES
1
D3
SET LOGICAL UNIT NUMBER AND CCW-ADDRESS IN CCB
E3
CLEAR OUTAR TO BLANKS
F3
MOVE APPRO-PRIATE MESSAGE INTO OUT AREA PUT CCB ADDR INTO REG 1
G3
SVC 0 PRINT A LINE
1
H3
I/O COMPLETED
NO
YES
J3
RETURN TO ADDRESS IN LINK REGISTER

Chart RJ. $$BEOJ1 - Prepare Cancel Cause Message
Refer to Chart 16.

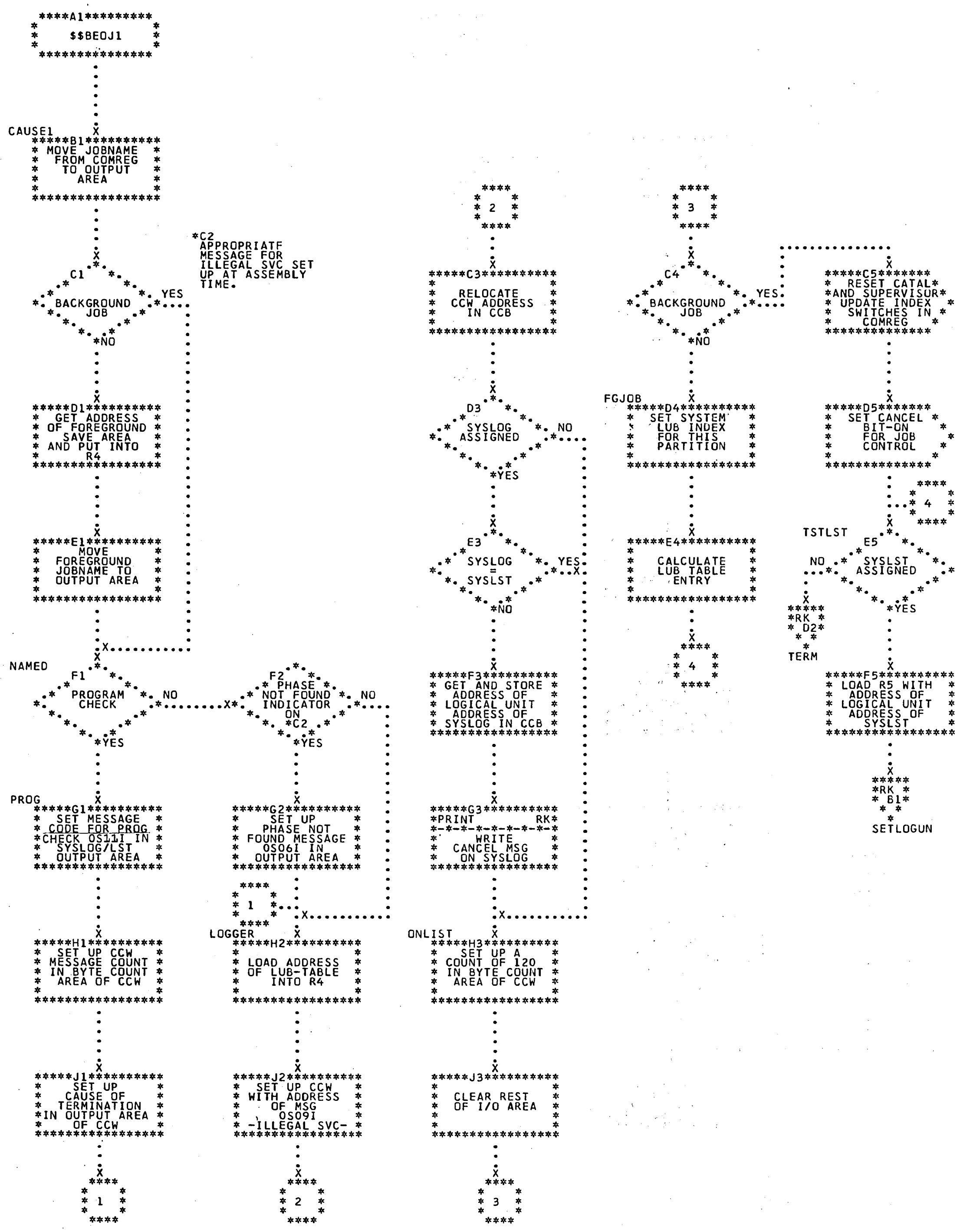

A1
$$BEOJ1
CAUSE1
B1
MOVE JOBNAME FROM COMREG TO OUTPUT AREA
C1
BACKGROUND JOB
YES
NO
*C2
APPROPRIATF MESSAGE FOR ILLEGAL SVC SET UP AT ASSEMBLY TIME.
D1
GET ADDRESS OF FOREGROUND SAVE AREA AND PUT INTO R4
E1
MOVE FOREGROUND JOBNAME TO OUTPUT AREA
NAMED
F1
PROGRAM CHECK
NO
YES
F2
PHASE NOT FOUND INDICATOR ON
*C2
NO
YES
PROG
G1
SET MESSAGE CODE FOR PROG CHECK OS11I IN SYSLOG/LST OUTPUT AREA
G2
SET UP PHASE NOT FOUND MESSAGE OS06I IN OUTPUT AREA
1
LOGGER
H1
SET UP CCW MESSAGE COUNT IN BYTE COUNT AREA OF CCW
H2
LOAD ADDRESS OF LUB-TABLE INTO R4
J1
SET UP CAUSE OF TERMINATION IN OUTPUT AREA OF CCW
J2
SET UP CCW WITH ADDRESS OF MSG OS09I -ILLEGAL SVC-
1
2
2
C3
RELOCATE CCW ADDRESS IN CCB
D3
SYSLOG ASSIGNED
NO
YES
E3
SYSLOG = SYSLST
YES
NO
F3
GET AND STORE ADDRESS OF LOGICAL UNIT ADDRESS OF SYSLOG IN CCB
G3
PRINT RK
WRITE CANCEL MSG ON SYSLOG
ONLIST
H3
SET UP A COUNT OF 120 IN BYTE COUNT AREA OF CCW
J3
CLEAR REST OF I/O AREA
3
3
C4
BACKGROUND JOB
YES
NO
FGJOB
D4
SET SYSTEM LUB INDEX FOR THIS PARTITION
E4
CALCULATE LUB TABLE ENTRY
4
C5
RESET CATAL AND SUPERVISOR UPDATE INDEX SWITCHES IN COMREG
D5
SET CANCEL BIT-ON FOR JOB CONTROL
4
TSTLST
E5
SYSLST ASSIGNED
NO
YES
RK D2
TERM
F5
LOAD R5 WITH ADDRESS OF LOGICAL UNIT ADDRESS OF SYSLST
RK B1
SETLOGUN

Chart RK. $$BEOJ1 - Output Cancel Message on SYSLST
Refer to Chart 16.

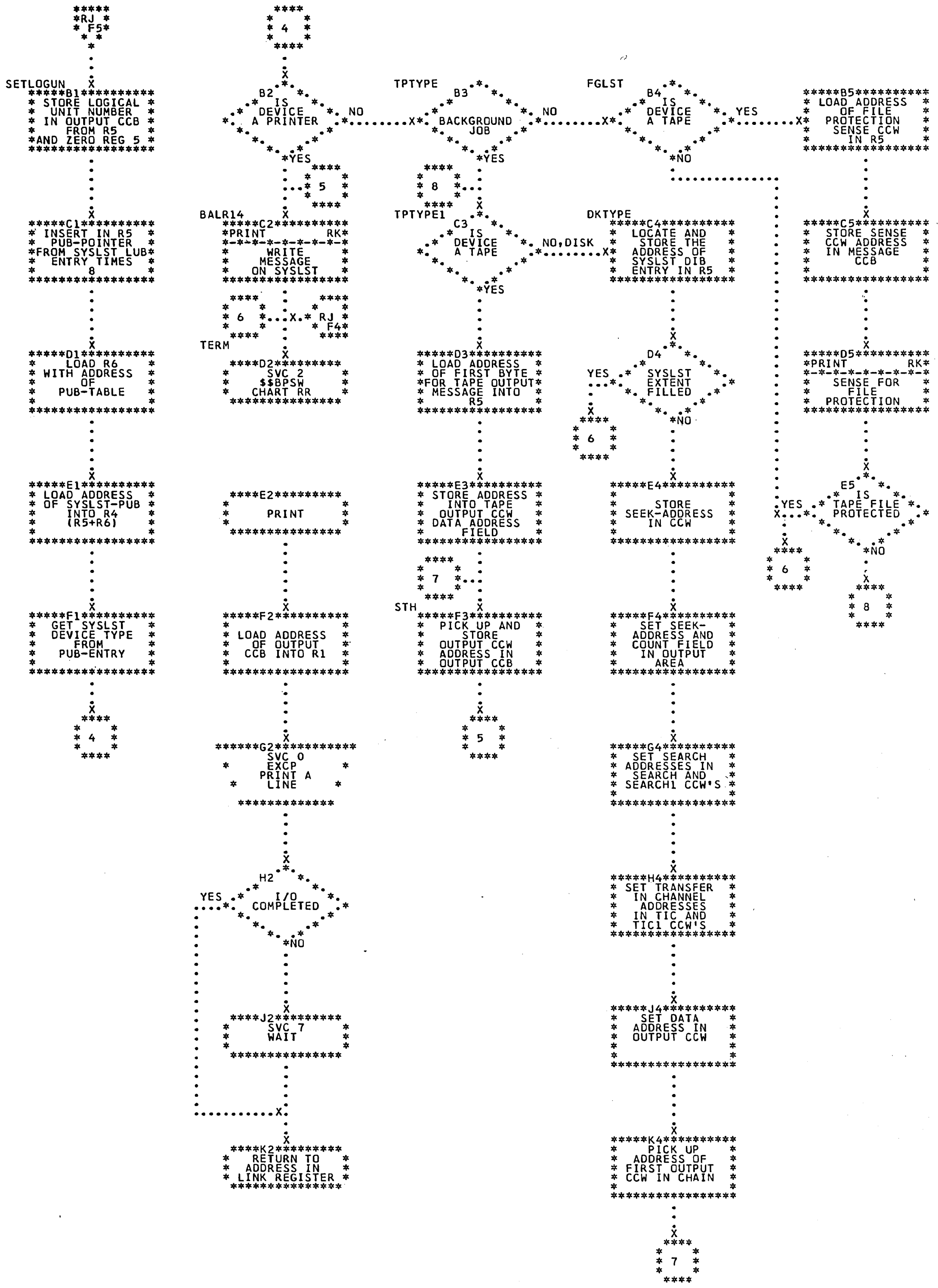
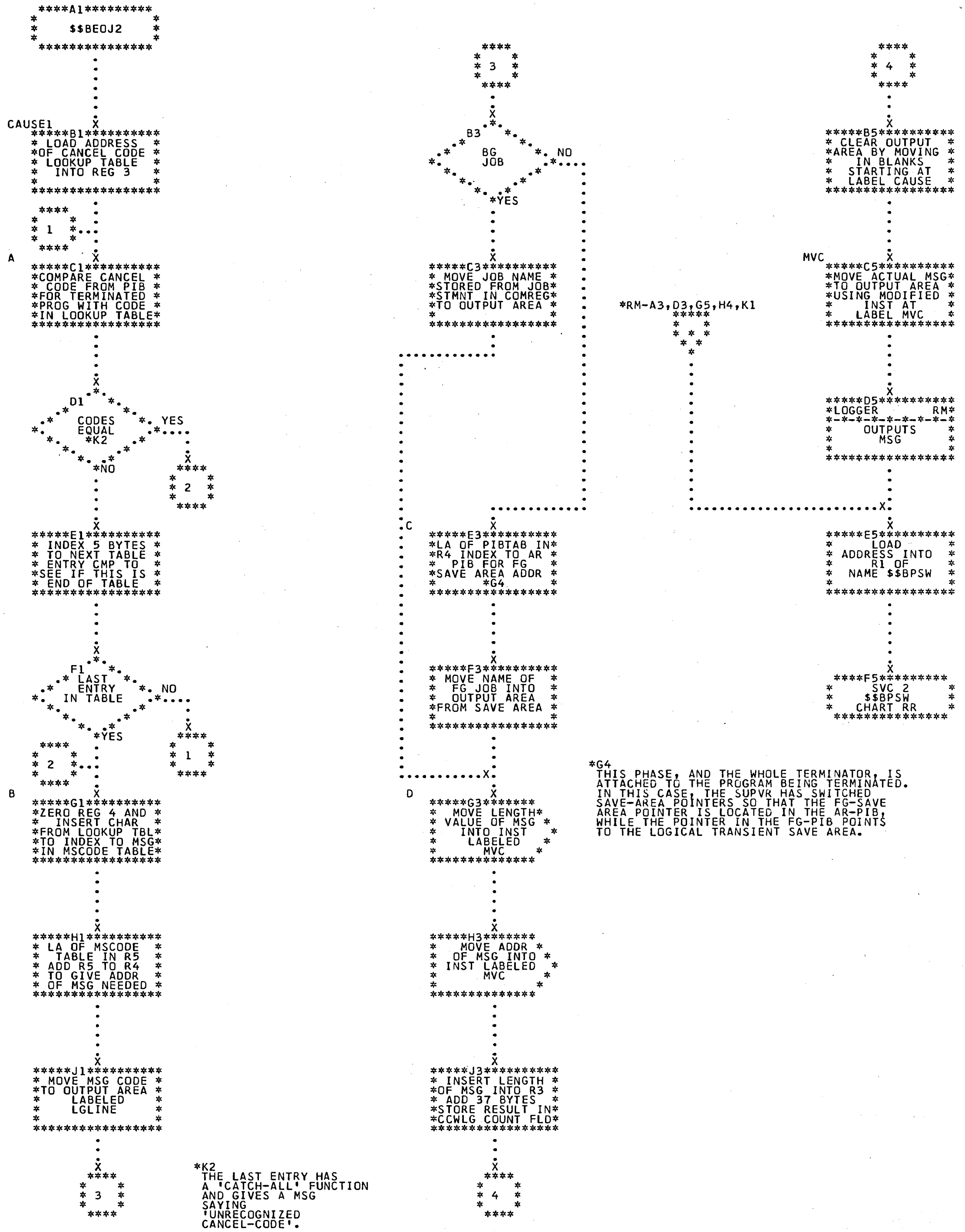

RJ F5
SETLOGUN
B1
STORE LOGICAL UNIT NUMBER IN OUTPUT CCB FROM R5 AND ZERO REG 5
C1
INSERT IN R5 PUB-POINTER FROM SYSLST LUB ENTRY TIMES 8
D1
LOAD R6 WITH ADDRESS OF PUB-TABLE
E1
LOAD ADDRESS OF SYSLST-PUB INTO R4 (R5+R6)
F1
GET SYSLST DEVICE TYPE FROM PUB-ENTRY
4
4
B2
IS DEVICE A PRINTER
NO
YES
5
BALR14
C2
PRINT RK
WRITE MESSAGE ON SYSLST
6
RJ F4
TERM
D2
SVC 2 $$BPSW CHART RR
E2
PRINT
F2
LOAD ADDRESS OF OUTPUT CCB INTO R1
G2
SVC 0 EXCP PRINT A LINE
H2
I/O COMPLETED
YES
NO
J2
SVC 7 WAIT
K2
RETURN TO ADDRESS IN LINK REGISTER
TPTYPE
B3
BACKGROUND JOB
NO
YES
8
TPTYPE1
C3
IS DEVICE A TAPE
NO,DISK
YES
D3
LOAD ADDRESS OF FIRST BYTE FOR TAPE OUTPUT MESSAGE INTO R5
E3
STORE ADDRESS INTO TAPE OUTPUT CCW DATA ADDRESS FIELD
7
STH
F3
PICK UP AND STORE OUTPUT CCW ADDRESS IN OUTPUT CCB
5
FGLST
B4
IS DEVICE A TAPE
YES
NO
DKTYPE
C4
LOCATE AND STORE THE ADDRESS OF SYSLST DIB ENTRY IN R5
D4
SYSLST EXTENT FILLED
YES
NO
6
E4
STORE SEEK-ADDRESS IN CCW
F4
SET SEEK-ADDRESS AND COUNT FIELD IN OUTPUT AREA
G4
SET SEARCH ADDRESSES IN SEARCH AND SEARCH1 CCW'S
H4
SET TRANSFER IN CHANNEL ADDRESSES IN TIC AND TIC1 CCW'S
J4
SET DATA ADDRESS IN OUTPUT CCW
K4
PICK UP ADDRESS OF FIRST OUTPUT CCW IN CHAIN
7
B5
LOAD ADDRESS OF FILE PROTECTION SENSE CCW IN R5
C5
STORE SENSE CCW ADDRESS IN MESSAGE CCB
D5
PRINT RK
SENSE FOR FILE PROTECTION
E5
IS TAPE FILE PROTECTED
YES
NO
6
8

Chart RL. $$BEOJ2 - Select Cancel Message and Program Identification
Refer to Chart 17.


A1
$$BEOJ2
CAUSE1
B1
LOAD ADDRESS OF CANCEL CODE LOOKUP TABLE INTO REG 3
1
A
C1
COMPARE CANCEL CODE FROM PIB FOR TERMINATED PROG WITH CODE IN LOOKUP TABLE
D1
CODES EQUAL *K2
YES
NO
2
E1
INDEX 5 BYTES TO NEXT TABLE ENTRY CMP TO SEE IF THIS IS END OF TABLE
F1
LAST ENTRY IN TABLE
NO
YES
1
2
B
G1
ZERO REG 4 AND INSERT CHAR FROM LOOKUP TBL TO INDEX TO MSG IN MSCODE TABLE
H1
LA OF MSCODE TABLE IN R5 ADD R5 TO R4 TO GIVE ADDR OF MSG NEEDED
J1
MOVE MSG CODE TO OUTPUT AREA LABELED LGLINE
3
3
B3
BG JOB
NO
YES
C3
MOVE JOB NAME STORED FROM JOB STMNT IN COMREG TO OUTPUT AREA
C
E3
LA OF PIBTAB IN R4 INDEX TO AR PIB FOR FG SAVE AREA ADDR *G4
F3
MOVE NAME OF FG JOB INTO OUTPUT AREA FROM SAVE AREA
D
G3
MOVE LENGTH VALUE OF MSG INTO INST LABELED MVC
H3
MOVE ADDR OF MSG INTO INST LABELED MVC
J3
INSERT LENGTH OF MSG INTO R3 ADD 37 BYTES STORE RESULT IN CCWLG COUNT FLD
4
*RM-A3,D3,G5,H4,K1
4
B5
CLEAR OUTPUT AREA BY MOVING IN BLANKS STARTING AT LABEL CAUSE
MVC
C5
MOVE ACTUAL MSG TO OUTPUT AREA USING MODIFIED INST AT LABEL MVC
D5
LOGGER RM
OUTPUTS MSG
E5
LOAD ADDRESS INTO R1 OF NAME $$BPSW
F5
SVC 2 $$BPSW CHART RR

*K2
THE LAST ENTRY HAS A 'CATCH-ALL' FUNCTION AND GIVES A MSG SAYING 'UNRECOGNIZED CANCEL-CODE'.

*G4
THIS PHASE, AND THE WHOLE TERMINATOR, IS ATTACHED TO THE PROGRAM BEING TERMINATED. IN THIS CASE, THE SUPVR HAS SWITCHED SAVE-AREA POINTERS SO THAT THE FG-SAVE AREA POINTER IS LOCATED IN THE AR-PIB, WHILE THE POINTER IN THE FG-PIB POINTS TO THE LOGICAL TRANSIENT SAVE AREA.

Chart RM. $$BEOJ2 - Select I/O Device and Output the Cancel Message
Refer to Chart 17.

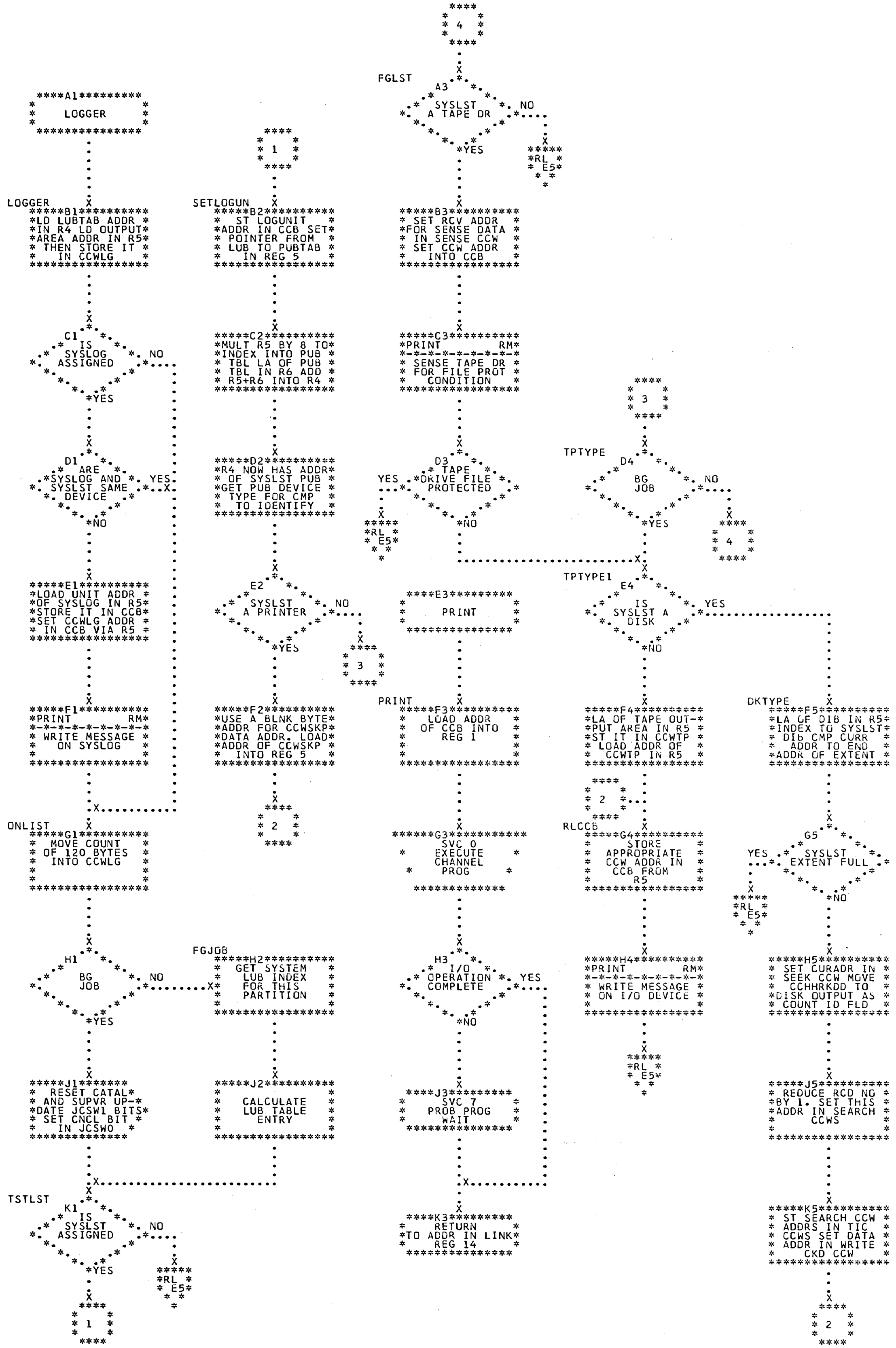

4
FGLST
A3
SYSLST A TAPE DR
NO
YES
RL E5
A1
LOGGER
1
LOGGER
B1
LD LUBTAB ADDR IN R4 LD OUTPUT AREA ADDR IN R5 THEN STORE IT IN CCWLG
SETLOGUN
B2
ST LOGUNIT ADDR IN CCB SET POINTER FROM LUB TO PUBTAB IN REG 5
B3
SET RCV ADDR FOR SENSE DATA IN SENSE CCW SET CCW ADDR INTO CCB
C1
IS SYSLOG ASSIGNED
NO
YES
C2
MULT R5 BY 8 TO INDEX INTO PUB TBL LA OF PUB TBL IN R6 ADD R5+R6 INTO R4
C3
PRINT RM
SENSE TAPE DR FOR FILE PROT CONDITION
3
D1
ARE SYSLOG AND SYSLST SAME DEVICE
YES
NO
D2
R4 NOW HAS ADDR OF SYSLST PUB GET PUB DEVICE TYPE FOR CMP TO IDENTIFY
D3
TAPE DRIVE FILE PROTECTED
YES
NO
RL E5
TPTYPE
D4
BG JOB
NO
YES
4
E1
LOAD UNIT ADDR OF SYSLOG IN R5 STORE IT IN CCB SET CCWLG ADDR IN CCB VIA R5
E2
SYSLST A PRINTER
NO
YES
3
E3
PRINT
TPTYPE1
E4
IS SYSLST A DISK
YES
NO
F1
PRINT RM
WRITE MESSAGE ON SYSLOG
F2
USE A BLNK BYTE ADDR FOR CCWSKP DATA ADDR. LOAD ADDR OF CCWSKP INTO REG 5
2
PRINT
F3
LOAD ADDR OF CCB INTO REG 1
F4
LA OF TAPE OUT- PUT AREA IN R5 ST IT IN CCWTP LOAD ADDR OF CCWTP IN R5
2
DKTYPE
F5
LA OF DIB IN R5 INDEX TO SYSLST DIB CMP CURR ADDR TO END ADDR OF EXTENT
ONLIST
G1
MOVE COUNT OF 120 BYTES INTO CCWLG
G3
SVC 0 EXECUTE CHANNEL PROG
RLCCB
G4
STORE APPROPRIATE CCW ADDR IN CCB FROM R5
G5
SYSLST EXTENT FULL
YES
NO
RL E5
H1
BG JOB
NO
YES
FGJOB
H2
GET SYSTEM LUB INDEX FOR THIS PARTITION
H3
I/O OPERATION COMPLETE
YES
NO
H4
PRINT RM
WRITE MESSAGE ON I/O DEVICE
RL E5
H5
SET CURADR IN SEEK CCW MOVE CCHHRKDD TO DISK OUTPUT AS COUNT ID FLD
J1
RESET CATAL AND SUPVR UP- DATE JCSW1 BITS SET CNCL BIT IN JCSW0
J2
CALCULATE LUB TABLE ENTRY
J3
SVC 7 PROB PROG WAIT
J5
REDUCE RCD NO BY 1. SET THIS ADDR IN SEARCH CCWS
TSTLST
K1
IS SYSLST ASSIGNED
NO
YES
RL E5
1
K3
RETURN TO ADDR IN LINK REG 14
K5
ST SEARCH CCW ADDRS IN TIC CCWS SET DATA ADDR IN WRITE CKD CCW
2

Chart RN. $$BILSVC - Prepare Information about Cancel Cause
Refer to Chart 17.

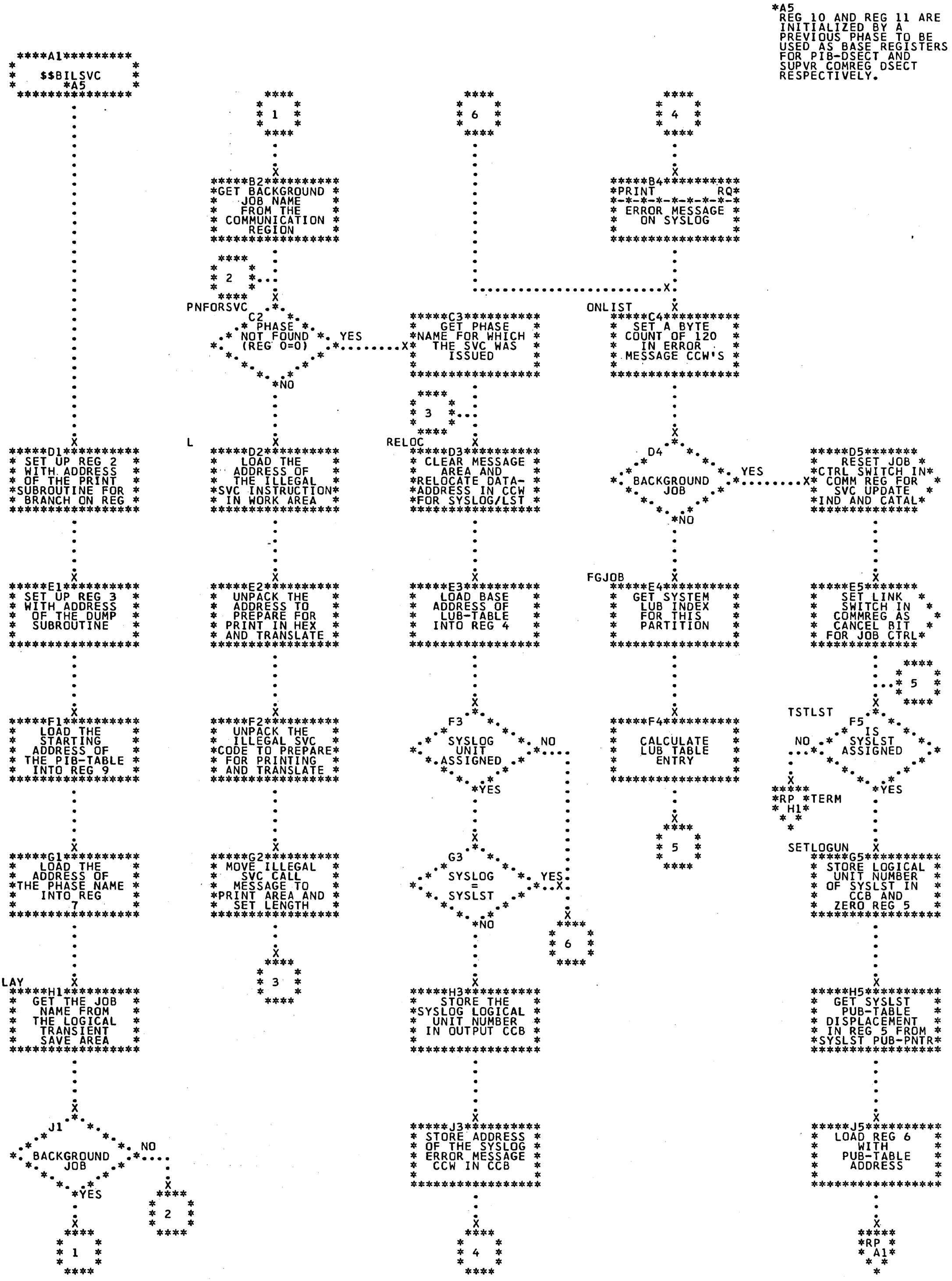

*A5
REG 10 AND REG 11 ARE INITIALIZED BY A PREVIOUS PHASE TO BE USED AS BASE REGISTERS FOR PIB-DSECT AND SUPVR COMREG DSECT RESPECTIVELY.
A1
$$BILSVC
*A5
1
6
4
B2
GET BACKGROUND JOB NAME FROM THE COMMUNICATION REGION
B4
PRINT RQ
ERROR MESSAGE ON SYSLOG
2
PNFORSVC
C2
PHASE NOT FOUND (REG 0=0)
YES
NO
C3
GET PHASE NAME FOR WHICH THE SVC WAS ISSUED
ONLIST
C4
SET A BYTE COUNT OF 120 IN ERROR MESSAGE CCW'S
3
L
RELOC
D1
SET UP REG 2 WITH ADDRESS OF THE PRINT SUBROUTINE FOR BRANCH ON REG
D2
LOAD THE ADDRESS OF THE ILLEGAL SVC INSTRUCTION IN WORK AREA
D3
CLEAR MESSAGE AREA AND RELOCATE DATA-ADDRESS IN CCW FOR SYSLOG/LST
D4
BACKGROUND JOB
YES
NO
D5
RESET JOB CTRL SWITCH IN COMM REG FOR SVC UPDATE IND AND CATAL
E1
SET UP REG 3 WITH ADDRESS OF THE DUMP SUBROUTINE
E2
UNPACK THE ADDRESS TO PREPARE FOR PRINT IN HEX AND TRANSLATE
E3
LOAD BASE ADDRESS OF LUB-TABLE INTO REG 4
FGJOB
E4
GET SYSTEM LUB INDEX FOR THIS PARTITION
E5
SET LINK SWITCH IN COMMREG AS CANCEL BIT FOR JOB CTRL
5
F1
LOAD THE STARTING ADDRESS OF THE PIB-TABLE INTO REG 9
F2
UNPACK THE ILLEGAL SVC CODE TO PREPARE FOR PRINTING AND TRANSLATE
F3
SYSLOG UNIT ASSIGNED
NO
YES
F4
CALCULATE LUB TABLE ENTRY
TSTLST
F5
IS SYSLST ASSIGNED
NO
YES
RP
TERM
H1
5
G1
LOAD THE ADDRESS OF THE PHASE NAME INTO REG 7
G2
MOVE ILLEGAL SVC CALL MESSAGE TO PRINT AREA AND SET LENGTH
G3
SYSLOG = SYSLST
YES
NO
6
SETLOGUN
G5
STORE LOGICAL UNIT NUMBER OF SYSLST IN CCB AND ZERO REG 5
3
OVLAY
H1
GET THE JOB NAME FROM THE LOGICAL TRANSIENT SAVE AREA
H3
STORE THE SYSLOG LOGICAL UNIT NUMBER IN OUTPUT CCB
H5
GET SYSLST PUB-TABLE DISPLACEMENT IN REG 5 FROM SYSLST PUB-PNTR
J1
BACKGROUND JOB
NO
YES
2
J3
STORE ADDRESS OF THE SYSLOG ERROR MESSAGE CCW IN CCB
J5
LOAD REG 6 WITH PUB-TABLE ADDRESS
1
4
RP
A1

Chart RP. $$BILSVC - Select I/O Device and Prepare to Output a Message
Refer to Chart 17.

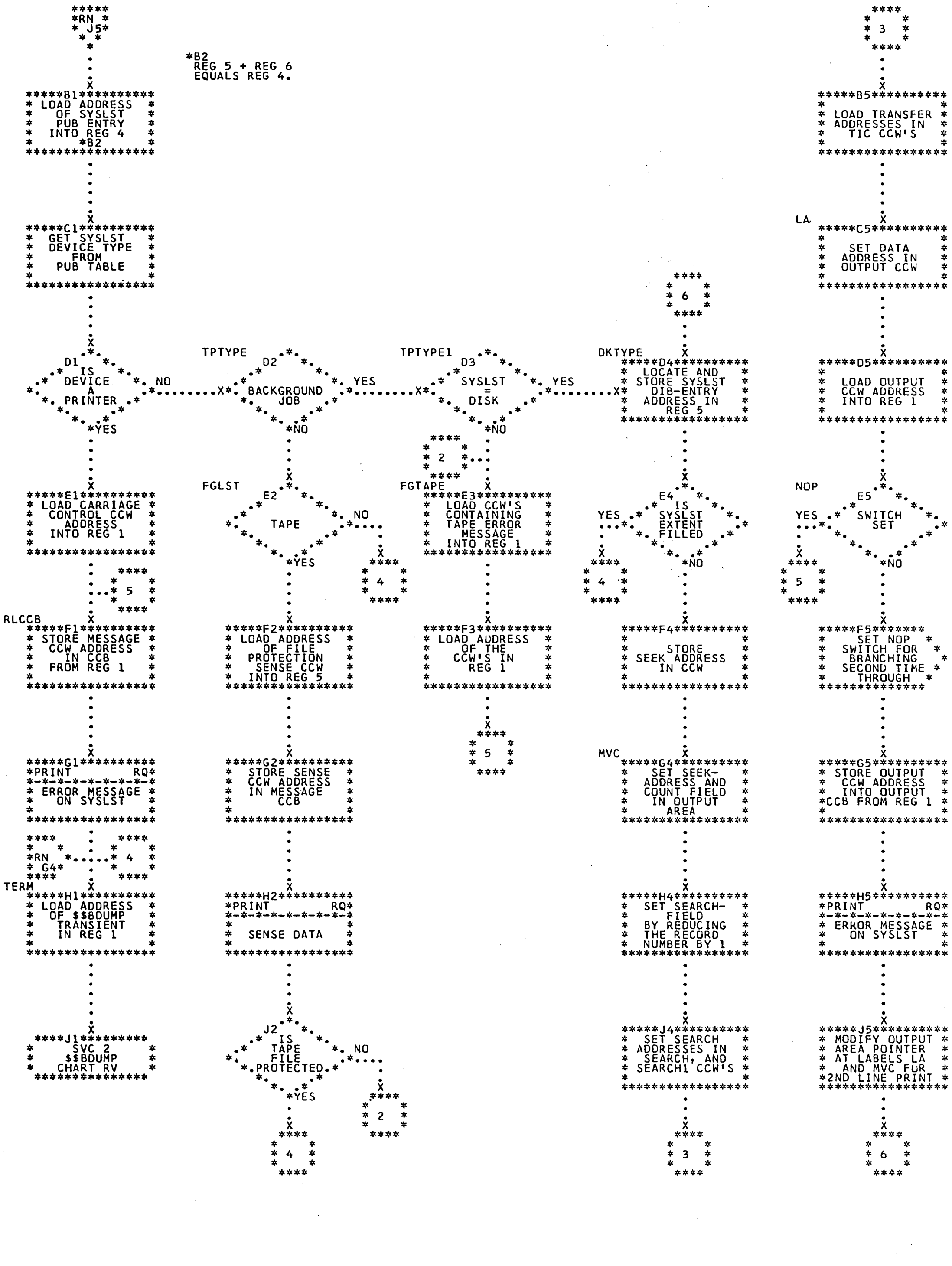

RN J5
*B2 REG 5 + REG 6 EQUALS REG 4.
B1 LOAD ADDRESS OF SYSLST PUB ENTRY INTO REG 4 *B2
C1 GET SYSLST DEVICE TYPE FROM PUB TABLE
D1 IS DEVICE A PRINTER
NO
YES
E1 LOAD CARRIAGE CONTROL CCW ADDRESS INTO REG 1
5
RLCCB
F1 STORE MESSAGE CCW ADDRESS IN CCB FROM REG 1
G1 PRINT RQ ERROR MESSAGE ON SYSLST
RN G4
4
TERM
H1 LOAD ADDRESS OF $$BDUMP TRANSIENT IN REG 1
J1 SVC 2 $$BDUMP CHART RV
TPTYPE
D2 BACKGROUND JOB
YES
NO
FGLST
E2 TAPE
NO
YES
4
F2 LOAD ADDRESS OF FILE PROTECTION SENSE CCW INTO REG 5
G2 STORE SENSE CCW ADDRESS IN MESSAGE CCB
H2 PRINT RQ SENSE DATA
J2 IS TAPE FILE PROTECTED
NO
YES
2
4
TPTYPE1
D3 SYSLST = DISK
YES
NO
2
FGTAPE
E3 LOAD CCW'S CONTAINING TAPE ERROR MESSAGE INTO REG 1
F3 LOAD ADDRESS OF THE CCW'S IN REG 1
5
6
DKTYPE
D4 LOCATE AND STORE SYSLST DIB-ENTRY ADDRESS IN REG 5
E4 IS SYSLST EXTENT FILLED
YES
NO
4
F4 STORE SEEK ADDRESS IN CCW
MVC
G4 SET SEEK-ADDRESS AND COUNT FIELD IN OUTPUT AREA
H4 SET SEARCH-FIELD BY REDUCING THE RECORD NUMBER BY 1
J4 SET SEARCH ADDRESSES IN SEARCH, AND SEARCH1 CCW'S
3
3
B5 LOAD TRANSFER ADDRESSES IN TIC CCW'S
LA
C5 SET DATA ADDRESS IN OUTPUT CCW
D5 LOAD OUTPUT CCW ADDRESS INTO REG 1
NOP
E5 SWITCH SET
YES
NO
5
F5 SET NOP SWITCH FOR BRANCHING SECOND TIME THROUGH
G5 STORE OUTPUT CCW ADDRESS INTO OUTPUT CCB FROM REG 1
H5 PRINT RQ ERROR MESSAGE ON SYSLST
J5 MODIFY OUTPUT AREA POINTER AT LABELS LA AND MVC FOR 2ND LINE PRINT
6

Chart RQ. $$BILSVC - Output Message on Selected I/O Device
Refer to Chart 17.

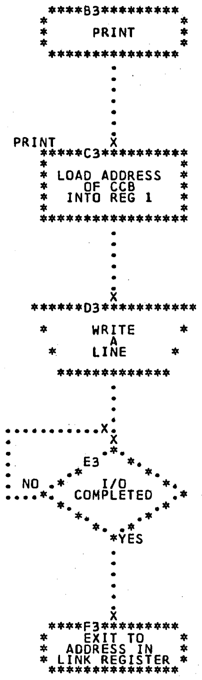

B3
PRINT
PRINT
C3
LOAD ADDRESS OF CCB INTO REG 1
D3
WRITE A LINE
E3
I/O COMPLETED
NO
YES
F3
EXIT TO ADDRESS IN LINK REGISTER

Chart RR. $$BPSW - Prepare Canceled Program's PSW for Output Message and PIOCS Subroutine
Refer to Chart 16.

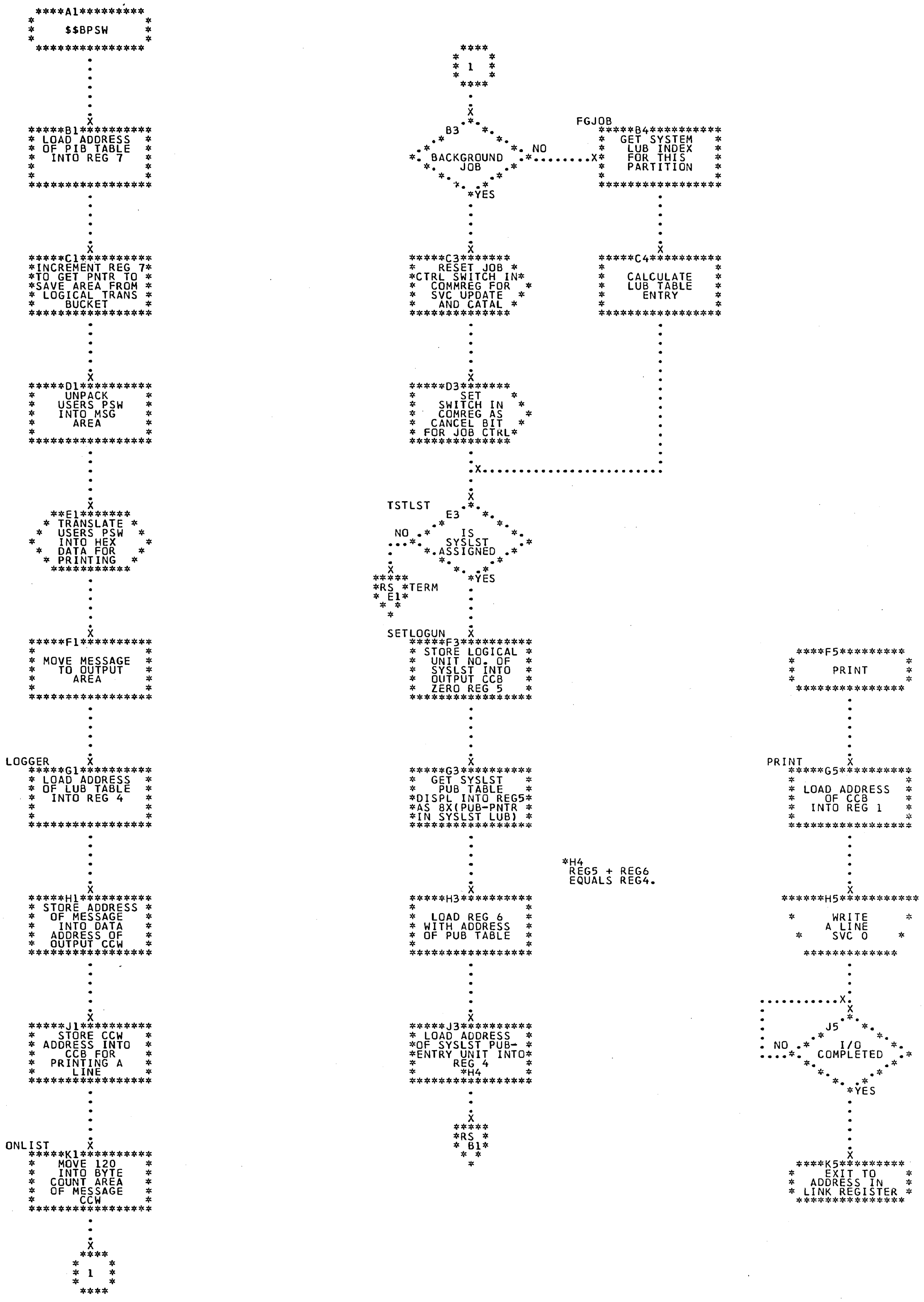
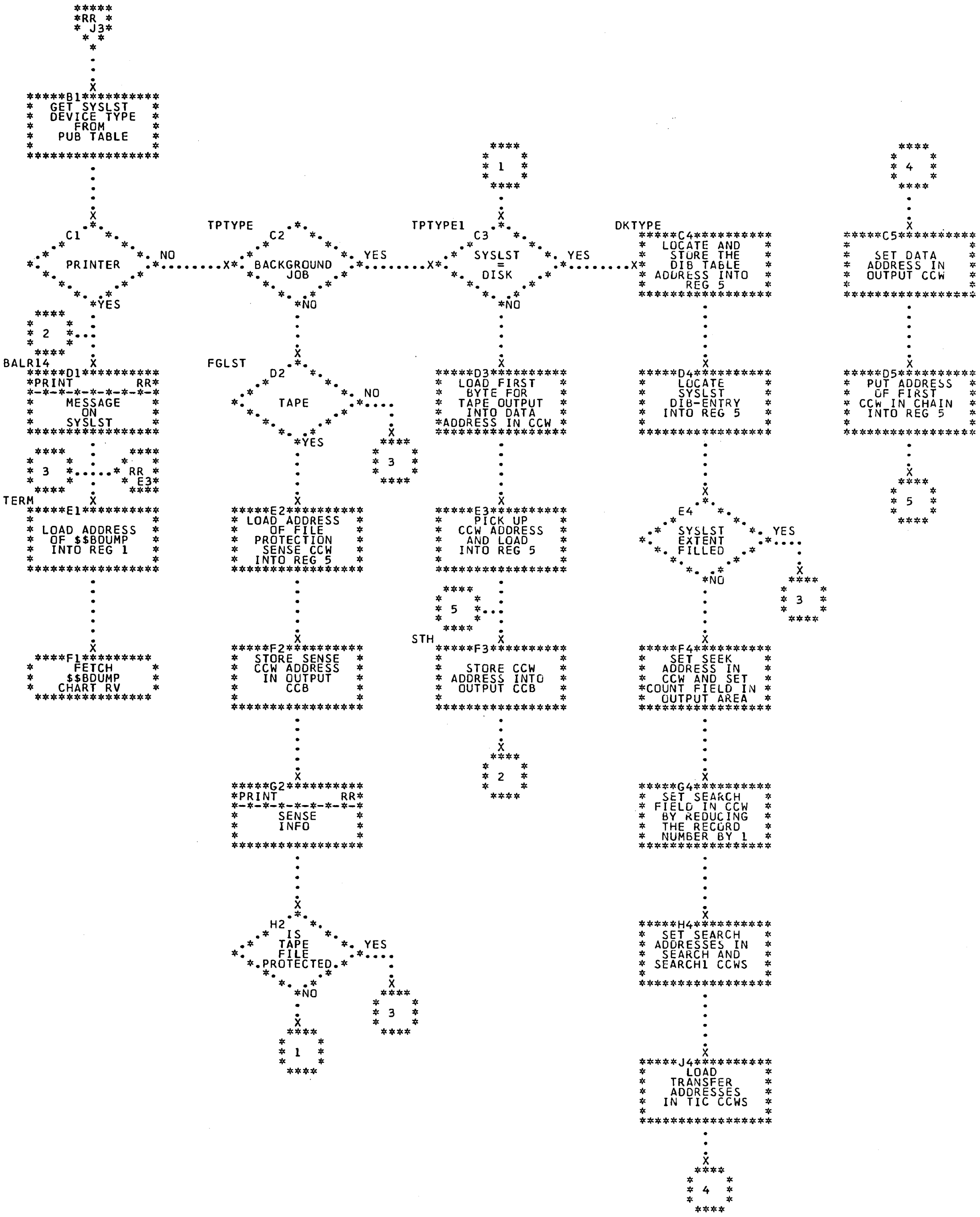

A1
$$BPSW
B1
LOAD ADDRESS OF PIB TABLE INTO REG 7
C1
INCREMENT REG 7 TO GET PNTR TO SAVE AREA FROM LOGICAL TRANS BUCKET
D1
UNPACK USERS PSW INTO MSG AREA
E1
TRANSLATE USERS PSW INTO HEX DATA FOR PRINTING
F1
MOVE MESSAGE TO OUTPUT AREA
LOGGER
G1
LOAD ADDRESS OF LUB TABLE INTO REG 4
H1
STORE ADDRESS OF MESSAGE INTO DATA ADDRESS OF OUTPUT CCW
J1
STORE CCW ADDRESS INTO CCB FOR PRINTING A LINE
ONLIST
K1
MOVE 120 INTO BYTE COUNT AREA OF MESSAGE CCW
1
1
B3
BACKGROUND JOB
NO
YES
FGJOB
B4
GET SYSTEM LUB INDEX FOR THIS PARTITION
C3
RESET JOB CTRL SWITCH IN COMMREG FOR SVC UPDATE AND CATAL
C4
CALCULATE LUB TABLE ENTRY
D3
SET SWITCH IN COMREG AS CANCEL BIT FOR JOB CTRL
TSTLST
E3
IS SYSLST ASSIGNED
NO
YES
RS E1
TERM
SETLOGUN
F3
STORE LOGICAL UNIT NO. OF SYSLST INTO OUTPUT CCB ZERO REG 5
G3
GET SYSLST PUB TABLE DISPL INTO REG5 AS 8X(PUB-PNTR IN SYSLST LUB)
H3
LOAD REG 6 WITH ADDRESS OF PUB TABLE
J3
LOAD ADDRESS OF SYSLST PUB-ENTRY UNIT INTO REG 4
*H4
RS B1
*H4
REG5 + REG6 EQUALS REG4.
F5
PRINT
PRINT
G5
LOAD ADDRESS OF CCB INTO REG 1
H5
WRITE A LINE SVC 0
J5
I/O COMPLETED
NO
YES
K5
EXIT TO ADDRESS IN LINK REGISTER

Chart RS. $$BPSW - Select I/O Device and Prepare to Output a Message
Refer to Chart 16.



RR J3
B1 GET SYSLST DEVICE TYPE FROM PUB TABLE
C1 PRINTER
NO
YES
2
BALR14
D1 PRINT RR MESSAGE ON SYSLST
3
RR E3
TERM
E1 LOAD ADDRESS OF $$BDUMP INTO REG 1
F1 FETCH $$BDUMP CHART RV
TPTYPE
C2 BACKGROUND JOB
YES
NO
FGLST
D2 TAPE
NO
YES
3
E2 LOAD ADDRESS OF FILE PROTECTION SENSE CCW INTO REG 5
F2 STORE SENSE CCW ADDRESS IN OUTPUT CCB
G2 PRINT RR SENSE INFO
H2 IS TAPE FILE PROTECTED
YES
3
NO
1
1
TPTYPE1
C3 SYSLST = DISK
YES
NO
D3 LOAD FIRST BYTE FOR TAPE OUTPUT INTO DATA ADDRESS IN CCW
E3 PICK UP CCW ADDRESS AND LOAD INTO REG 5
5
STH
F3 STORE CCW ADDRESS INTO OUTPUT CCB
2
DKTYPE
C4 LOCATE AND STORE THE DIB TABLE ADDRESS INTO REG 5
D4 LOCATE SYSLST DIB-ENTRY INTO REG 5
E4 SYSLST EXTENT FILLED
YES
3
NO
F4 SET SEEK ADDRESS IN CCW AND SET COUNT FIELD IN OUTPUT AREA
G4 SET SEARCH FIELD IN CCW BY REDUCING THE RECORD NUMBER BY 1
H4 SET SEARCH ADDRESSES IN SEARCH AND SEARCH1 CCWS
J4 LOAD TRANSFER ADDRESSES IN TIC CCWS
4
4
C5 SET DATA ADDRESS IN OUTPUT CCW
D5 PUT ADDRESS OF FIRST CCW IN CHAIN INTO REG 5
5

Chart RT. $$BPCHK - Prepare Information for Message about PC Cancel and Select I/O Device

Refer to Chart 17.

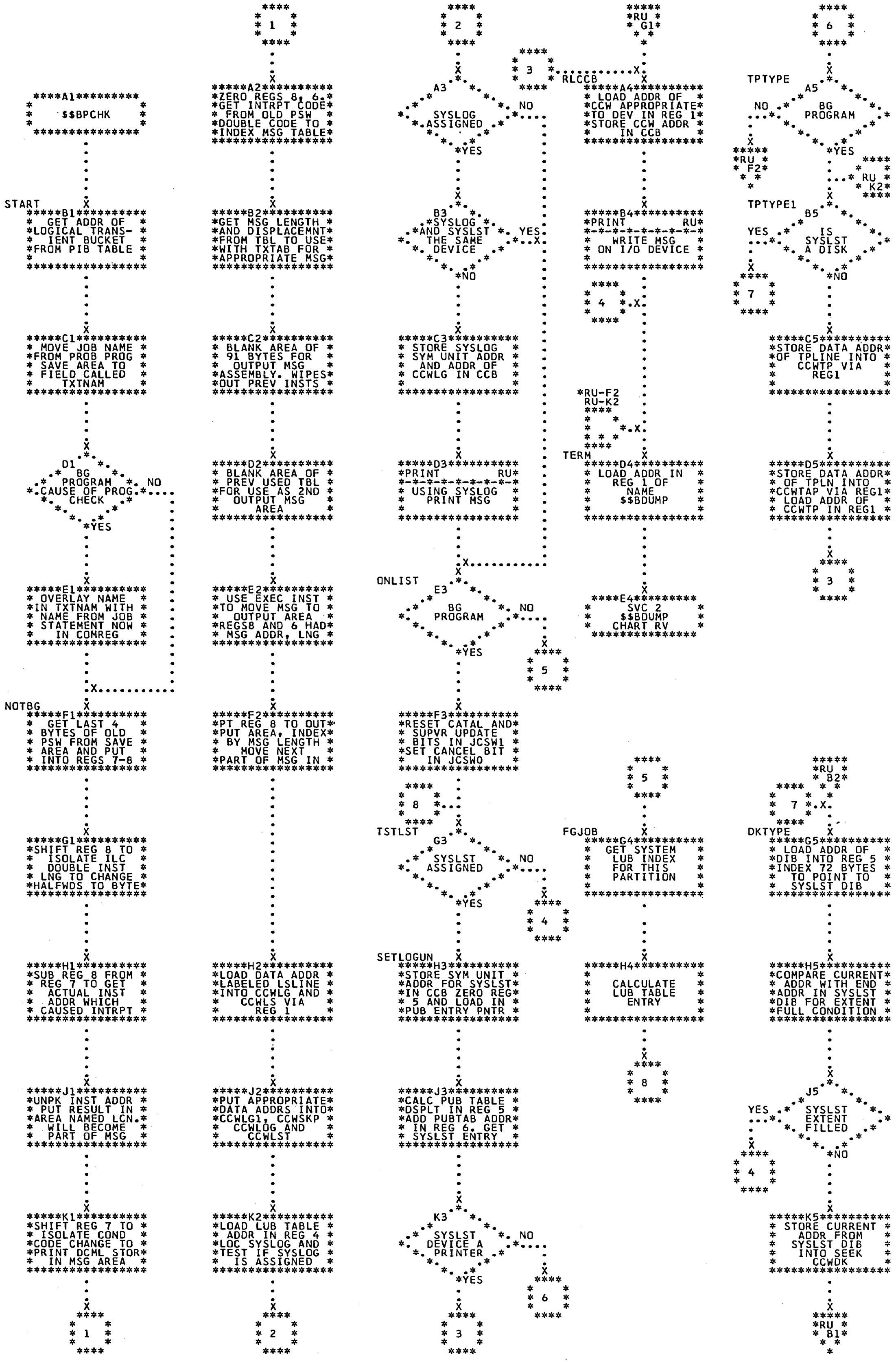

A1
$$BPCHK
START
B1
GET ADDR OF LOGICAL TRANS- IENT BUCKET FROM PIB TABLE
C1
MOVE JOB NAME FROM PROB PROG SAVE AREA TO FIELD CALLED TXTNAM
D1
BG PROGRAM CAUSE OF PROG. CHECK
NO
YES
E1
OVERLAY NAME IN TXTNAM WITH NAME FROM JOB STATEMENT NOW IN COMREG
NOTBG
F1
GET LAST 4 BYTES OF OLD PSW FROM SAVE AREA AND PUT INTO REGS 7-8
G1
SHIFT REG 8 TO ISOLATE ILC DOUBLE INST LNG TO CHANGE HALFWDS TO BYTE
H1
SUB REG 8 FROM REG 7 TO GET ACTUAL INST ADDR WHICH CAUSED INTRPT
J1
UNPK INST ADDR PUT RESULT IN AREA NAMED LCN. WILL BECOME PART OF MSG
K1
SHIFT REG 7 TO ISOLATE COND CODE CHANGE TO PRINT DCML STOR IN MSG AREA
1
1
A2
ZERO REGS 8, 6. GET INTRPT CODE FROM OLD PSW DOUBLE CODE TO INDEX MSG TABLE
B2
GET MSG LENGTH AND DISPLACEMNT FROM TBL TO USE WITH TXTAB FOR APPROPRIATE MSG
C2
BLANK AREA OF 91 BYTES FOR OUTPUT MSG ASSEMBLY. WIPES OUT PREV INSTS
D2
BLANK AREA OF PREV USED TBL FOR USE AS 2ND OUTPUT MSG AREA
E2
USE EXEC INST TO MOVE MSG TO OUTPUT AREA REGS8 AND 6 HAD MSG ADDR, LNG
F2
PT REG 8 TO OUTPUT AREA, INDEX BY MSG LENGTH MOVE NEXT PART OF MSG IN
H2
LOAD DATA ADDR LABELED LSLINE INTO CCWLG AND CCWLS VIA REG 1
J2
PUT APPROPRIATE DATA ADDRS INTO CCWLG1, CCWSKP CCWLOG AND CCWLST
K2
LOAD LUB TABLE ADDR IN REG 4 LOC SYSLOG AND TEST IF SYSLOG IS ASSIGNED
2
2
A3
SYSLOG ASSIGNED
NO
YES
B3
SYSLOG AND SYSLST THE SAME DEVICE
YES
NO
C3
STORE SYSLOG SYM UNIT ADDR AND ADDR OF CCWLG IN CCB
D3
PRINT RU
USING SYSLOG PRINT MSG
ONLIST
E3
BG PROGRAM
NO
YES
5
F3
RESET CATAL AND SUPVR UPDATE BITS IN JCSW1 SET CANCEL BIT IN JCSW0
8
TSTLST
G3
SYSLST ASSIGNED
NO
YES
4
SETLOGUN
H3
STORE SYM UNIT ADDR FOR SYSLST IN CCB ZERO REG 5 AND LOAD IN PUB ENTRY PNTR
J3
CALC PUB TABLE DSPLT IN REG 5 ADD PUBTAB ADDR IN REG 6. GET SYSLST ENTRY
K3
SYSLST DEVICE A PRINTER
NO
YES
6
3
3
RU G1
RLCCB
A4
LOAD ADDR OF CCW APPROPRIATE TO DEV IN REG 1 STORE CCW ADDR IN CCB
B4
PRINT RU
WRITE MSG ON I/O DEVICE
4
RU-F2 RU-K2
TERM
D4
LOAD ADDR IN REG 1 OF NAME $$BDUMP
E4
SVC 2 $$BDUMP CHART RV
5
FGJOB
G4
GET SYSTEM LUB INDEX FOR THIS PARTITION
H4
CALCULATE LUB TABLE ENTRY
8
6
TPTYPE
A5
BG PROGRAM
NO
YES
RU F2
RU K2
TPTYPE1
B5
IS SYSLST A DISK
YES
NO
7
C5
STORE DATA ADDR OF TPLINE INTO CCWTP VIA REG1
D5
STORE DATA ADDR OF TPLN INTO CCWTAP VIA REG1 LOAD ADDR OF CCWTP IN REG1
3
RU B2
7
DKTYPE
G5
LOAD ADDR OF DIB INTO REG 5 INDEX 72 BYTES TO POINT TO SYSLST DIB
H5
COMPARE CURRENT ADDR WITH END ADDR IN SYSLST DIB FOR EXTENT FULL CONDITION
J5
SYSLST EXTENT FILLED
YES
NO
4
K5
STORE CURRENT ADDR FROM SYSLST DIB INTO SEEK CCWDK
RU B1

Chart RU. $$BPCHK - Set Up for I/O and Output the Message
Refer to Chart 17.

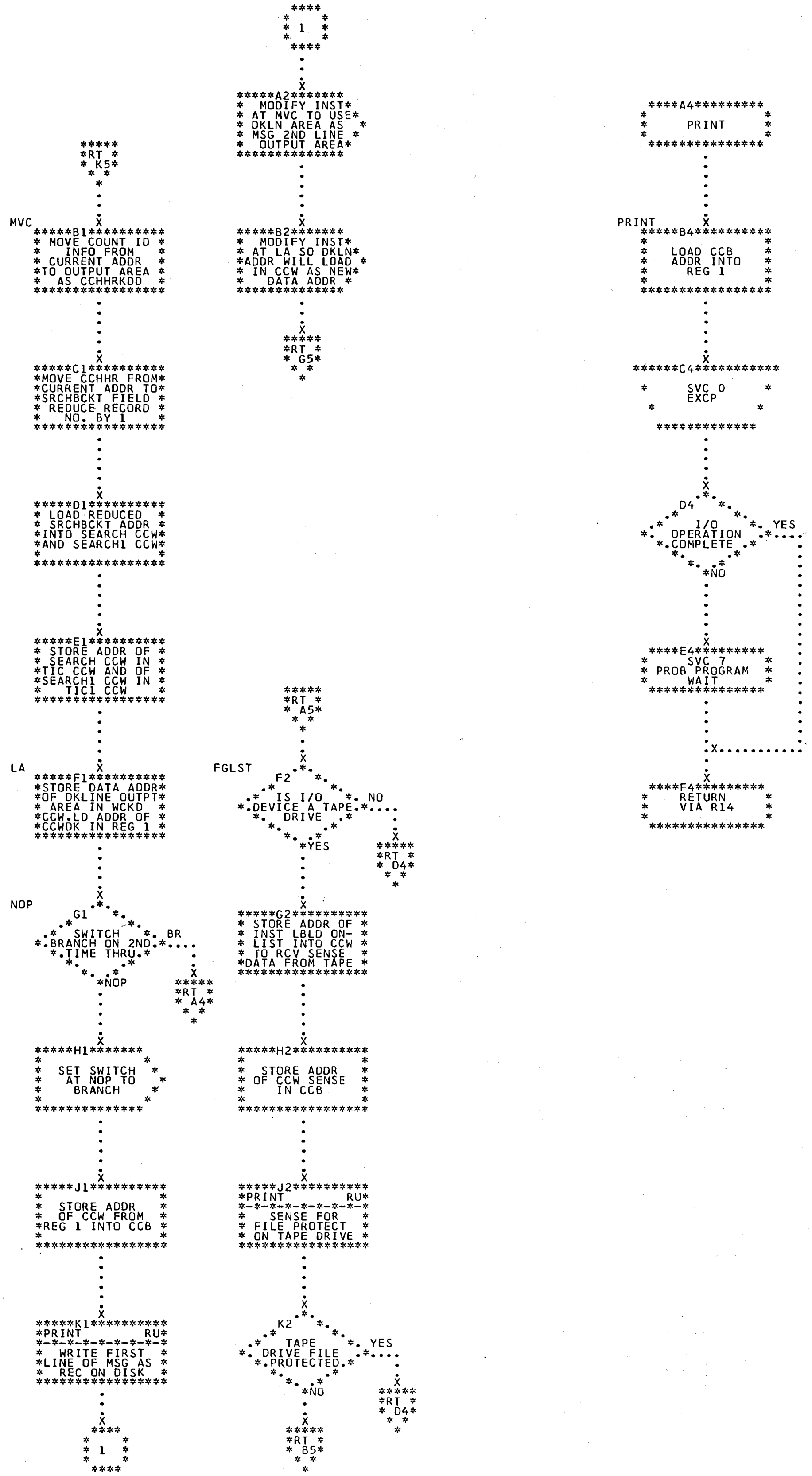
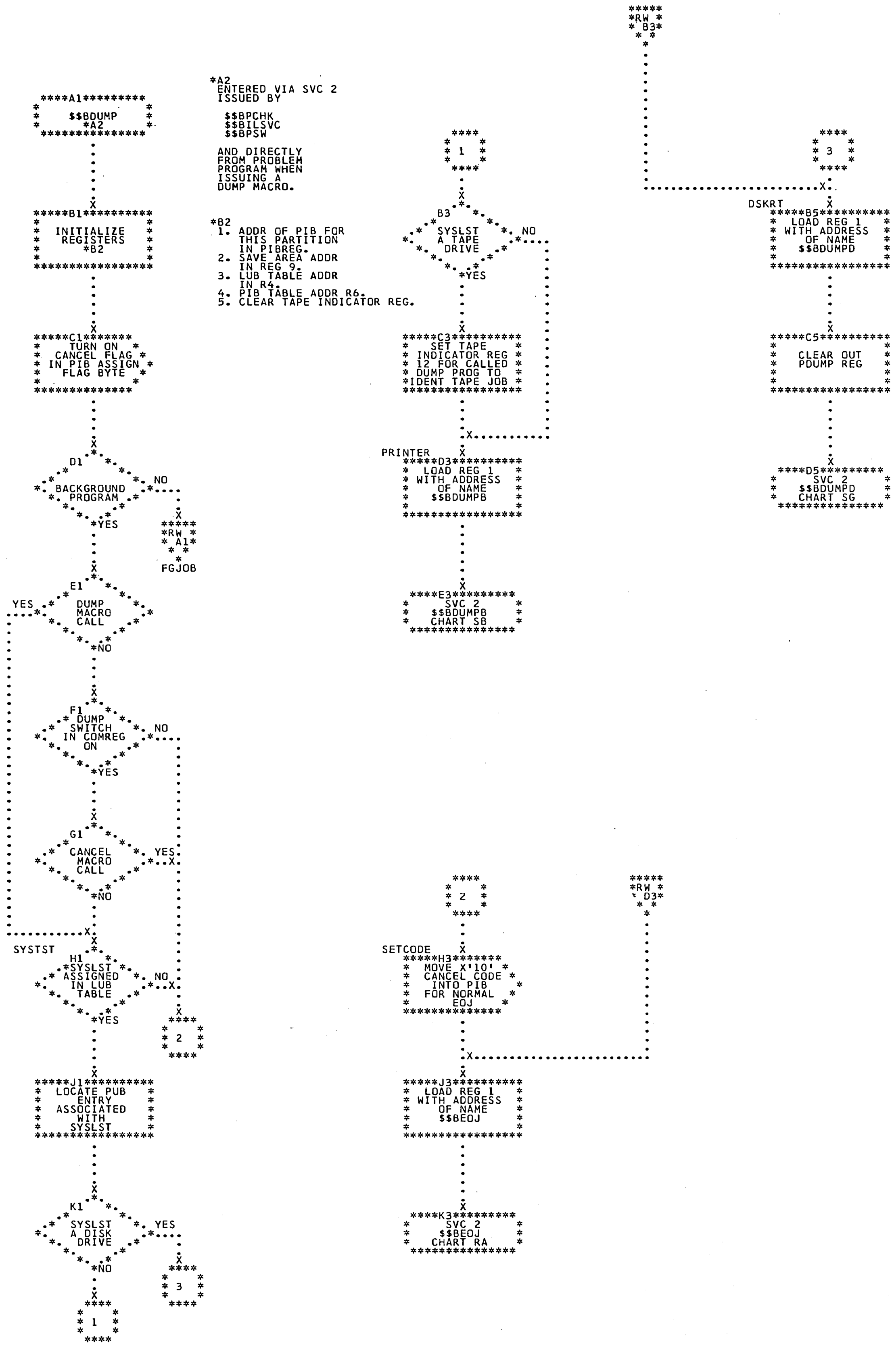

1
A2
MODIFY INST AT MVC TO USE DKLN AREA AS MSG 2ND LINE OUTPUT AREA
RT K5
MVC
B1
MOVE COUNT ID INFO FROM CURRENT ADDR TO OUTPUT AREA AS CCHHRKDD
B2
MODIFY INST AT LA SO DKLN ADDR WILL LOAD IN CCW AS NEW DATA ADDR
RT G5
C1
MOVE CCHHR FROM CURRENT ADDR TO SRCHBCKT FIELD REDUCE RECORD NO. BY 1
D1
LOAD REDUCED SRCHBCKT ADDR INTO SEARCH CCW AND SEARCH1 CCW
E1
STORE ADDR OF SEARCH CCW IN TIC CCW AND OF SEARCH1 CCW IN TIC1 CCW
LA
F1
STORE DATA ADDR OF DKLINE OUTPT AREA IN WCKD CCW.LD ADDR OF CCWDK IN REG 1
NOP
G1
SWITCH BRANCH ON 2ND TIME THRU
BR
RT A4
NOP
H1
SET SWITCH AT NOP TO BRANCH
J1
STORE ADDR OF CCW FROM REG 1 INTO CCB
K1
PRINT RU
WRITE FIRST LINE OF MSG AS REC ON DISK
1
RT A5
FGLST
F2
IS I/O DEVICE A TAPE DRIVE
NO
RT D4
YES
G2
STORE ADDR OF INST LBLD ON-LIST INTO CCW TO RCV SENSE DATA FROM TAPE
H2
STORE ADDR OF CCW SENSE IN CCB
J2
PRINT RU
SENSE FOR FILE PROTECT ON TAPE DRIVE
K2
TAPE DRIVE FILE PROTECTED
YES
RT D4
NO
RT B5
A4
PRINT
PRINT
B4
LOAD CCB ADDR INTO REG 1
C4
SVC 0 EXCP
D4
I/O OPERATION COMPLETE
YES
NO
E4
SVC 7 PROB PROGRAM WAIT
F4
RETURN VIA R14

Chart RV. $$BDUMP - Monitor Background Program Dump
Refer to Chart 16.


A1 $$BDUMP *A2
B1 INITIALIZE REGISTERS *B2
C1 TURN ON CANCEL FLAG IN PIB ASSIGN FLAG BYTE
D1 BACKGROUND PROGRAM
NO
YES
RW A1
FGJOB
E1 DUMP MACRO CALL
YES
NO
F1 DUMP SWITCH IN COMREG ON
NO
YES
G1 CANCEL MACRO CALL
YES
NO
SYSTST
H1 SYSLST ASSIGNED IN LUB TABLE
NO
YES
2
J1 LOCATE PUB ENTRY ASSOCIATED WITH SYSLST
K1 SYSLST A DISK DRIVE
YES
NO
3
1
1
B3 SYSLST A TAPE DRIVE
NO
YES
C3 SET TAPE INDICATOR REG 12 FOR CALLED DUMP PROG TO IDENT TAPE JOB
PRINTER
D3 LOAD REG 1 WITH ADDRESS OF NAME $$BDUMPB
E3 SVC 2 $$BDUMPB CHART SB
2
SETCODE
H3 MOVE X'10' CANCEL CODE INTO PIB FOR NORMAL EOJ
J3 LOAD REG 1 WITH ADDRESS OF NAME $$BEOJ
K3 SVC 2 $$BEOJ CHART RA
RW B3
3
DSKRT
B5 LOAD REG 1 WITH ADDRESS OF NAME $$BDUMPD
C5 CLEAR OUT PDUMP REG
D5 SVC 2 $$BDUMPD CHART SG
RW D3

*A2
ENTERED VIA SVC 2
ISSUED BY

$$BPCHK
$$BILSVC
$$BPSW

AND DIRECTLY
FROM PROBLEM
PROGRAM WHEN
ISSUING A
DUMP MACRO.

*B2
1. ADDR OF PIB FOR THIS PARTITION IN PIBREG.
2. SAVE AREA ADDR IN REG 9.
3. LUB TABLE ADDR IN R4.
4. PIB TABLE ADDR R6.
5. CLEAR TAPE INDICATOR REG.

Chart RW. $$BDUMP - Monitor Foreground Program Dump
Refer to Chart 16.

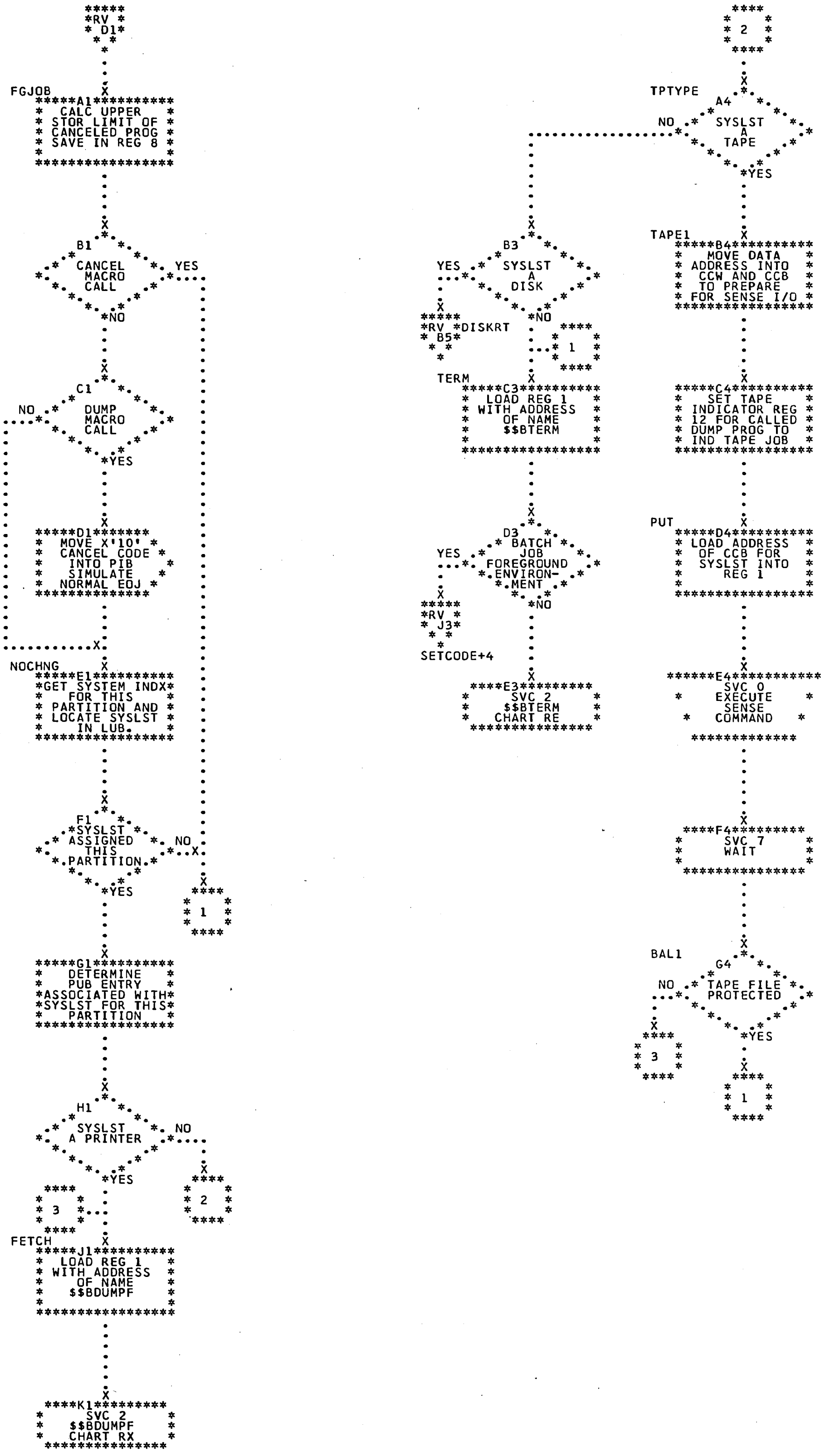

RV D1
FGJOB
A1
CALC UPPER STOR LIMIT OF CANCELED PROG SAVE IN REG 8
B1
CANCEL MACRO CALL
YES
NO
C1
DUMP MACRO CALL
NO
YES
D1
MOVE X'10' CANCEL CODE INTO PIB SIMULATE NORMAL EOJ
NOCHNG
E1
GET SYSTEM INDX FOR THIS PARTITION AND LOCATE SYSLST IN LUB.
F1
SYSLST ASSIGNED THIS PARTITION
NO
YES
1
G1
DETERMINE PUB ENTRY ASSOCIATED WITH SYSLST FOR THIS PARTITION
H1
SYSLST A PRINTER
NO
YES
3
2
FETCH
J1
LOAD REG 1 WITH ADDRESS OF NAME $$BDUMPF
K1
SVC 2 $$BDUMPF CHART RX
B3
SYSLST A DISK
YES
NO
RV B5 DISKRT
1
TERM
C3
LOAD REG 1 WITH ADDRESS OF NAME $$BTERM
D3
BATCH JOB FOREGROUND ENVIRON-MENT
YES
NO
RV J3
SETCODE+4
E3
SVC 2 $$BTERM CHART RE
2
TPTYPE
A4
SYSLST A TAPE
NO
YES
TAPE1
B4
MOVE DATA ADDRESS INTO CCW AND CCB TO PREPARE FOR SENSE I/O
C4
SET TAPE INDICATOR REG 12 FOR CALLED DUMP PROG TO IND TAPE JOB
PUT
D4
LOAD ADDRESS OF CCB FOR SYSLST INTO REG 1
E4
SVC 0 EXECUTE SENSE COMMAND
F4
SVC 7 WAIT
BAL1
G4
TAPE FILE PROTECTED
NO
YES
3
1

Chart RX. $$BDUMPF - Foreground Program Dump
Refer to Chart 18.

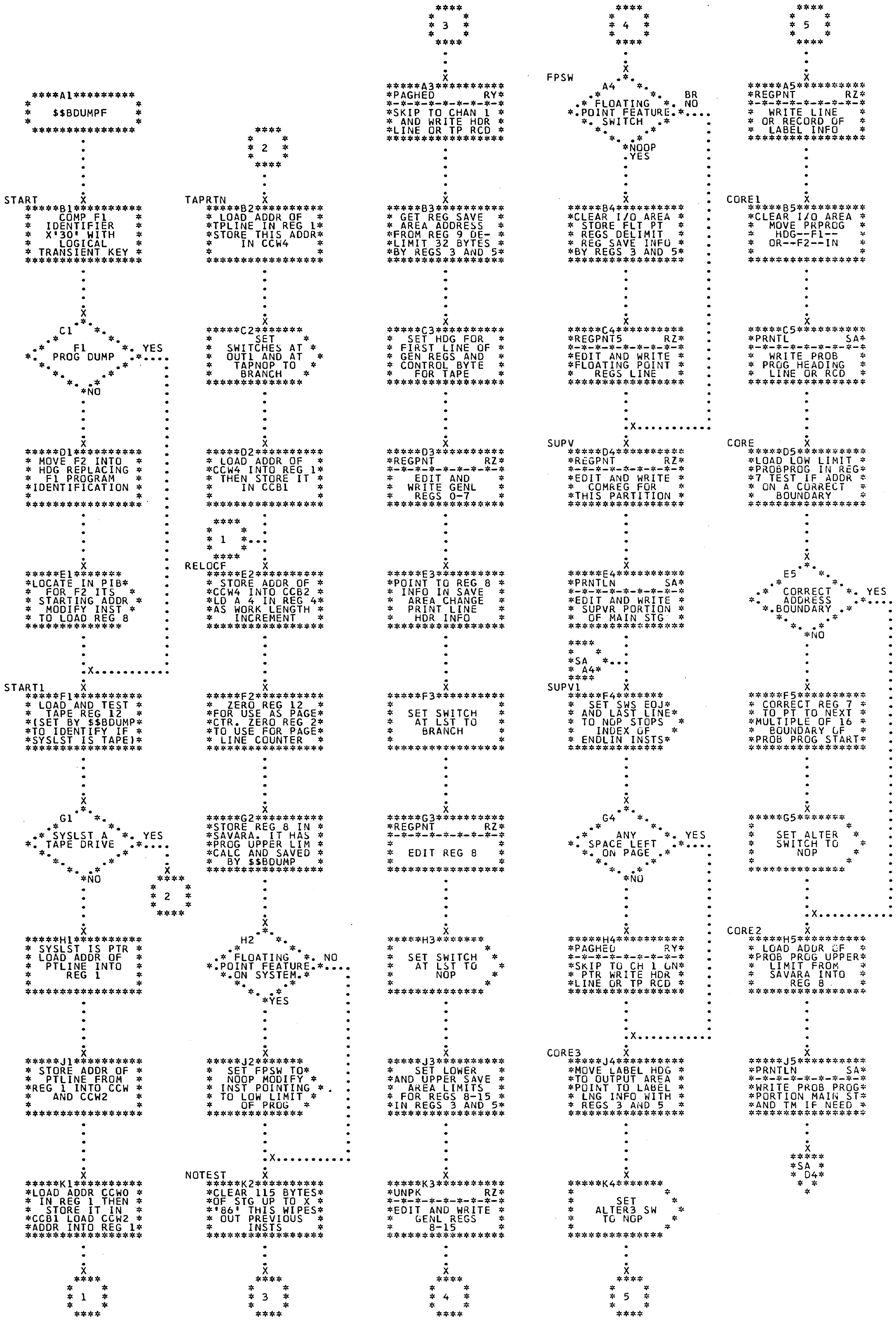

A1
$$BDUMPF
START
B1
COMP F1 IDENTIFIER X'30' WITH LOGICAL TRANSIENT KEY
C1
F1 PROG DUMP
YES
NO
D1
MOVE F2 INTO HDG REPLACING F1 PROGRAM IDENTIFICATION
E1
LOCATE IN PIB FOR F2 ITS STARTING ADDR MODIFY INST TO LOAD REG 8
START1
F1
LOAD AND TEST TAPE REG 12 (SET BY $$BDUMP TO IDENTIFY IF SYSLST IS TAPE)
G1
SYSLST A TAPE DRIVE
YES
NO
2
H1
SYSLST IS PTR LOAD ADDR OF PTLINE INTO REG 1
J1
STORE ADDR OF PTLINE FROM REG 1 INTO CCW AND CCW2
K1
LOAD ADDR CCWO IN REG 1 THEN STORE IT IN CCB1 LOAD CCW2 ADDR INTO REG 1
1
2
TAPRTN
B2
LOAD ADDR OF TPLINE IN REG 1 STORE THIS ADDR IN CCW4
C2
SET SWITCHES AT OUT1 AND AT TAPNOP TO BRANCH
D2
LOAD ADDR OF CCW4 INTO REG 1 THEN STORE IT IN CCB1
1
RELOCF
E2
STORE ADDR OF CCW4 INTO CCB2 LD A 4 IN REG 4 AS WORK LENGTH INCREMENT
F2
ZERO REG 12 FOR USE AS PAGE CTR. ZERO REG 2 TO USE FOR PAGE LINE COUNTER
G2
STORE REG 8 IN SAVARA. IT HAS PROG UPPER LIM CALC AND SAVED BY $$BDUMP
H2
FLOATING POINT FEATURE. ON SYSTEM.
NO
YES
J2
SET FPSW TO NOOP MODIFY INST POINTING TO LOW LIMIT OF PROG
NOTEST
K2
CLEAR 115 BYTES OF STG UP TO X '86' THIS WIPES OUT PREVIOUS INSTS
3
3
A3
PAGHED RY
SKIP TO CHAN 1 AND WRITE HDR LINE OR TP RCD
B3
GET REG SAVE AREA ADDRESS FROM REG 9 DE- LIMIT 32 BYTES BY REGS 3 AND 5
C3
SET HDG FOR FIRST LINE OF GEN REGS AND CONTROL BYTE FOR TAPE
D3
REGPNT RZ
EDIT AND WRITE GENL REGS 0-7
E3
POINT TO REG 8 INFO IN SAVE AREA CHANGE PRINT LINE HDR INFO
F3
SET SWITCH AT LST TO BRANCH
G3
REGPNT RZ
EDIT REG 8
H3
SET SWITCH AT LST TO NOP
J3
SET LOWER AND UPPER SAVE AREA LIMITS FOR REGS 8-15 IN REGS 3 AND 5
K3
UNPK RZ
EDIT AND WRITE GENL REGS 8-15
4
4
FPSW
A4
FLOATING POINT FEATURE. SWITCH
BR NO
NOOP YES
B4
CLEAR I/O AREA STORE FLT PT REGS DELIMIT REG SAVE INFO BY REGS 3 AND 5
C4
REGPNT5 RZ
EDIT AND WRITE FLOATING POINT REGS LINE
SUPV
D4
REGPNT RZ
EDIT AND WRITE COMREG FOR THIS PARTITION
E4
PRNTLN SA
EDIT AND WRITE SUPVR PORTION OF MAIN STG
SA A4
SUPV1
F4
SET SWS EOJ AND LAST LINE TO NOP STOPS INDEX OF ENDLIN INSTS
G4
ANY SPACE LEFT ON PAGE
YES
NO
H4
PAGHED RY
SKIP TO CH 1 ON PTR WRITE HDR LINE OR TP RCD
CORE3
J4
MOVE LABEL HDG TO OUTPUT AREA POINT TO LABEL LNG INFO WITH REGS 3 AND 5
K4
SET ALTER3 SW TO NOP
5
5
A5
REGPNT RZ
WRITE LINE OR RECORD OF LABEL INFO
CORE1
B5
CLEAR I/O AREA MOVE PRPROG HDG--F1-- OR--F2--IN
C5
PRNTL SA
WRITE PROB PROG HEADING LINE OR RCD
CORE
D5
LOAD LOW LIMIT PROBPROG IN REG 7 TEST IF ADDR ON A CORRECT BOUNDARY
E5
CORRECT ADDRESS BOUNDARY
YES
NO
F5
CORRECT REG 7 TO PT TO NEXT MULTIPLE OF 16 BOUNDARY OF PROB PROG START
G5
SET ALTER SWITCH TO NOP
CORE2
H5
LOAD ADDR OF PROB PROG UPPER LIMIT FROM SAVARA INTO REG 8
J5
PRNTLN SA
WRITE PROB PROG PORTION MAIN ST AND TM IF NEED
SA D4

Chart RY. $$BDUMPF - Prepare Page Headings and PIOCS Subroutines
Refer to Chart 18.

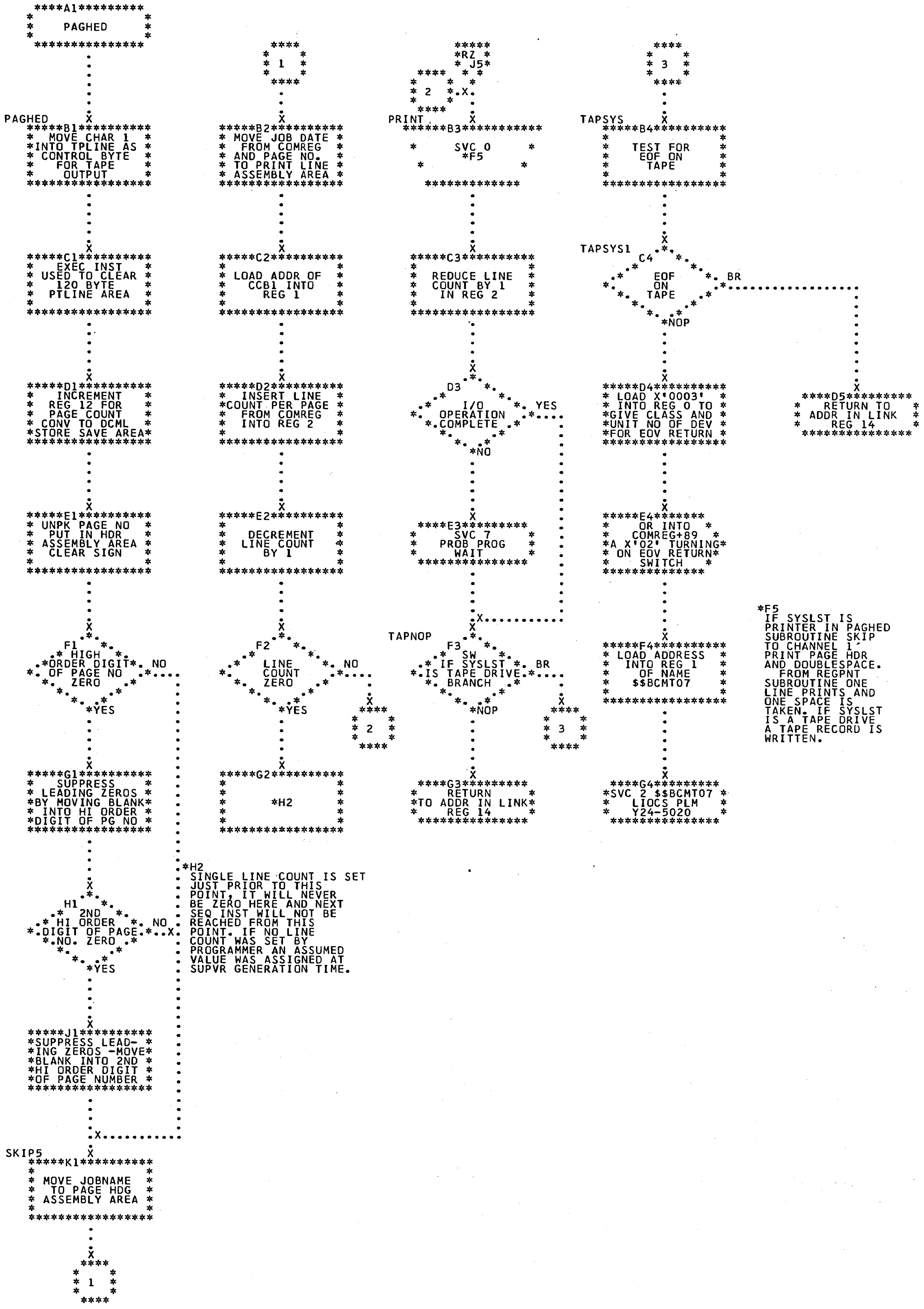

A1 PAGHED
PAGHED
B1 MOVE CHAR 1 INTO TPLINE AS CONTROL BYTE FOR TAPE OUTPUT
C1 EXEC INST USED TO CLEAR 120 BYTE PTLINE AREA
D1 INCREMENT REG 12 FOR PAGE COUNT CONV TO DCML STORE SAVE AREA
E1 UNPK PAGE NO PUT IN HDR ASSEMBLY AREA CLEAR SIGN
F1 HIGH ORDER DIGIT OF PAGE NO ZERO
NO
YES
G1 SUPPRESS LEADING ZEROS BY MOVING BLANK INTO HI ORDER DIGIT OF PG NO
H1 2ND HI ORDER DIGIT OF PAGE. NO. ZERO
NO
YES
J1 SUPPRESS LEADING ZEROS -MOVE BLANK INTO 2ND HI ORDER DIGIT OF PAGE NUMBER
SKIP5
K1 MOVE JOBNAME TO PAGE HDG ASSEMBLY AREA
1
1
B2 MOVE JOB DATE FROM COMREG AND PAGE NO. TO PRINT LINE ASSEMBLY AREA
C2 LOAD ADDR OF CCB1 INTO REG 1
D2 INSERT LINE COUNT PER PAGE FROM COMREG INTO REG 2
E2 DECREMENT LINE COUNT BY 1
F2 LINE COUNT ZERO
NO
2
YES
G2 *H2
*H2
SINGLE LINE COUNT IS SET JUST PRIOR TO THIS POINT, IT WILL NEVER BE ZERO HERE AND NEXT SEQ INST WILL NOT BE REACHED FROM THIS POINT. IF NO LINE COUNT WAS SET BY PROGRAMMER AN ASSUMED VALUE WAS ASSIGNED AT SUPVR GENERATION TIME.
RZ J5
2
PRINT
B3 SVC 0 *F5
C3 REDUCE LINE COUNT BY 1 IN REG 2
D3 I/O OPERATION COMPLETE
YES
NO
E3 SVC 7 PROB PROG WAIT
TAPNOP
F3 SW IF SYSLST IS TAPE DRIVE. BRANCH
BR
NOP
3
G3 RETURN TO ADDR IN LINK REG 14
3
TAPSYS
B4 TEST FOR EOF ON TAPE
TAPSYS1
C4 EOF ON TAPE
BR
NOP
D4 LOAD X'0003' INTO REG 0 TO GIVE CLASS AND UNIT NO OF DEV FOR EOV RETURN
E4 OR INTO COMREG+89 A X'02' TURNING ON EOV RETURN SWITCH
F4 LOAD ADDRESS INTO REG 1 OF NAME $$BCMT07
G4 SVC 2 $$BCMT07 LIOCS PLM Y24-5020
D5 RETURN TO ADDR IN LINK REG 14
*F5
IF SYSLST IS PRINTER IN PAGHED SUBROUTINE SKIP TO CHANNEL 1 PRINT PAGE HDR AND DOUBLESPACE. FROM REGPNT SUBROUTINE ONE LINE PRINTS AND ONE SPACE IS TAKEN. IF SYSLST IS A TAPE DRIVE A TAPE RECORD IS WRITTEN.

Chart RZ. $$BDUMPF - Prepare and Edit a Line Subroutine
Refer to Chart 18.

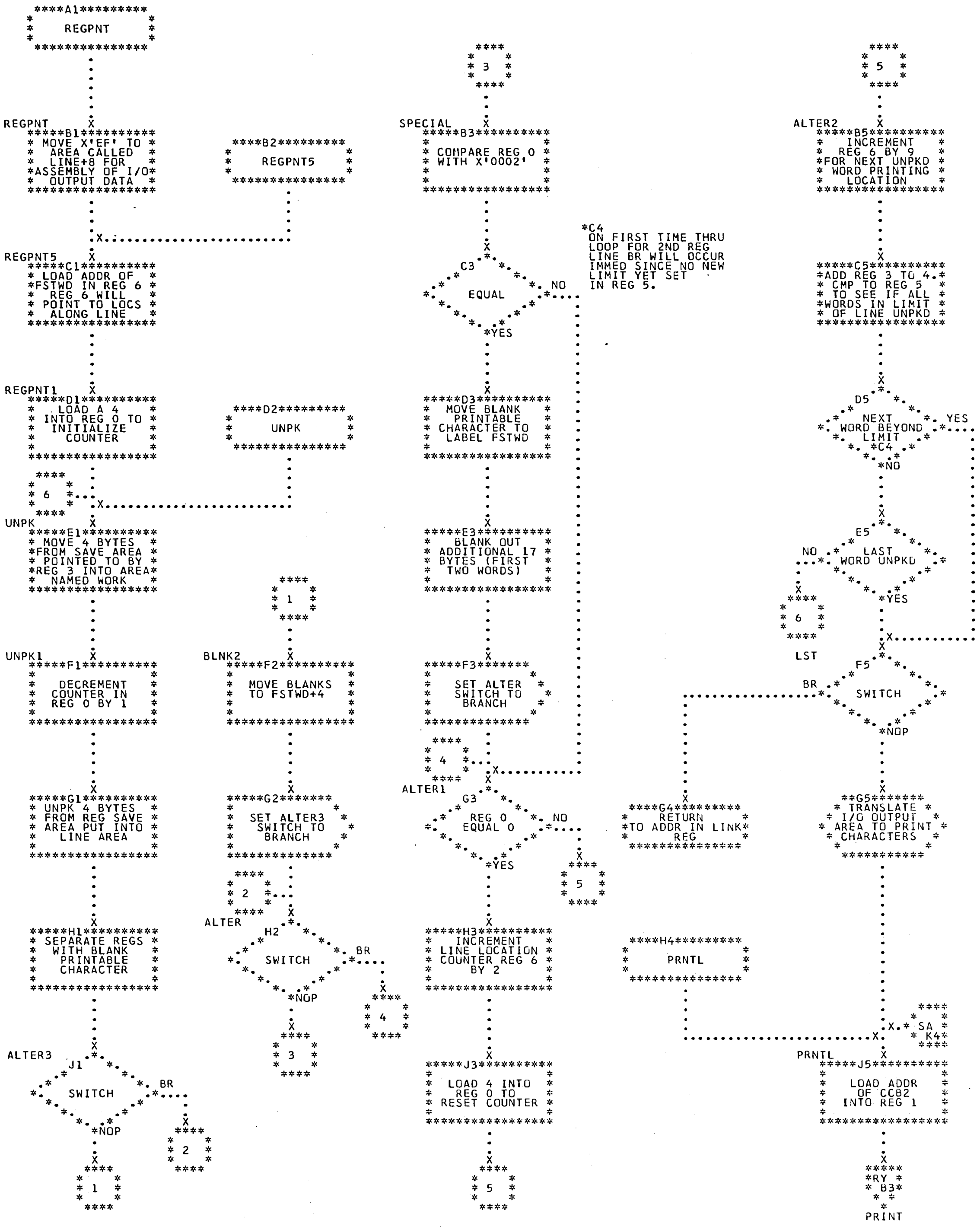

A1
REGPNT
REGPNT
B1
MOVE X'EF' TO AREA CALLED LINE+8 FOR ASSEMBLY OF I/O OUTPUT DATA
B2
REGPNT5
REGPNT5
C1
LOAD ADDR OF FSTWD IN REG 6 REG 6 WILL POINT TO LOCS ALONG LINE
REGPNT1
D1
LOAD A 4 INTO REG 0 TO INITIALIZE COUNTER
D2
UNPK
6
UNPK
E1
MOVE 4 BYTES FROM SAVE AREA POINTED TO BY REG 3 INTO AREA NAMED WORK
1
UNPK1
F1
DECREMENT COUNTER IN REG 0 BY 1
BLNK2
F2
MOVE BLANKS TO FSTWD+4
G1
UNPK 4 BYTES FROM REG SAVE AREA PUT INTO LINE AREA
G2
SET ALTER3 SWITCH TO BRANCH
2
ALTER
H1
SEPARATE REGS WITH BLANK PRINTABLE CHARACTER
H2
SWITCH
BR
NOP
4
3
ALTER3
J1
SWITCH
BR
NOP
2
1
3
SPECIAL
B3
COMPARE REG 0 WITH X'0002'
C3
EQUAL
NO
YES
C4
ON FIRST TIME THRU LOOP FOR 2ND REG LINE BR WILL OCCUR IMMED SINCE NO NEW LIMIT YET SET IN REG 5.
D3
MOVE BLANK PRINTABLE CHARACTER TO LABEL FSTWD
E3
BLANK OUT ADDITIONAL 17 BYTES (FIRST TWO WORDS)
F3
SET ALTER SWITCH TO BRANCH
4
ALTER1
G3
REG 0 EQUAL 0
NO
YES
5
H3
INCREMENT LINE LOCATION COUNTER REG 6 BY 2
J3
LOAD 4 INTO REG 0 TO RESET COUNTER
5
G4
RETURN TO ADDR IN LINK REG
H4
PRNTL
5
ALTER2
B5
INCREMENT REG 6 BY 9 FOR NEXT UNPKD WORD PRINTING LOCATION
C5
ADD REG 3 TO 4. CMP TO REG 5 TO SEE IF ALL WORDS IN LIMIT OF LINE UNPKD
D5
NEXT WORD BEYOND LIMIT
C4
YES
NO
E5
LAST WORD UNPKD
NO
YES
6
LST
F5
SWITCH
BR
NOP
G5
TRANSLATE I/O OUTPUT AREA TO PRINT CHARACTERS
SA
K4
PRNTL
J5
LOAD ADDR OF CCB2 INTO REG 1
RY
B3
PRINT

Chart SA. $$BDUMPF - Line Test Subroutines
Refer to Chart 18.

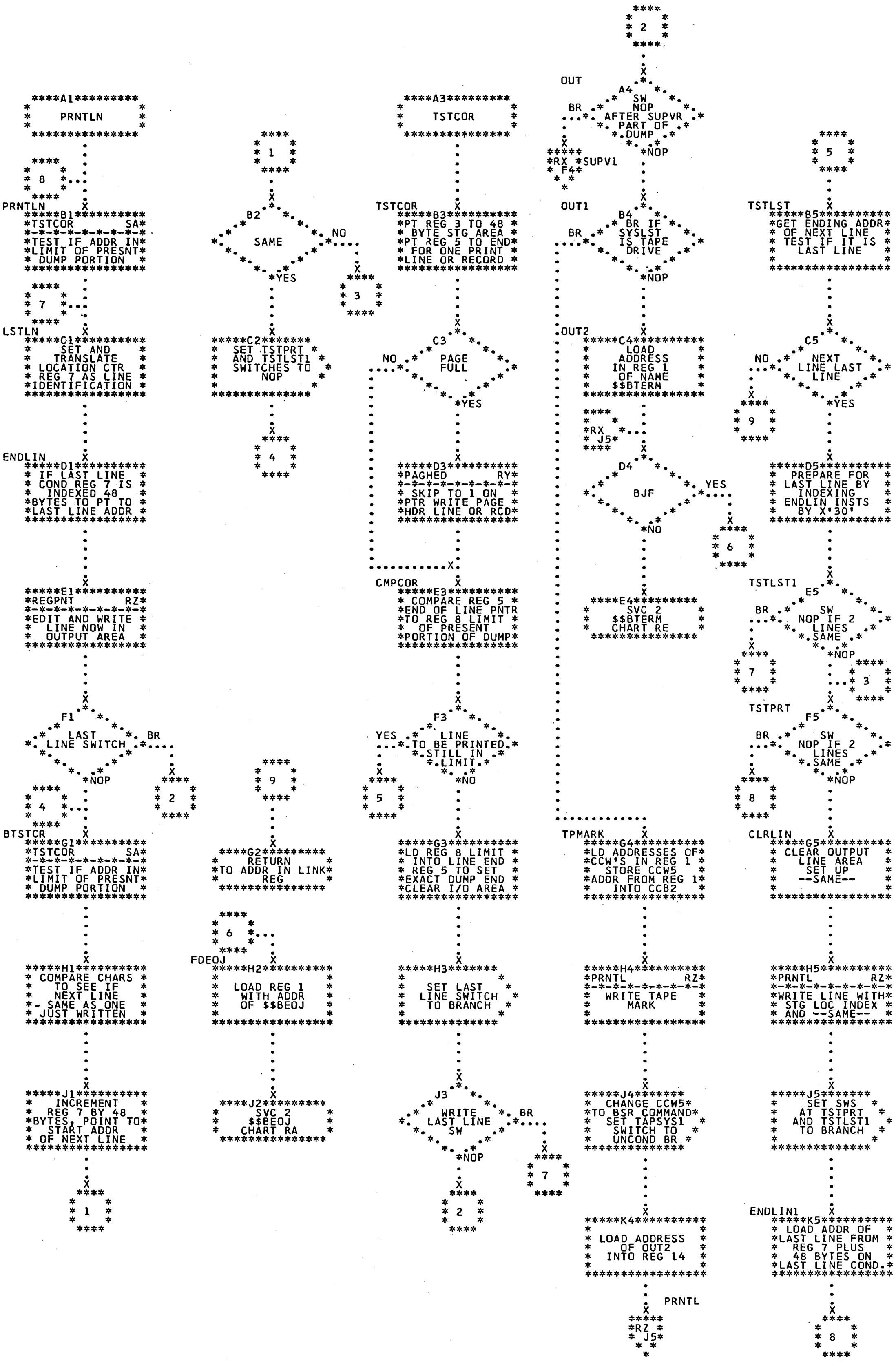

A1
PRNTLN
8
PRNTLN
B1
TSTCOR SA
TEST IF ADDR IN LIMIT OF PRESNT DUMP PORTION
7
LSTLN
C1
SET AND TRANSLATE LOCATION CTR REG 7 AS LINE IDENTIFICATION
ENDLIN
D1
IF LAST LINE COND REG 7 IS INDEXED 48 BYTES TO PT TO LAST LINE ADDR
E1
REGPNT RZ
EDIT AND WRITE LINE NOW IN OUTPUT AREA
F1
LAST LINE SWITCH
BR
NOP
2
4
BTSTCR
G1
TSTCOR SA
TEST IF ADDR IN LIMIT OF PRESNT DUMP PORTION
H1
COMPARE CHARS TO SEE IF NEXT LINE - SAME AS ONE JUST WRITTEN
J1
INCREMENT REG 7 BY 48 BYTES, POINT TO START ADDR OF NEXT LINE
1
1
B2
SAME
NO
YES
3
C2
SET TSTPRT AND TSTLST1 SWITCHES TO NOP
4
9
G2
RETURN TO ADDR IN LINK REG
6
FDEOJ
H2
LOAD REG 1 WITH ADDR OF $$BEOJ
J2
SVC 2 $$BEOJ CHART RA
A3
TSTCOR
TSTCOR
B3
PT REG 3 TO 48 BYTE STG AREA PT REG 5 TO END FOR ONE PRINT LINE OR RECORD
C3
PAGE FULL
NO
YES
D3
PAGHED RY
SKIP TO 1 ON PTR WRITE PAGE HDR LINE OR RCD
CMPCOR
E3
COMPARE REG 5 END OF LINE PNTR TO REG 8 LIMIT OF PRESENT PORTION OF DUMP
F3
LINE TO BE PRINTED STILL IN LIMIT
YES
NO
5
G3
LD REG 8 LIMIT INTO LINE END REG 5 TO SET EXACT DUMP END CLEAR I/O AREA
H3
SET LAST LINE SWITCH TO BRANCH
J3
WRITE LAST LINE SW
BR
NOP
7
2
2
OUT
A4
SW NOP AFTER SUPVR PART OF DUMP
BR
NOP
RX F4
SUPV1
OUT1
B4
BR IF SYSLST IS TAPE DRIVE
BR
NOP
OUT2
C4
LOAD ADDRESS IN REG 1 OF NAME $$BTERM
RX J5
D4
BJF
YES
NO
6
E4
SVC 2 $$BTERM CHART RE
TPMARK
G4
LD ADDRESSES OF CCW'S IN REG 1 STORE CCW5 ADDR FROM REG 1 INTO CCB2
H4
PRNTL RZ
WRITE TAPE MARK
J4
CHANGE CCW5 TO BSR COMMAND SET TAPSYS1 SWITCH TO UNCOND BR
K4
LOAD ADDRESS OF OUT2 INTO REG 14
PRNTL
RZ J5
5
TSTLST
B5
GET ENDING ADDR OF NEXT LINE TEST IF IT IS LAST LINE
C5
NEXT LINE LAST LINE
NO
YES
9
D5
PREPARE FOR LAST LINE BY INDEXING ENDLIN INSTS BY X'30'
TSTLST1
E5
SW NOP IF 2 LINES SAME
BR
NOP
7
3
TSTPRT
F5
SW NOP IF 2 LINES SAME
BR
NOP
8
CLRLIN
G5
CLEAR OUTPUT LINE AREA SET UP --SAME--
H5
PRNTL RZ
WRITE LINE WITH STG LOC INDEX AND --SAME--
J5
SET SWS AT TSTPRT AND TSTLST1 TO BRANCH
ENDLIN1
K5
LOAD ADDR OF LAST LINE FROM REG 7 PLUS 48 BYTES ON LAST LINE COND.
8

Chart SB. $$BDUMPB - Initialization for BG Storage Dump on Printer or Tape
Refer to Chart 18.

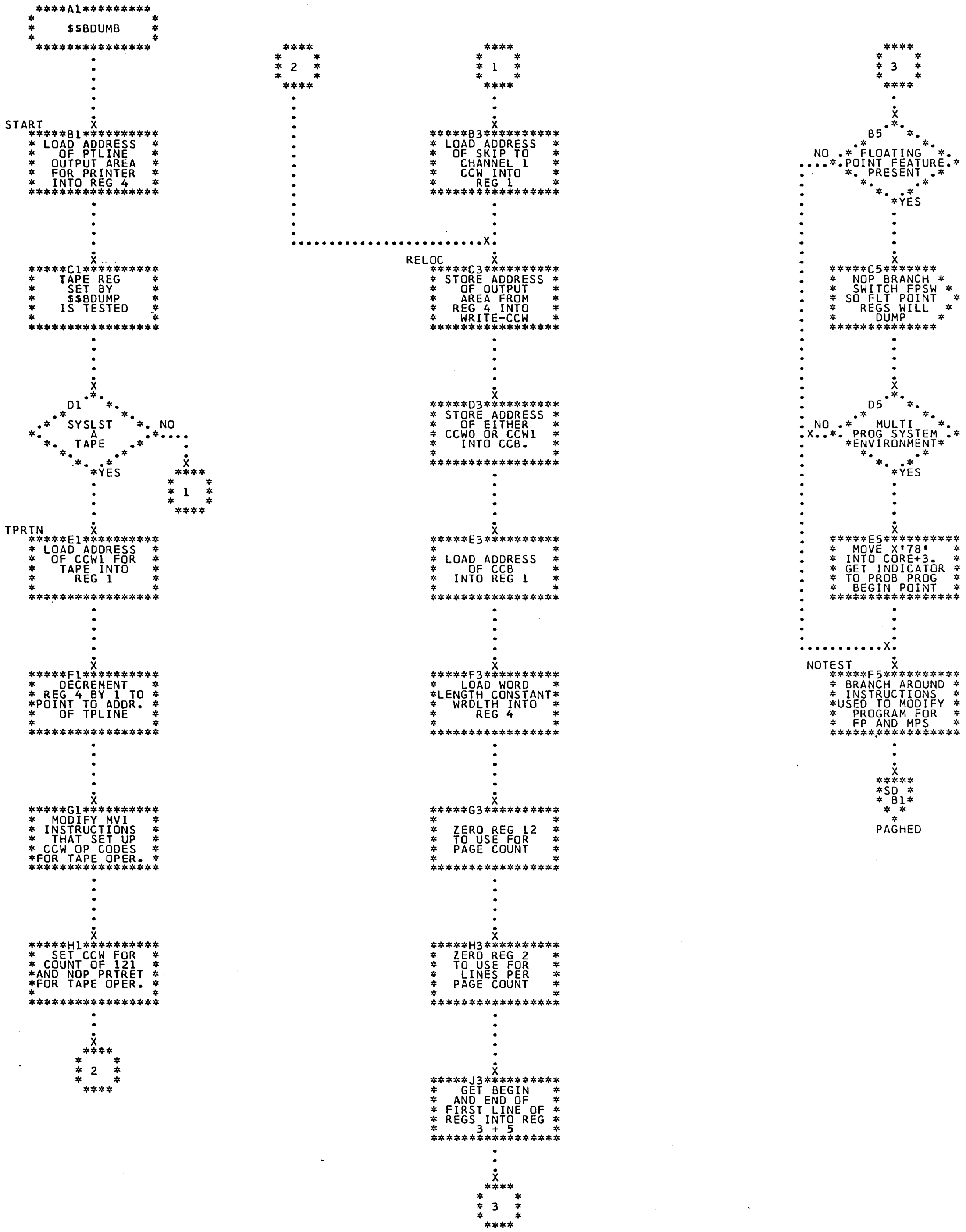

A1
$$BDUMB
START
B1
LOAD ADDRESS OF PTLINE OUTPUT AREA FOR PRINTER INTO REG 4
C1
TAPE REG SET BY $$BDUMP IS TESTED
D1
SYSLST A TAPE
NO
YES
1
TPRTN
E1
LOAD ADDRESS OF CCW1 FOR TAPE INTO REG 1
F1
DECREMENT REG 4 BY 1 TO POINT TO ADDR. OF TPLINE
G1
MODIFY MVI INSTRUCTIONS THAT SET UP CCW OP CODES FOR TAPE OPER.
H1
SET CCW FOR COUNT OF 121 AND NOP PRTRET FOR TAPE OPER.
2
2
1
B3
LOAD ADDRESS OF SKIP TO CHANNEL 1 CCW INTO REG 1
RELOC
C3
STORE ADDRESS OF OUTPUT AREA FROM REG 4 INTO WRITE-CCW
D3
STORE ADDRESS OF EITHER CCW0 OR CCW1 INTO CCB.
E3
LOAD ADDRESS OF CCB INTO REG 1
F3
LOAD WORD LENGTH CONSTANT WRDLTH INTO REG 4
G3
ZERO REG 12 TO USE FOR PAGE COUNT
H3
ZERO REG 2 TO USE FOR LINES PER PAGE COUNT
J3
GET BEGIN AND END OF FIRST LINE OF REGS INTO REG 3 + 5
3
3
B5
FLOATING POINT FEATURE PRESENT
NO
YES
C5
NOP BRANCH SWITCH FPSW SO FLT POINT REGS WILL DUMP
D5
MULTI PROG SYSTEM ENVIRONMENT
NO
YES
E5
MOVE X'78' INTO CORE+3. GET INDICATOR TO PROB PROG BEGIN POINT
NOTEST
F5
BRANCH AROUND INSTRUCTIONS USED TO MODIFY PROGRAM FOR FP AND MPS
SD B1
PAGHED

Chart SC. $$BDUMPB - BG Dump on Printer or Tape Refer to Chart 18.

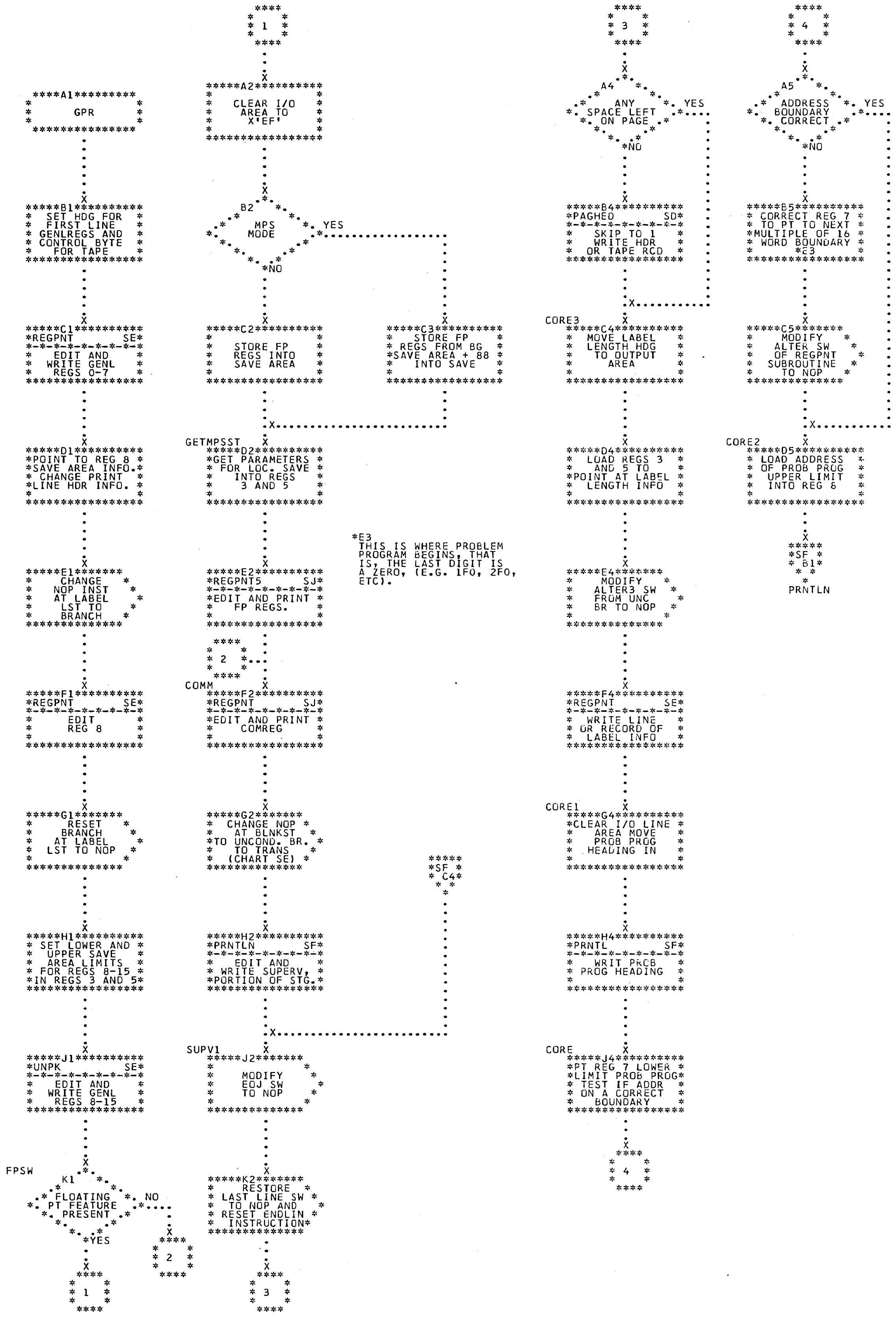
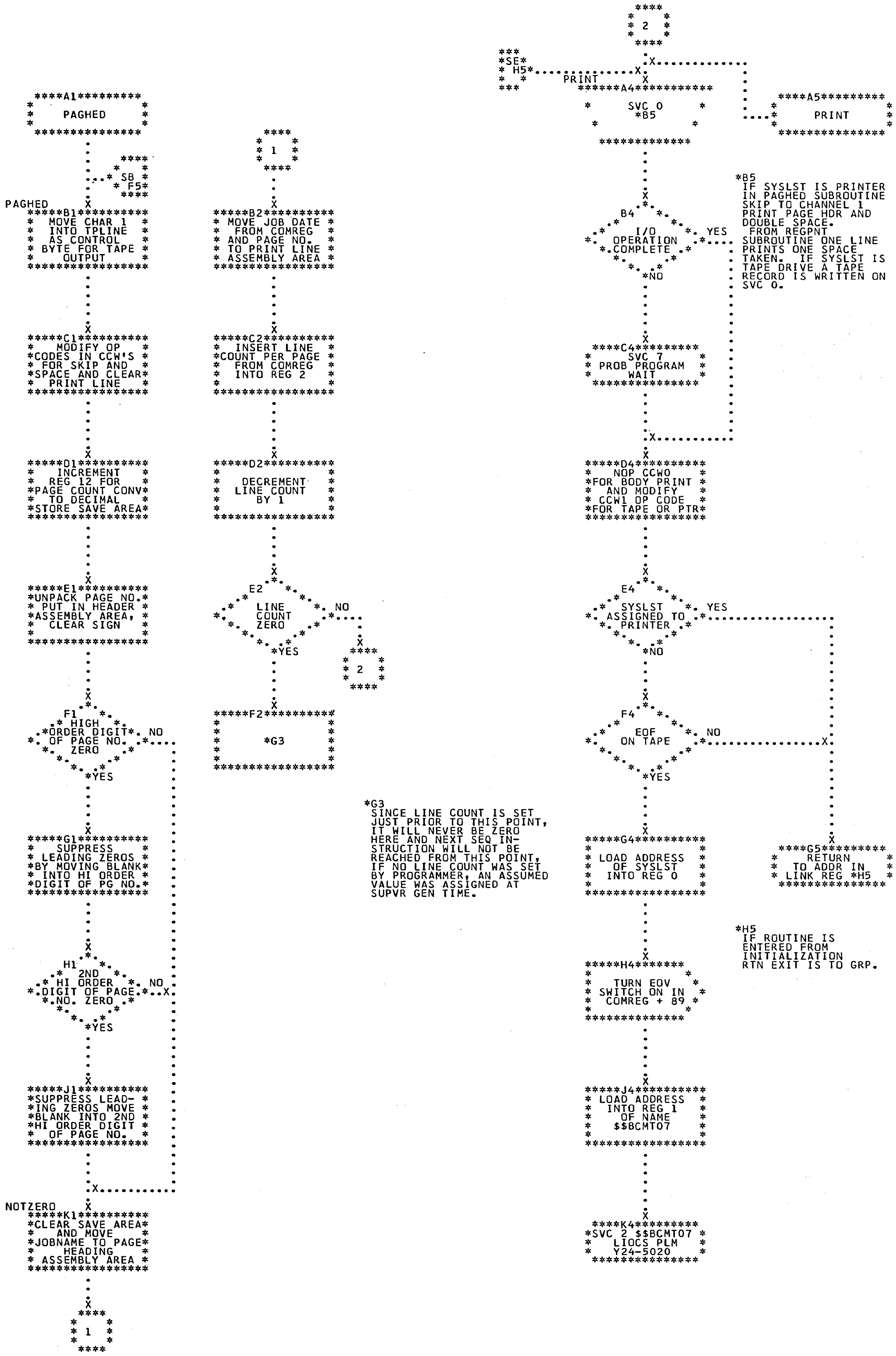

1
A1
GPR
A2
CLEAR I/O AREA TO X'EF'
B1
SET HDG FOR FIRST LINE GENLREGS AND CONTROL BYTE FOR TAPE
B2
MPS MODE
YES
NO
C1
REGPNT SE
EDIT AND WRITE GENL REGS 0-7
C2
STORE FP REGS INTO SAVE AREA
C3
STORE FP REGS FROM BG SAVE AREA + 88 INTO SAVE
D1
POINT TO REG 8 SAVE AREA INFO. CHANGE PRINT LINE HDR INFO.
GETMPSST
D2
GET PARAMETERS FOR LOC. SAVE INTO REGS 3 AND 5
*E3
THIS IS WHERE PROBLEM PROGRAM BEGINS, THAT IS, THE LAST DIGIT IS A ZERO, (E.G. 1FO, 2FO, ETC).
E1
CHANGE NOP INST AT LABEL LST TO BRANCH
E2
REGPNT5 SJ
EDIT AND PRINT FP REGS.
2
COMM
F1
REGPNT SE
EDIT REG 8
F2
REGPNT SJ
EDIT AND PRINT COMREG
G1
RESET BRANCH AT LABEL LST TO NOP
G2
CHANGE NOP AT BLNKST TO UNCOND. BR. TO TRANS (CHART SE)
SF C4
H1
SET LOWER AND UPPER SAVE AREA LIMITS FOR REGS 8-15 IN REGS 3 AND 5
H2
PRNTLN SF
EDIT AND WRITE SUPERV, PORTION OF STG.
SUPV1
J1
UNPK SE
EDIT AND WRITE GENL REGS 8-15
J2
MODIFY EOJ SW TO NOP
FPSW
K1
FLOATING PT FEATURE PRESENT
NO
YES
2
1
K2
RESTORE LAST LINE SW TO NOP AND RESET ENDLIN INSTRUCTION
3
3
A4
ANY SPACE LEFT ON PAGE
YES
NO
B4
PAGHED SD
SKIP TO 1 WRITE HDR OR TAPE RCD
CORE3
C4
MOVE LABEL LENGTH HDG TO OUTPUT AREA
D4
LOAD REGS 3 AND 5 TO POINT AT LABEL LENGTH INFO
E4
MODIFY ALTER3 SW FROM UNC BR TO NOP
F4
REGPNT SE
WRITE LINE OR RECORD OF LABEL INFO
CORE1
G4
CLEAR I/O LINE AREA MOVE PROB PROG HEADING IN
H4
PRNTL SF
WRIT PRCB PROG HEADING
CORE
J4
PT REG 7 LOWER LIMIT PROB PROG TEST IF ADDR ON A CORRECT BOUNDARY
4
4
A5
ADDRESS BOUNDARY CORRECT
YES
NO
B5
CORRECT REG 7 TO PT TO NEXT MULTIPLE OF 16 WORD BOUNDARY *E3
C5
MODIFY ALTER SW OF REGPNT SUBROUTINE TO NOP
CORE2
D5
LOAD ADDRESS OF PROB PROG UPPER LIMIT INTO REG 6
SF B1
PRNTLN

Chart SD. $$BDUMPB - Prepare Page Headings and PIOCS Subroutines
Refer to Chart 18.


A1 PAGHED
SB F5
PAGHED
B1 MOVE CHAR 1 INTO TPLINE AS CONTROL BYTE FOR TAPE OUTPUT
C1 MODIFY OP CODES IN CCW'S FOR SKIP AND SPACE AND CLEAR PRINT LINE
D1 INCREMENT REG 12 FOR PAGE COUNT CONV TO DECIMAL STORE SAVE AREA
E1 UNPACK PAGE NO. PUT IN HEADER ASSEMBLY AREA, CLEAR SIGN
F1 HIGH ORDER DIGIT OF PAGE NO. ZERO
NO
YES
G1 SUPPRESS LEADING ZEROS BY MOVING BLANK INTO HI ORDER DIGIT OF PG NO.
H1 2ND HI ORDER DIGIT OF PAGE NO. ZERO
NO
YES
J1 SUPPRESS LEADING ZEROS MOVE BLANK INTO 2ND HI ORDER DIGIT OF PAGE NO.
NOTZERO
K1 CLEAR SAVE AREA AND MOVE JOBNAME TO PAGE HEADING ASSEMBLY AREA
1
1
B2 MOVE JOB DATE FROM COMREG AND PAGE NO. TO PRINT LINE ASSEMBLY AREA
C2 INSERT LINE COUNT PER PAGE FROM COMREG INTO REG 2
D2 DECREMENT LINE COUNT BY 1
E2 LINE COUNT ZERO
NO
2
YES
F2 *G3
2
SE H5
PRINT
A4 SVC 0 *B5
A5 PRINT
B4 I/O OPERATION COMPLETE
YES
NO
C4 SVC 7 PROB PROGRAM WAIT
D4 NOP CCW0 FOR BODY PRINT AND MODIFY CCW1 OP CODE FOR TAPE OR PTR
E4 SYSLST ASSIGNED TO PRINTER
YES
NO
F4 EOF ON TAPE
NO
YES
G4 LOAD ADDRESS OF SYSLST INTO REG 0
G5 RETURN TO ADDR IN LINK REG *H5
H4 TURN EOV SWITCH ON IN COMREG + 89
J4 LOAD ADDRESS INTO REG 1 OF NAME $$BCMT07
K4 SVC 2 $$BCMT07 LIOCS PLM Y24-5020

*B5
IF SYSLST IS PRINTER IN PAGHED SUBROUTINE SKIP TO CHANNEL 1 PRINT PAGE HDR AND DOUBLE SPACE. FROM REGPNT SUBROUTINE ONE LINE PRINTS ONE SPACE TAKEN. IF SYSLST IS TAPE DRIVE A TAPE RECORD IS WRITTEN ON SVC 0.

*G3
SINCE LINE COUNT IS SET JUST PRIOR TO THIS POINT, IT WILL NEVER BE ZERO HERE AND NEXT SEQ INSTRUCTION WILL NOT BE REACHED FROM THIS POINT. IF NO LINE COUNT WAS SET BY PROGRAMMER, AN ASSUMED VALUE WAS ASSIGNED AT SUPVR GEN TIME.

*H5
IF ROUTINE IS ENTERED FROM INITIALIZATION RTN EXIT IS TO GRP.

Chart SE. $$BDUMPB - Prepare and Edit a Line Subroutine
Refer to Chart 18.

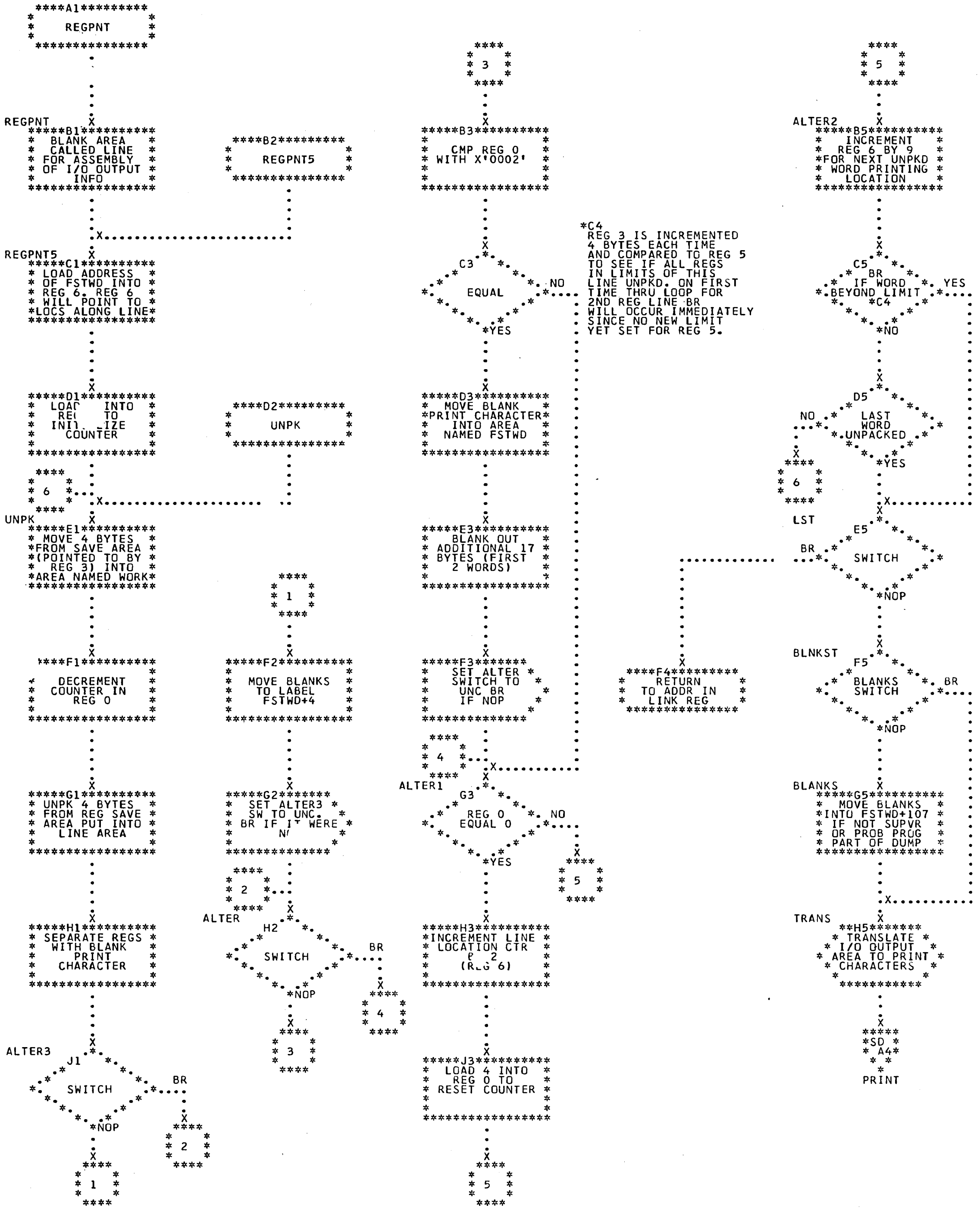

A1
REGPNT
REGPNT
B1
BLANK AREA CALLED LINE FOR ASSEMBLY OF I/O OUTPUT INFO
B2
REGPNT5
REGPNT5
C1
LOAD ADDRESS OF FSTWD INTO REG 6. REG 6 WILL POINT TO LOCS ALONG LINE
D1
LOAΓ INTO RE(TO INI1. _IZE COUNTER
D2
UNPK
6
UNPK
E1
MOVE 4 BYTES FROM SAVE AREA (POINTED TO BY REG 3) INTO AREA NAMED WORK
F1
DECREMENT COUNTER IN REG 0
G1
UNPK 4 BYTES FROM REG SAVE AREA PUT INTO LINE AREA
H1
SEPARATE REGS WITH BLANK PRINT CHARACTER
ALTER3
J1
SWITCH
BR
NOP
2
1
1
F2
MOVE BLANKS TO LABEL FSTWD+4
G2
SET ALTER3 SW TO UNC. BR IF IT WERE N(
2
ALTER
H2
SWITCH
BR
NOP
4
3
3
B3
CMP REG 0 WITH X'0002'
C3
EQUAL
NO
YES
*C4
REG 3 IS INCREMENTED 4 BYTES EACH TIME AND COMPARED TO REG 5 TO SEE IF ALL REGS IN LIMITS OF THIS LINE UNPKD. ON FIRST TIME THRU LOOP FOR 2ND REG LINE BR WILL OCCUR IMMEDIATELY SINCE NO NEW LIMIT YET SET FOR REG 5.
D3
MOVE BLANK PRINT CHARACTER INTO AREA NAMED FSTWD
E3
BLANK OUT ADDITIONAL 17 BYTES (FIRST 2 WORDS)
F3
SET ALTER SWITCH TO UNC BR IF NOP
4
ALTER1
G3
REG 0 EQUAL 0
NO
YES
5
H3
INCREMENT LINE LOCATION CTR ℓ 2 (RᴌG 6)
J3
LOAD 4 INTO REG 0 TO RESET COUNTER
5
F4
RETURN TO ADDR IN LINK REG
5
ALTER2
B5
INCREMENT REG 6 BY 9 FOR NEXT UNPKD WORD PRINTING LOCATION
C5
BR IF WORD BEYOND LIMIT *C4
YES
NO
D5
LAST WORD UNPACKED
NO
YES
6
LST
E5
SWITCH
BR
NOP
BLNKST
F5
BLANKS SWITCH
BR
NOP
BLANKS
G5
MOVE BLANKS INTO FSTWD+107 IF NOT SUPVR OR PROB PROG PART OF DUMP
TRANS
H5
TRANSLATE I/O OUTPUT AREA TO PRINT CHARACTERS
SD A4
PRINT

Chart SF. $$BDUMPB - Line Test Subroutines
Refer to Chart 18.

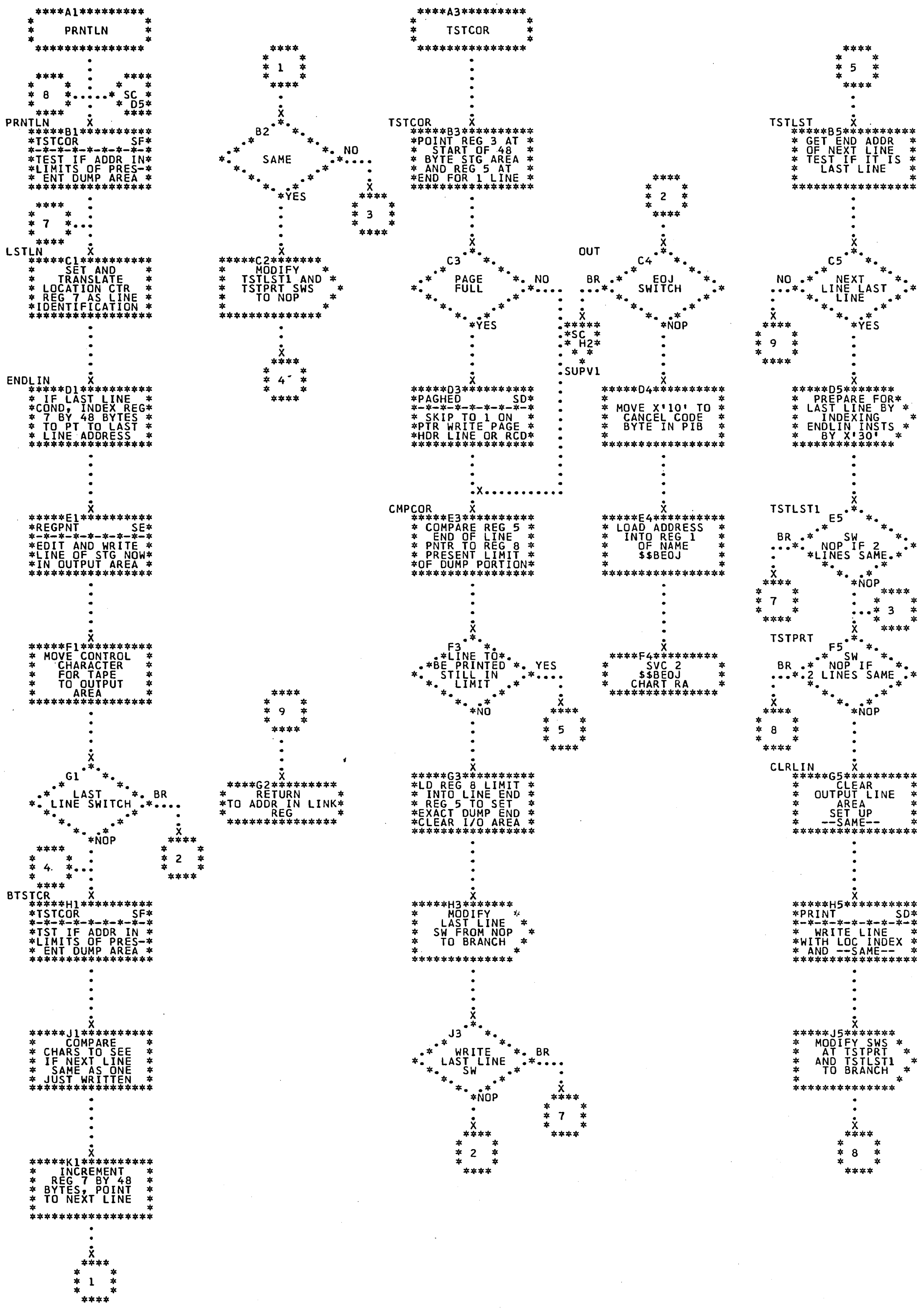

A1
PRNTLN
8
SC D5
PRNTLN
B1
TSTCOR SF
TEST IF ADDR IN LIMITS OF PRES-ENT DUMP AREA
7
LSTLN
C1
SET AND TRANSLATE LOCATION CTR REG 7 AS LINE IDENTIFICATION
ENDLIN
D1
IF LAST LINE COND, INDEX REG 7 BY 48 BYTES TO PT TO LAST LINE ADDRESS
E1
REGPNT SE
EDIT AND WRITE LINE OF STG NOW IN OUTPUT AREA
F1
MOVE CONTROL CHARACTER FOR TAPE TO OUTPUT AREA
G1
LAST LINE SWITCH
BR
NOP
4
2
BTSTCR
H1
TSTCOR SF
TST IF ADDR IN LIMITS OF PRES-ENT DUMP AREA
J1
COMPARE CHARS TO SEE IF NEXT LINE SAME AS ONE JUST WRITTEN
K1
INCREMENT REG 7 BY 48 BYTES, POINT TO NEXT LINE
1
1
B2
SAME
NO
YES
3
C2
MODIFY TSTLST1 AND TSTPRT SWS TO NOP
4
9
G2
RETURN TO ADDR IN LINK REG
A3
TSTCOR
TSTCOR
B3
POINT REG 3 AT START OF 48 BYTE STG AREA AND REG 5 AT END FOR 1 LINE
C3
PAGE FULL
NO
YES
SC H2
SUPV1
D3
PAGHED SD
SKIP TO 1 ON PTR WRITE PAGE HDR LINE OR RCD
CMPCOR
E3
COMPARE REG 5 END OF LINE PNTR TO REG 8 PRESENT LIMIT OF DUMP PORTION
F3
LINE TO BE PRINTED STILL IN LIMIT
YES
NO
5
G3
LD REG 8 LIMIT INTO LINE END REG 5 TO SET EXACT DUMP END CLEAR I/O AREA
H3
MODIFY LAST LINE SW FROM NOP TO BRANCH
J3
WRITE LAST LINE SW
BR
NOP
7
2
2
OUT
C4
BR
EOJ SWITCH
NOP
D4
MOVE X'10' TO CANCEL CODE BYTE IN PIB
E4
LOAD ADDRESS INTO REG 1 OF NAME $$BEOJ
F4
SVC 2 $$BEOJ CHART RA
5
TSTLST
B5
GET END ADDR OF NEXT LINE TEST IF IT IS LAST LINE
C5
NO
NEXT LINE LAST LINE
YES
9
D5
PREPARE FOR LAST LINE BY INDEXING ENDLIN INSTS BY X'30'
TSTLST1
E5
BR
SW NOP IF 2 LINES SAME
NOP
7
3
TSTPRT
F5
BR
SW NOP IF 2 LINES SAME
NOP
8
CLRLIN
G5
CLEAR OUTPUT LINE AREA SET UP --SAME--
H5
PRINT SD
WRITE LINE WITH LOC INDEX AND --SAME--
J5
MODIFY SWS AT TSTPRT AND TSTLST1 TO BRANCH
8

Chart SG. $$BDUMPD - Dump on Disk Device
Refer to Chart 18.

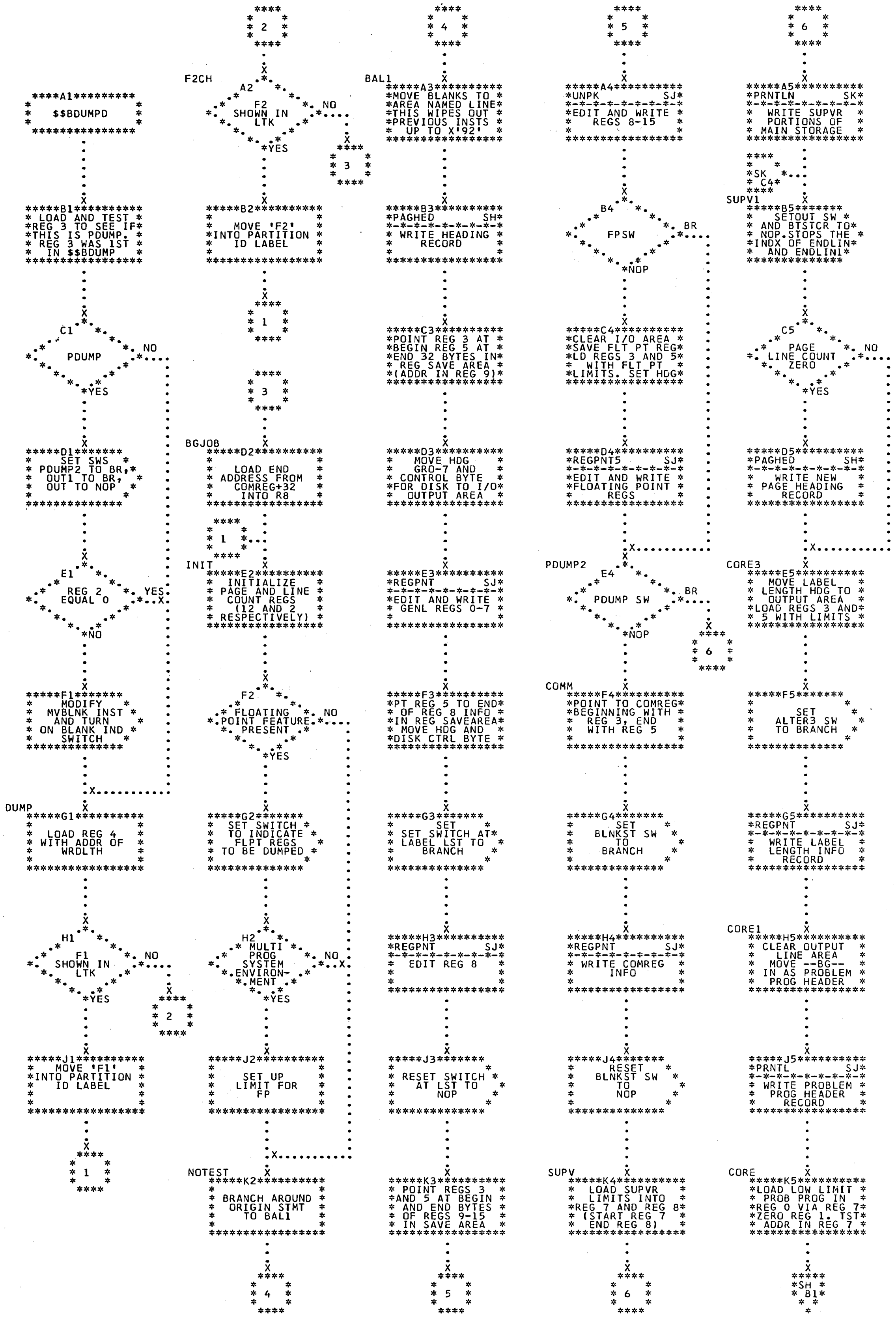


Chart SH. $$BDUMPD - Prepare Page Headings and PIOCS Subroutines
Refer to Chart 18.

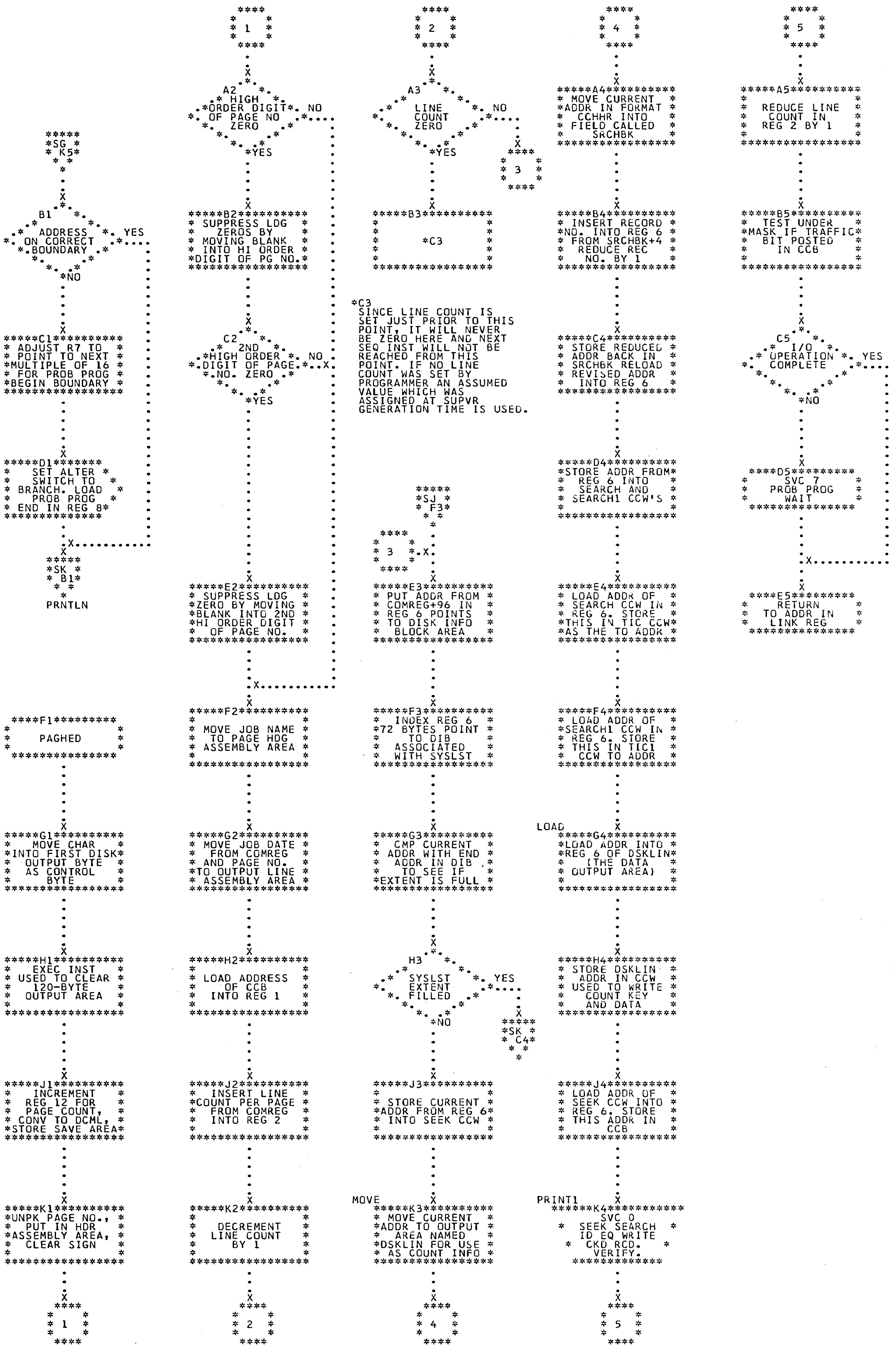

1
A2 HIGH ORDER DIGIT OF PAGE NO ZERO
NO
YES
2
A3 LINE COUNT ZERO
NO
YES
3
4
A4 MOVE CURRENT ADDR IN FORMAT CCHHR INTO FIELD CALLED SRCHBK
5
A5 REDUCE LINE COUNT IN REG 2 BY 1
SG K5
B1 ADDRESS ON CORRECT BOUNDARY
YES
NO
B2 SUPPRESS LDG ZEROS BY MOVING BLANK INTO HI ORDER DIGIT OF PG NO.
B3 *C3
B4 INSERT RECORD NO. INTO REG 6 FROM SRCHBK+4 REDUCE REC NO. BY 1
B5 TEST UNDER MASK IF TRAFFIC BIT POSTED IN CCB
*C3 SINCE LINE COUNT IS SET JUST PRIOR TO THIS POINT, IT WILL NEVER BE ZERO HERE AND NEXT SEQ INST WILL NOT BE REACHED FROM THIS POINT. IF NO LINE COUNT WAS SET BY PROGRAMMER AN ASSUMED VALUE WHICH WAS ASSIGNED AT SUPVR GENERATION TIME IS USED.
C1 ADJUST R7 TO POINT TO NEXT MULTIPLE OF 16 FOR PROB PROG BEGIN BOUNDARY
C2 2ND HIGH ORDER DIGIT OF PAGE NO. ZERO
NO
YES
C4 STORE REDUCED ADDR BACK IN SRCHBK RELOAD REVISED ADDR INTO REG 6
C5 I/O OPERATION COMPLETE
YES
NO
D1 SET ALTER SWITCH TO BRANCH. LOAD PROB PROG END IN REG 8
D4 STORE ADDR FROM REG 6 INTO SEARCH AND SEARCH1 CCW'S
D5 SVC 7 PROB PROG WAIT
SJ F3
3
SK B1
PRNTLN
E2 SUPPRESS LDG ZERO BY MOVING BLANK INTO 2ND HI ORDER DIGIT OF PAGE NO.
E3 PUT ADDR FROM COMREG+96 IN REG 6 POINTS TO DISK INFO BLOCK AREA
E4 LOAD ADDR OF SEARCH CCW IN REG 6. STORE THIS IN TIC CCW AS THE TO ADDR
E5 RETURN TO ADDR IN LINK REG
F1 PAGHED
F2 MOVE JOB NAME TO PAGE HDG ASSEMBLY AREA
F3 INDEX REG 6 72 BYTES POINT TO DIB ASSOCIATED WITH SYSLST
F4 LOAD ADDR OF SEARCH1 CCW IN REG 6. STORE THIS IN TIC1 CCW TO ADDR
G1 MOVE CHAR INTO FIRST DISK OUTPUT BYTE AS CONTROL BYTE
G2 MOVE JOB DATE FROM COMREG AND PAGE NO. TO OUTPUT LINE ASSEMBLY AREA
G3 CMP CURRENT ADDR WITH END ADDR IN DIB TO SEE IF EXTENT IS FULL
LOAD
G4 LOAD ADDR INTO REG 6 OF DSKLIN (THE DATA OUTPUT AREA)
H1 EXEC INST USED TO CLEAR 120-BYTE OUTPUT AREA
H2 LOAD ADDRESS OF CCB INTO REG 1
H3 SYSLST EXTENT FILLED
YES
NO
SK C4
H4 STORE DSKLIN ADDR IN CCW USED TO WRITE COUNT KEY AND DATA
J1 INCREMENT REG 12 FOR PAGE COUNT, CONV TO DCML, STORE SAVE AREA
J2 INSERT LINE COUNT PER PAGE FROM COMREG INTO REG 2
J3 STORE CURRENT ADDR FROM REG 6 INTO SEEK CCW
J4 LOAD ADDR OF SEEK CCW INTO REG 6. STORE THIS ADDR IN CCB
K1 UNPK PAGE NO., PUT IN HDR ASSEMBLY AREA, CLEAR SIGN
K2 DECREMENT LINE COUNT BY 1
MOVE
K3 MOVE CURRENT ADDR TO OUTPUT AREA NAMED DSKLIN FOR USE AS COUNT INFO
PRINT1
K4 SVC 0 SEEK SEARCH ID EQ WRITE CKD RCD. VERIFY.
1
2
4
5

Chart SJ. $$BDUMPD - Prepare and Edit a Line Subroutine
Refer to Chart 18.

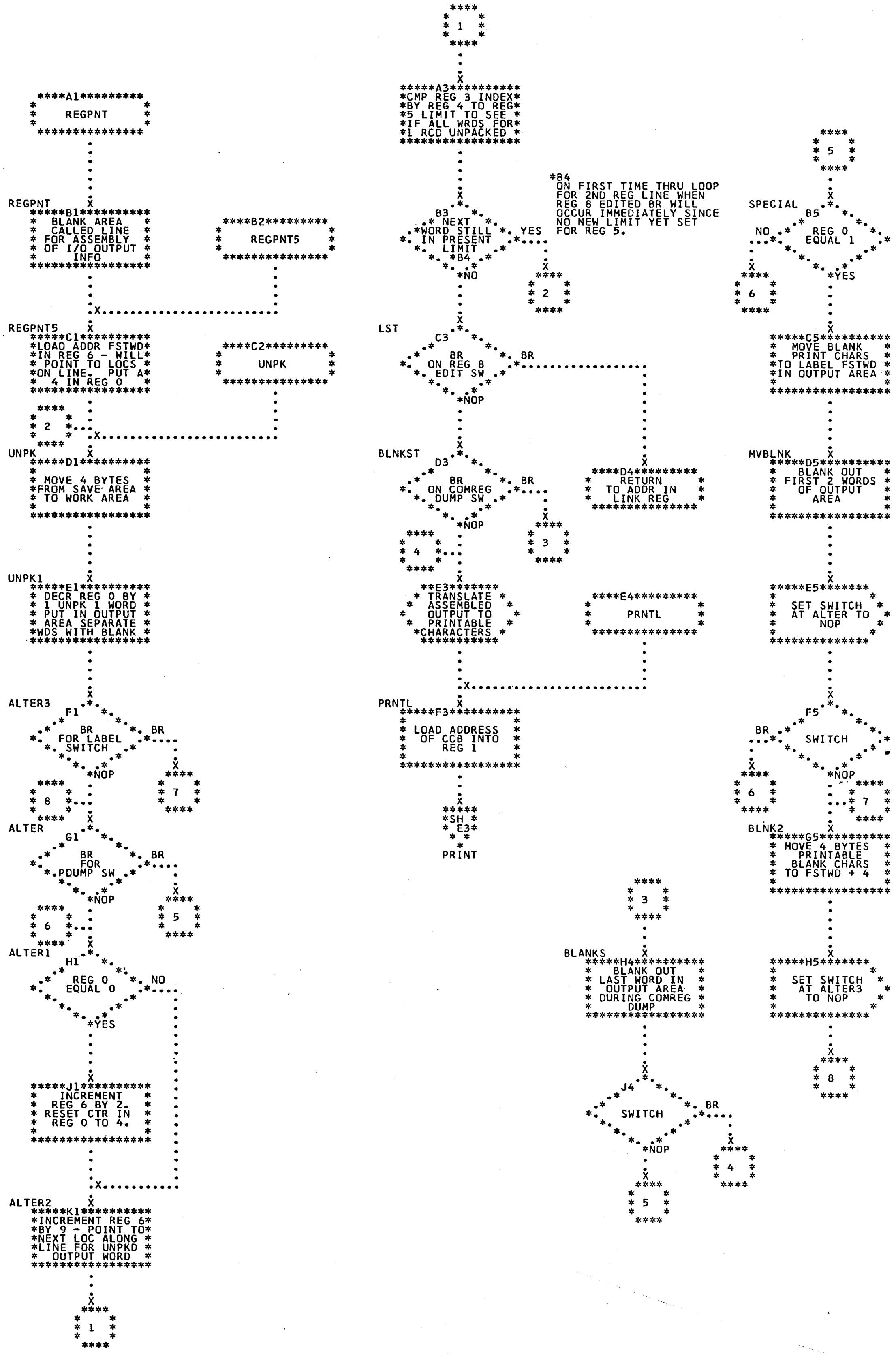

1
A1 REGPNT
REGPNT
B1 BLANK AREA CALLED LINE FOR ASSEMBLY OF I/O OUTPUT INFO
B2 REGPNT5
REGPNT5
C1 LOAD ADDR FSTWD IN REG 6 - WILL POINT TO LOCS ON LINE. PUT A 4 IN REG 0
C2 UNPK
2
UNPK
D1 MOVE 4 BYTES FROM SAVE AREA TO WORK AREA
UNPK1
E1 DECR REG 0 BY 1 UNPK 1 WORD PUT IN OUTPUT AREA SEPARATE WDS WITH BLANK
ALTER3
F1 BR FOR LABEL SWITCH
BR
NOP
8
7
ALTER
G1 BR FOR PDUMP SW
BR
NOP
6
5
ALTER1
H1 REG 0 EQUAL 0
NO
YES
J1 INCREMENT REG 6 BY 2. RESET CTR IN REG 0 TO 4.
ALTER2
K1 INCREMENT REG 6 BY 9 - POINT TO NEXT LOC ALONG LINE FOR UNPKD OUTPUT WORD
1
1
A3 CMP REG 3 INDEX BY REG 4 TO REG 5 LIMIT TO SEE IF ALL WRDS FOR 1 RCD UNPACKED
B3 NEXT WORD STILL IN PRESENT LIMIT *B4
YES
2
NO
*B4 ON FIRST TIME THRU LOOP FOR 2ND REG LINE WHEN REG 8 EDITED BR WILL OCCUR IMMEDIATELY SINCE NO NEW LIMIT YET SET FOR REG 5.
LST
C3 BR ON REG 8 EDIT SW
BR
NOP
BLNKST
D3 BR ON COMREG DUMP SW
BR
NOP
3
4
D4 RETURN TO ADDR IN LINK REG
E3 TRANSLATE ASSEMBLED OUTPUT TO PRINTABLE CHARACTERS
E4 PRNTL
PRNTL
F3 LOAD ADDRESS OF CCB INTO REG 1
SH E3
PRINT
3
BLANKS
H4 BLANK OUT LAST WORD IN OUTPUT AREA DURING COMREG DUMP
J4 SWITCH
BR
NOP
4
5
5
SPECIAL
B5 REG 0 EQUAL 1
NO
6
YES
C5 MOVE BLANK PRINT CHARS TO LABEL FSTWD IN OUTPUT AREA
MVBLNK
D5 BLANK OUT FIRST 2 WORDS OF OUTPUT AREA
E5 SET SWITCH AT ALTER TO NOP
F5 SWITCH
BR
6
NOP
7
BLNK2
G5 MOVE 4 BYTES PRINTABLE BLANK CHARS TO FSTWD + 4
H5 SET SWITCH AT ALTER3 TO NOP
8

Chart SK. $$BDUMPD - Line Test Subroutines
Refer to Chart 18.

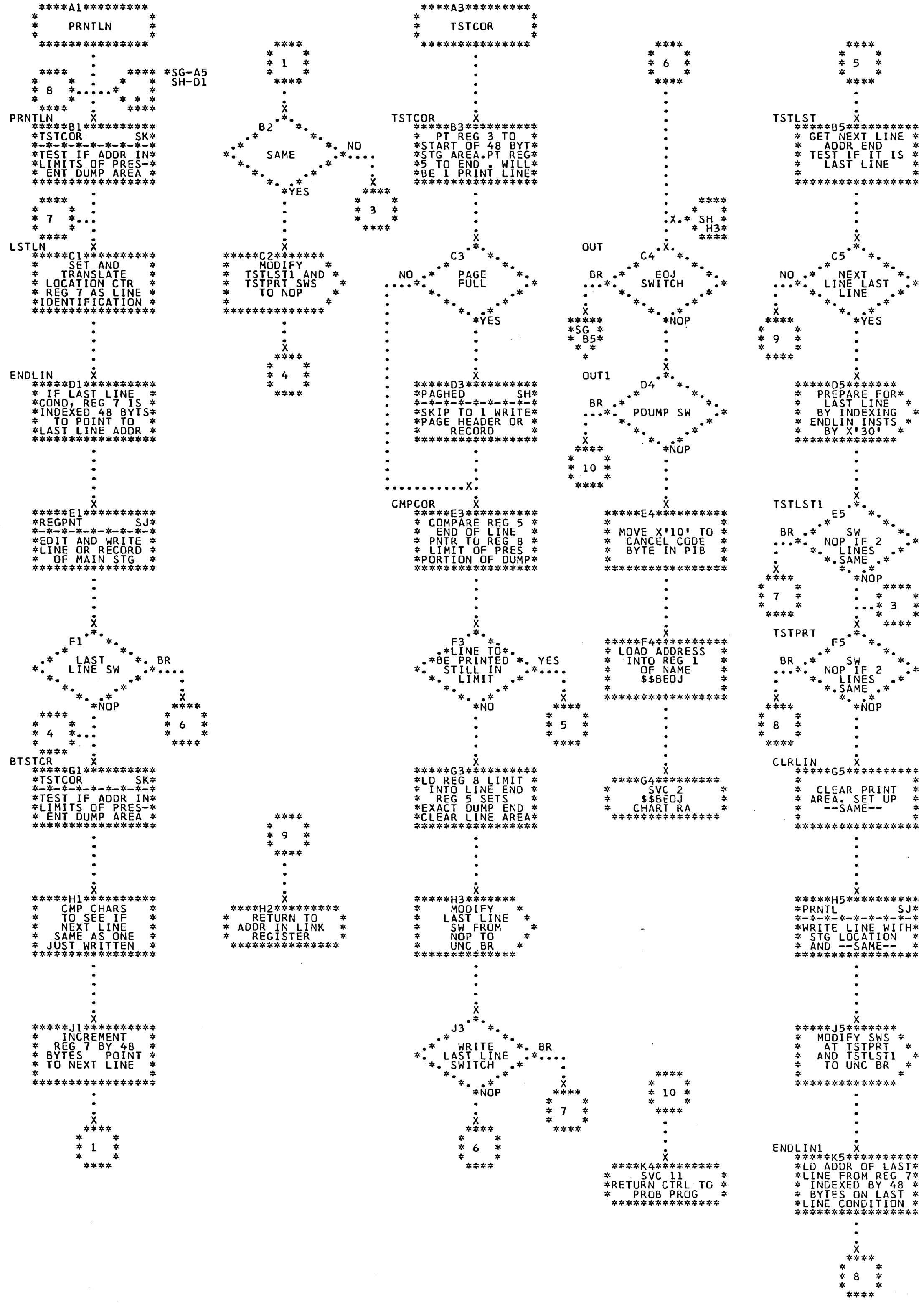

A1 PRNTLN
8
*SG-A5 SH-D1
PRNTLN B1 TSTCOR SK
TEST IF ADDR IN LIMITS OF PRESENT DUMP AREA
7
LSTLN C1 SET AND TRANSLATE LOCATION CTR REG 7 AS LINE IDENTIFICATION
ENDLIN D1 IF LAST LINE COND, REG 7 IS INDEXED 48 BYTS TO POINT TO LAST LINE ADDR
E1 REGPNT SJ
EDIT AND WRITE LINE OR RECORD OF MAIN STG
F1 LAST LINE SW
BR
NOP
4
6
BTSTCR G1 TSTCOR SK
TEST IF ADDR IN LIMITS OF PRESENT DUMP AREA
H1 CMP CHARS TO SEE IF NEXT LINE SAME AS ONE JUST WRITTEN
J1 INCREMENT REG 7 BY 48 BYTES POINT TO NEXT LINE
1
1
B2 SAME
NO
YES
3
C2 MODIFY TSTLST1 AND TSTPRT SWS TO NOP
4
9
H2 RETURN TO ADDR IN LINK REGISTER
A3 TSTCOR
TSTCOR B3 PT REG 3 TO START OF 48 BYT STG AREA.PT REG 5 TO END . WILL BE 1 PRINT LINE
C3 PAGE FULL
NO
YES
D3 PAGHED SH
SKIP TO 1 WRITE PAGE HEADER OR RECORD
CMPCOR E3 COMPARE REG 5 END OF LINE PNTR TO REG 8 LIMIT OF PRES PORTION OF DUMP
F3 LINE TO BE PRINTED STILL IN LIMIT
YES
NO
5
G3 LD REG 8 LIMIT INTO LINE END REG 5 SETS EXACT DUMP END CLEAR LINE AREA
H3 MODIFY LAST LINE SW FROM NOP TO UNC BR
J3 WRITE LAST LINE SWITCH
BR
NOP
7
6
6
SH H3
OUT C4 EOJ SWITCH
BR
NOP
SG B5
OUT1 D4 PDUMP SW
BR
NOP
10
E4 MOVE X'10' TO CANCEL CODE BYTE IN PIB
F4 LOAD ADDRESS INTO REG 1 OF NAME $$BEOJ
G4 SVC 2 $$BEOJ CHART RA
10
K4 SVC 11 RETURN CTRL TO PROB PROG
5
TSTLST B5 GET NEXT LINE ADDR END TEST IF IT IS LAST LINE
C5 NEXT LINE LAST LINE
NO
YES
9
D5 PREPARE FOR LAST LINE BY INDEXING ENDLIN INSTS BY X'30'
TSTLST1 E5 SW NOP IF 2 LINES SAME
BR
NOP
7
3
TSTPRT F5 SW NOP IF 2 LINES SAME
BR
NOP
8
CLRLIN G5 CLEAR PRINT AREA. SET UP --SAME--
H5 PRNTL SJ
WRITE LINE WITH STG LOCATION AND --SAME--
J5 MODIFY SWS AT TSTPRT AND TSTLST1 TO UNC BR
ENDLIN1 K5 LD ADDR OF LAST LINE FROM REG 7 INDEXED BY 48 BYTES ON LAST LINE CONDITION
8

Chart SL. $$BPDUMP - Parameter Storage Dump Monitor
Refer to Chart 19.

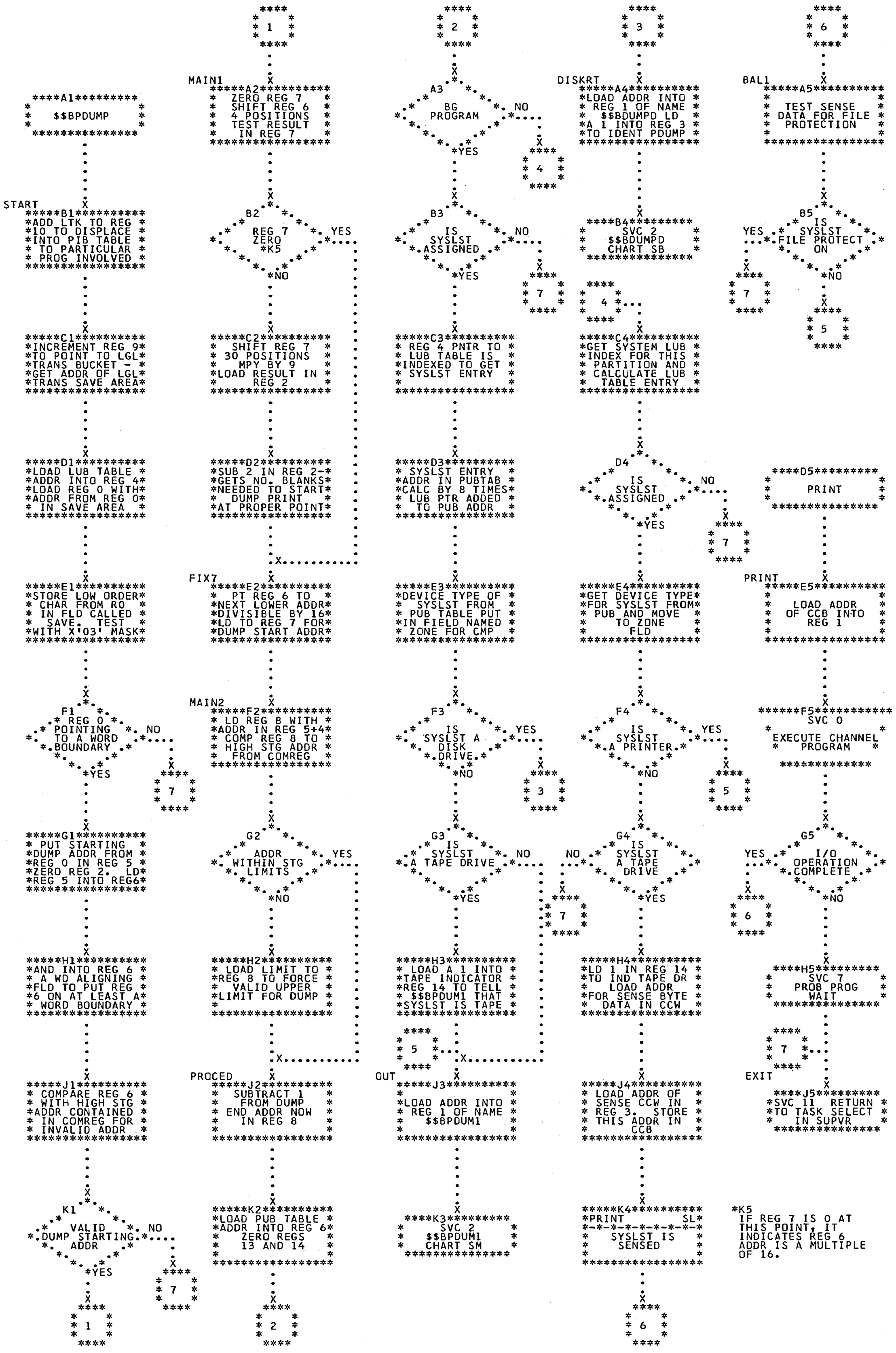

1
2
3
6
A1
$$BPDUMP
MAIN1
A2
ZERO REG 7
SHIFT REG 6
4 POSITIONS
TEST RESULT
IN REG 7
A3
BG PROGRAM
NO
YES
4
DISKRT
A4
LOAD ADDR INTO REG 1 OF NAME $$BDUMPD LD A 1 INTO REG 3 TO IDENT PDUMP
BAL1
A5
TEST SENSE DATA FOR FILE PROTECTION
START
B1
ADD LTK TO REG 10 TO DISPLACE INTO PIB TABLE TO PARTICULAR PROG INVOLVED
B2
REG 7 ZERO *K5
YES
NO
B3
IS SYSLST ASSIGNED
NO
YES
7
B4
SVC 2
$$BDUMPD
CHART SB
4
B5
IS SYSLST FILE PROTECT ON
YES
NO
7
5
C1
INCREMENT REG 9 TO POINT TO LGL TRANS BUCKET - GET ADDR OF LGL TRANS SAVE AREA
C2
SHIFT REG 7 30 POSITIONS MPY BY 9 LOAD RESULT IN REG 2
C3
REG 4 PNTR TO LUB TABLE IS INDEXED TO GET SYSLST ENTRY
C4
GET SYSTEM LUB INDEX FOR THIS PARTITION AND CALCULATE LUB TABLE ENTRY
D1
LOAD LUB TABLE ADDR INTO REG 4 LOAD REG 0 WITH ADDR FROM REG 0 IN SAVE AREA
D2
SUB 2 IN REG 2- GETS NO. BLANKS NEEDED TO START DUMP PRINT AT PROPER POINT
D3
SYSLST ENTRY ADDR IN PUBTAB CALC BY 8 TIMES LUB PTR ADDED TO PUB ADDR
D4
IS SYSLST ASSIGNED
NO
YES
7
D5
PRINT
E1
STORE LOW ORDER CHAR FROM R0 IN FLD CALLED SAVE. TEST WITH X'03' MASK
FIX7
E2
PT REG 6 TO NEXT LOWER ADDR DIVISIBLE BY 16 LD TO REG 7 FOR DUMP START ADDR
E3
DEVICE TYPE OF SYSLST FROM PUB TABLE PUT IN FIELD NAMED ZONE FOR CMP
E4
GET DEVICE TYPE FOR SYSLST FROM PUB AND MOVE TO ZONE FLD
PRINT
E5
LOAD ADDR OF CCB INTO REG 1
F1
REG 0 POINTING TO A WORD BOUNDARY
NO
YES
7
MAIN2
F2
LD REG 8 WITH ADDR IN REG 5+4 COMP REG 8 TO HIGH STG ADDR FROM COMREG
F3
IS SYSLST A DISK DRIVE
YES
NO
3
F4
IS SYSLST A PRINTER
YES
NO
5
F5
SVC 0
EXECUTE CHANNEL PROGRAM
G1
PUT STARTING DUMP ADDR FROM REG 0 IN REG 5 ZERO REG 2. LD REG 5 INTO REG6
G2
ADDR WITHIN STG LIMITS
YES
NO
G3
IS SYSLST A TAPE DRIVE
NO
YES
7
G4
IS SYSLST A TAPE DRIVE
NO
YES
G5
I/O OPERATION COMPLETE
YES
NO
6
H1
AND INTO REG 6 A WD ALIGNING FLD TO PUT REG 6 ON AT LEAST A WORD BOUNDARY
H2
LOAD LIMIT TO REG 8 TO FORCE VALID UPPER LIMIT FOR DUMP
H3
LOAD A 1 INTO TAPE INDICATOR REG 14 TO TELL $$BPDUM1 THAT SYSLST IS TAPE
H4
LD 1 IN REG 14 TO IND TAPE DR LOAD ADDR FOR SENSE BYTE DATA IN CCW
H5
SVC 7
PROB PROG WAIT
5
7
J1
COMPARE REG 6 WITH HIGH STG ADDR CONTAINED IN COMREG FOR INVALID ADDR
PROCED
J2
SUBTRACT 1 FROM DUMP END ADDR NOW IN REG 8
OUT
J3
LOAD ADDR INTO REG 1 OF NAME $$BPDUM1
J4
LOAD ADDR OF SENSE CCW IN REG 3. STORE THIS ADDR IN CCB
EXIT
J5
SVC 11 RETURN TO TASK SELECT IN SUPVR
K1
VALID DUMP STARTING ADDR
NO
YES
7
1
K2
LOAD PUB TABLE ADDR INTO REG 6 ZERO REGS 13 AND 14
2
K3
SVC 2
$$BPDUM1
CHART SM
K4
PRINT SL
SYSLST IS SENSED
6
*K5
IF REG 7 IS 0 AT THIS POINT, IT INDICATES REG 6 ADDR IS A MULTIPLE OF 16.

Chart SM. $$BPDUM1 - Initialize Parameter Dump on Printer or Tape
Refer to Chart 19.

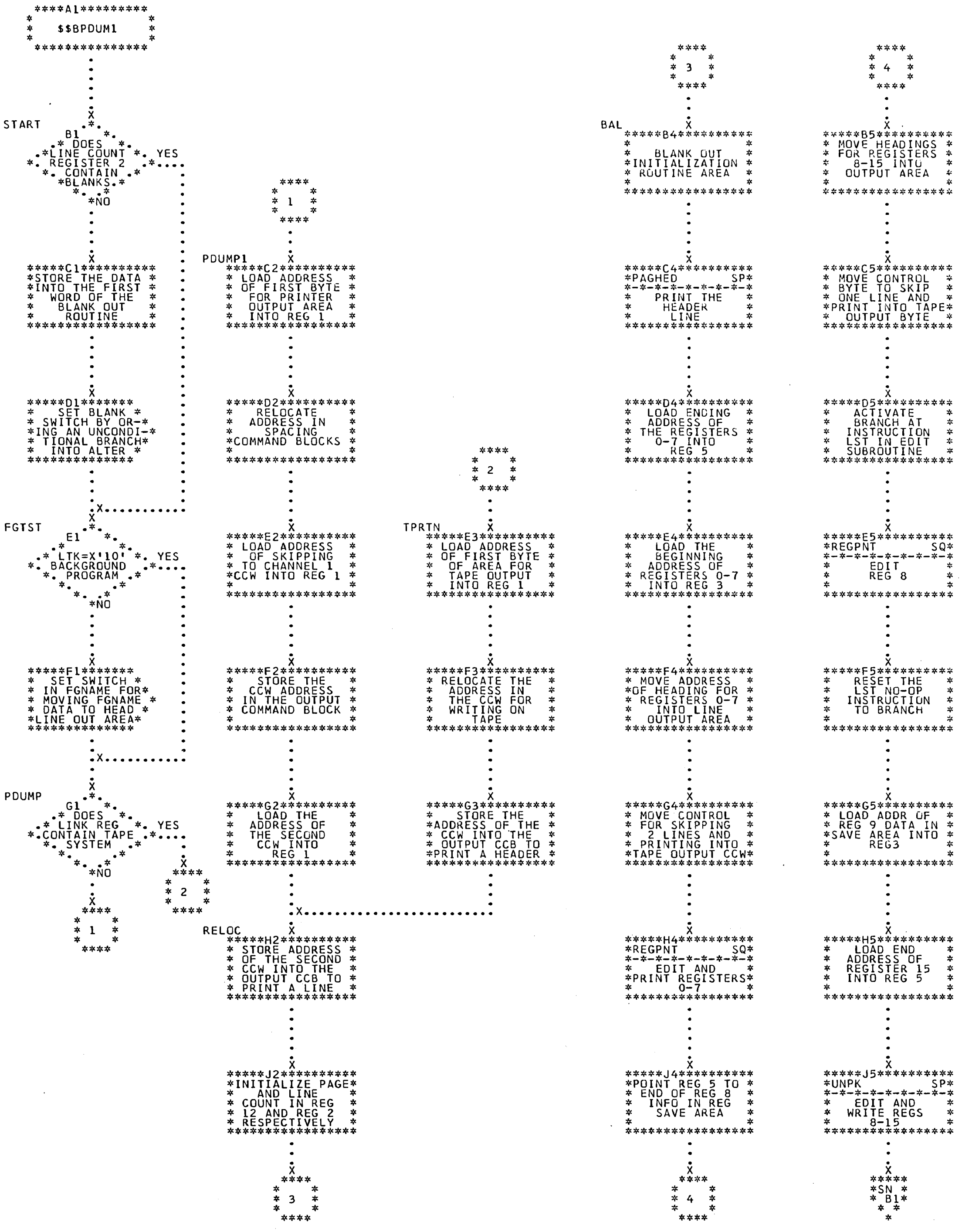

A1
$$BPDUM1
START
B1
DOES LINE COUNT REGISTER 2 CONTAIN BLANKS
YES
NO
C1
STORE THE DATA INTO THE FIRST WORD OF THE BLANK OUT ROUTINE
D1
SET BLANK SWITCH BY OR-ING AN UNCONDI-TIONAL BRANCH INTO ALTER
FGTST
E1
LTK=X'10' BACKGROUND PROGRAM
YES
NO
F1
SET SWITCH IN FGNAME FOR MOVING FGNAME DATA TO HEAD LINE OUT AREA
PDUMP
G1
DOES LINK REG CONTAIN TAPE SYSTEM
YES
NO
2
1
1
PDUMP1
C2
LOAD ADDRESS OF FIRST BYTE FOR PRINTER OUTPUT AREA INTO REG 1
D2
RELOCATE ADDRESS IN SPACING COMMAND BLOCKS
E2
LOAD ADDRESS OF SKIPPING TO CHANNEL 1 CCW INTO REG 1
F2
STORE THE CCW ADDRESS IN THE OUTPUT COMMAND BLOCK
G2
LOAD THE ADDRESS OF THE SECOND CCW INTO REG 1
RELOC
H2
STORE ADDRESS OF THE SECOND CCW INTO THE OUTPUT CCB TO PRINT A LINE
J2
INITIALIZE PAGE AND LINE COUNT IN REG 12 AND REG 2 RESPECTIVELY
3
2
TPRTN
E3
LOAD ADDRESS OF FIRST BYTE OF AREA FOR TAPE OUTPUT INTO REG 1
F3
RELOCATE THE ADDRESS IN THE CCW FOR WRITING ON TAPE
G3
STORE THE ADDRESS OF THE CCW INTO THE OUTPUT CCB TO PRINT A HEADER
3
BAL
B4
BLANK OUT INITIALIZATION ROUTINE AREA
C4
PAGHED SP
PRINT THE HEADER LINE
D4
LOAD ENDING ADDRESS OF THE REGISTERS 0-7 INTO REG 5
E4
LOAD THE BEGINNING ADDRESS OF REGISTERS 0-7 INTO REG 3
F4
MOVE ADDRESS OF HEADING FOR REGISTERS 0-7 INTO LINE OUTPUT AREA
G4
MOVE CONTROL FOR SKIPPING 2 LINES AND PRINTING INTO TAPE OUTPUT CCW
H4
REGPNT SQ
EDIT AND PRINT REGISTERS 0-7
J4
POINT REG 5 TO END OF REG 8 INFO IN REG SAVE AREA
4
4
B5
MOVE HEADINGS FOR REGISTERS 8-15 INTO OUTPUT AREA
C5
MOVE CONTROL BYTE TO SKIP ONE LINE AND PRINT INTO TAPE OUTPUT BYTE
D5
ACTIVATE BRANCH AT INSTRUCTION LST IN EDIT SUBROUTINE
E5
REGPNT SQ
EDIT REG 8
F5
RESET THE LST NO-OP INSTRUCTION TO BRANCH
G5
LOAD ADDR OF REG 9 DATA IN SAVE AREA INTO REG3
H5
LOAD END ADDRESS OF REGISTER 15 INTO REG 5
J5
UNPK SP
EDIT AND WRITE REGS 8-15
SN B1

Chart SN. $$BPDUM1 - Parameter Storage Dump on Printer or Tape
Refer to Chart 19.

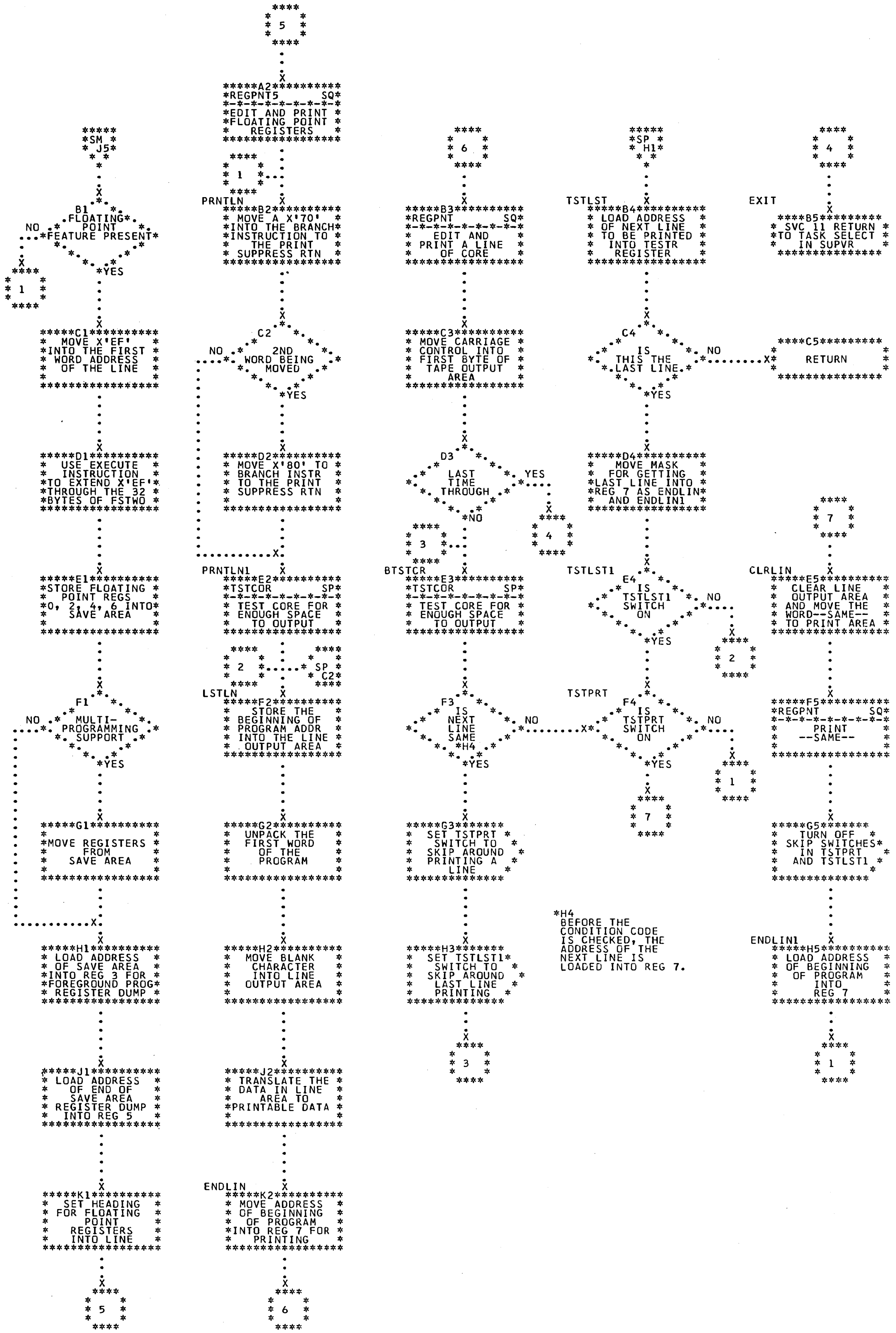

5
A2 REGPNT5 SQ
EDIT AND PRINT FLOATING POINT REGISTERS
SM J5
1
B1 FLOATING POINT FEATURE PRESENT
NO
YES
1
PRNTLN
B2 MOVE A X'70' INTO THE BRANCH INSTRUCTION TO THE PRINT SUPPRESS RTN
6
B3 REGPNT SQ
EDIT AND PRINT A LINE OF CORE
SP H1
TSTLST
B4 LOAD ADDRESS OF NEXT LINE TO BE PRINTED INTO TESTR REGISTER
4
EXIT
B5 SVC 11 RETURN TO TASK SELECT IN SUPVR
C1 MOVE X'EF' INTO THE FIRST WORD ADDRESS OF THE LINE
C2 2ND WORD BEING MOVED
NO
YES
C3 MOVE CARRIAGE CONTROL INTO FIRST BYTE OF TAPE OUTPUT AREA
C4 IS THIS THE LAST LINE.
NO
YES
C5 RETURN
D1 USE EXECUTE INSTRUCTION TO EXTEND X'EF' THROUGH THE 32 BYTES OF FSTWO
D2 MOVE X'80' TO BRANCH INSTR TO THE PRINT SUPPRESS RTN
D3 LAST TIME THROUGH
YES
NO
4
3
D4 MOVE MASK FOR GETTING LAST LINE INTO REG 7 AS ENDLIN AND ENDLIN1
7
E1 STORE FLOATING POINT REGS 0, 2, 4, 6 INTO SAVE AREA
PRNTLN1
E2 TSTCOR SP
TEST CORE FOR ENOUGH SPACE TO OUTPUT
BTSTCR
E3 TSTCOR SP
TEST CORE FOR ENOUGH SPACE TO OUTPUT
TSTLST1
E4 IS TSTLST1 SWITCH ON
NO
YES
2
CLRLIN
E5 CLEAR LINE OUTPUT AREA AND MOVE THE WORD--SAME-- TO PRINT AREA
2
SP C2
F1 MULTI-PROGRAMMING SUPPORT
NO
YES
LSTLN
F2 STORE THE BEGINNING OF PROGRAM ADDR INTO THE LINE OUTPUT AREA
F3 IS NEXT LINE SAME *H4
NO
YES
TSTPRT
F4 IS TSTPRT SWITCH ON
NO
YES
1
7
F5 REGPNT SQ
PRINT --SAME--
G1 MOVE REGISTERS FROM SAVE AREA
G2 UNPACK THE FIRST WORD OF THE PROGRAM
G3 SET TSTPRT SWITCH TO SKIP AROUND PRINTING A LINE
G5 TURN OFF SKIP SWITCHES IN TSTPRT AND TSTLST1
*H4 BEFORE THE CONDITION CODE IS CHECKED, THE ADDRESS OF THE NEXT LINE IS LOADED INTO REG 7.
H1 LOAD ADDRESS OF SAVE AREA INTO REG 3 FOR FOREGROUND PROG REGISTER DUMP
H2 MOVE BLANK CHARACTER INTO LINE OUTPUT AREA
H3 SET TSTLST1 SWITCH TO SKIP AROUND LAST LINE PRINTING
ENDLIN1
H5 LOAD ADDRESS OF BEGINNING OF PROGRAM INTO REG 7
3
1
J1 LOAD ADDRESS OF END OF SAVE AREA REGISTER DUMP INTO REG 5
J2 TRANSLATE THE DATA IN LINE AREA TO PRINTABLE DATA
K1 SET HEADING FOR FLOATING POINT REGISTERS INTO LINE
ENDLIN
K2 MOVE ADDRESS OF BEGINNING OF PROGRAM INTO REG 7 FOR PRINTING
5
6

Chart SP. $$BPDUM1 - Line Test Subroutines
Refer to Chart 19.

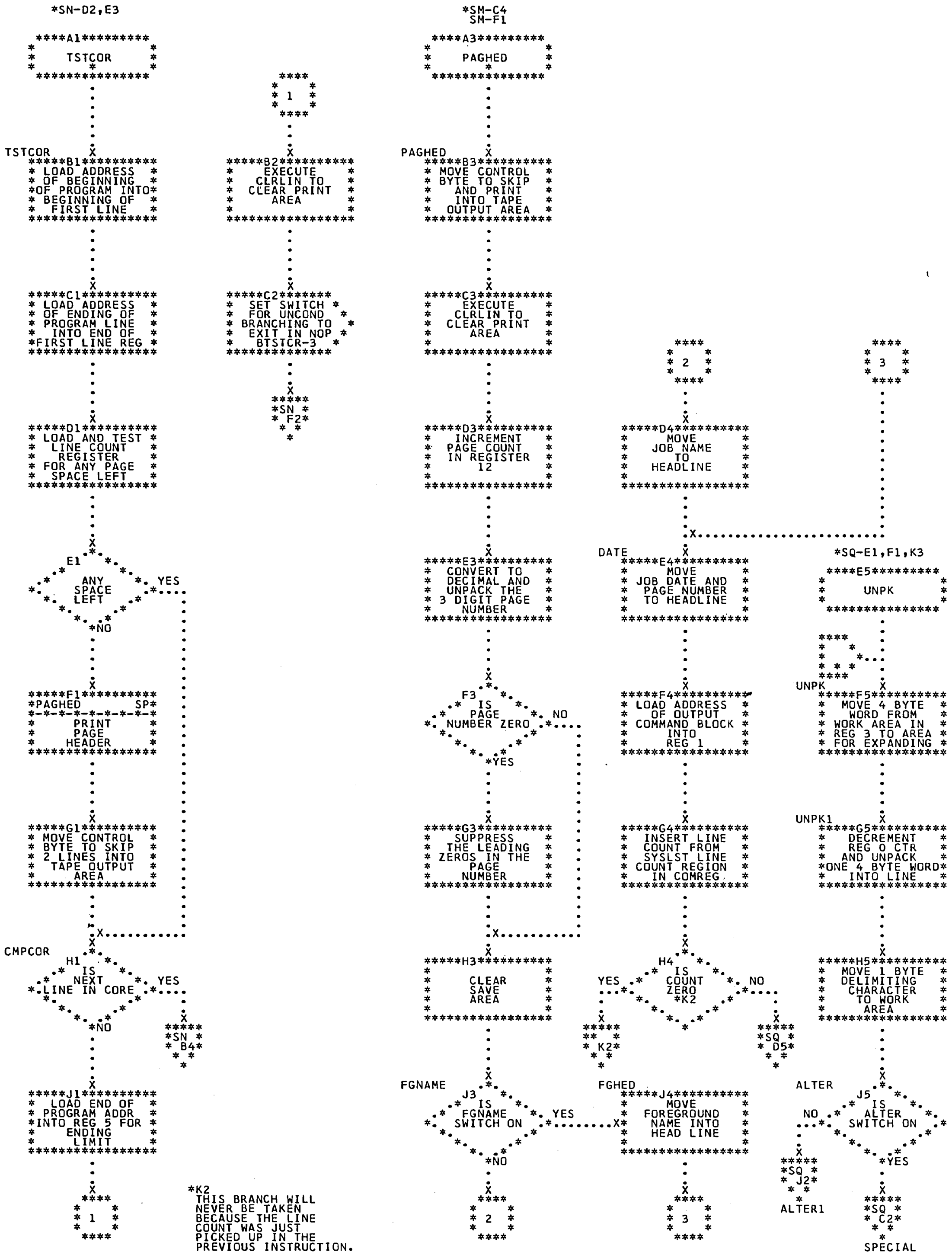

*SN-D2,E3
A1 TSTCOR
TSTCOR B1 LOAD ADDRESS OF BEGINNING OF PROGRAM INTO BEGINNING OF FIRST LINE
C1 LOAD ADDRESS OF ENDING OF PROGRAM LINE INTO END OF FIRST LINE REG
D1 LOAD AND TEST LINE COUNT REGISTER FOR ANY PAGE SPACE LEFT
E1 ANY SPACE LEFT
YES
NO
F1 PAGHED SP PRINT PAGE HEADER
G1 MOVE CONTROL BYTE TO SKIP 2 LINES INTO TAPE OUTPUT AREA
CMPCOR H1 IS NEXT LINE IN CORE
YES
SN B4
NO
J1 LOAD END OF PROGRAM ADDR INTO REG 5 FOR ENDING LIMIT
1
1
B2 EXECUTE CLRLIN TO CLEAR PRINT AREA
C2 SET SWITCH FOR UNCOND BRANCHING TO EXIT IN NOP BTSTCR-3
SN F2
*K2
THIS BRANCH WILL NEVER BE TAKEN BECAUSE THE LINE COUNT WAS JUST PICKED UP IN THE PREVIOUS INSTRUCTION.
*SM-C4
SM-F1
A3 PAGHED
PAGHED B3 MOVE CONTROL BYTE TO SKIP AND PRINT INTO TAPE OUTPUT AREA
C3 EXECUTE CLRLIN TO CLEAR PRINT AREA
D3 INCREMENT PAGE COUNT IN REGISTER 12
E3 CONVERT TO DECIMAL AND UNPACK THE 3 DIGIT PAGE NUMBER
F3 IS PAGE NUMBER ZERO
NO
YES
G3 SUPPRESS THE LEADING ZEROS IN THE PAGE NUMBER
H3 CLEAR SAVE AREA
FGNAME J3 IS FGNAME SWITCH ON
YES
NO
2
2
D4 MOVE JOB NAME TO HEADLINE
DATE E4 MOVE JOB DATE AND PAGE NUMBER TO HEADLINE
F4 LOAD ADDRESS OF OUTPUT COMMAND BLOCK INTO REG 1
G4 INSERT LINE COUNT FROM SYSLST LINE COUNT REGION IN COMREG
H4 IS COUNT ZERO *K2
YES
K2
NO
SQ D5
FGHED J4 MOVE FOREGROUND NAME INTO HEAD LINE
3
3
*SQ-E1,F1,K3
E5 UNPK
UNPK F5 MOVE 4 BYTE WORD FROM WORK AREA IN REG 3 TO AREA FOR EXPANDING
UNPK1 G5 DECREMENT REG 0 CTR AND UNPACK ONE 4 BYTE WORD INTO LINE
H5 MOVE 1 BYTE DELIMITING CHARACTER TO WORK AREA
ALTER J5 IS ALTER SWITCH ON
NO
SQ J2
ALTER1
YES
SQ C2
SPECIAL

Chart SQ. $$BPDUM1 - Prepare and Edit a Line Subroutine
Refer to Chart 19.

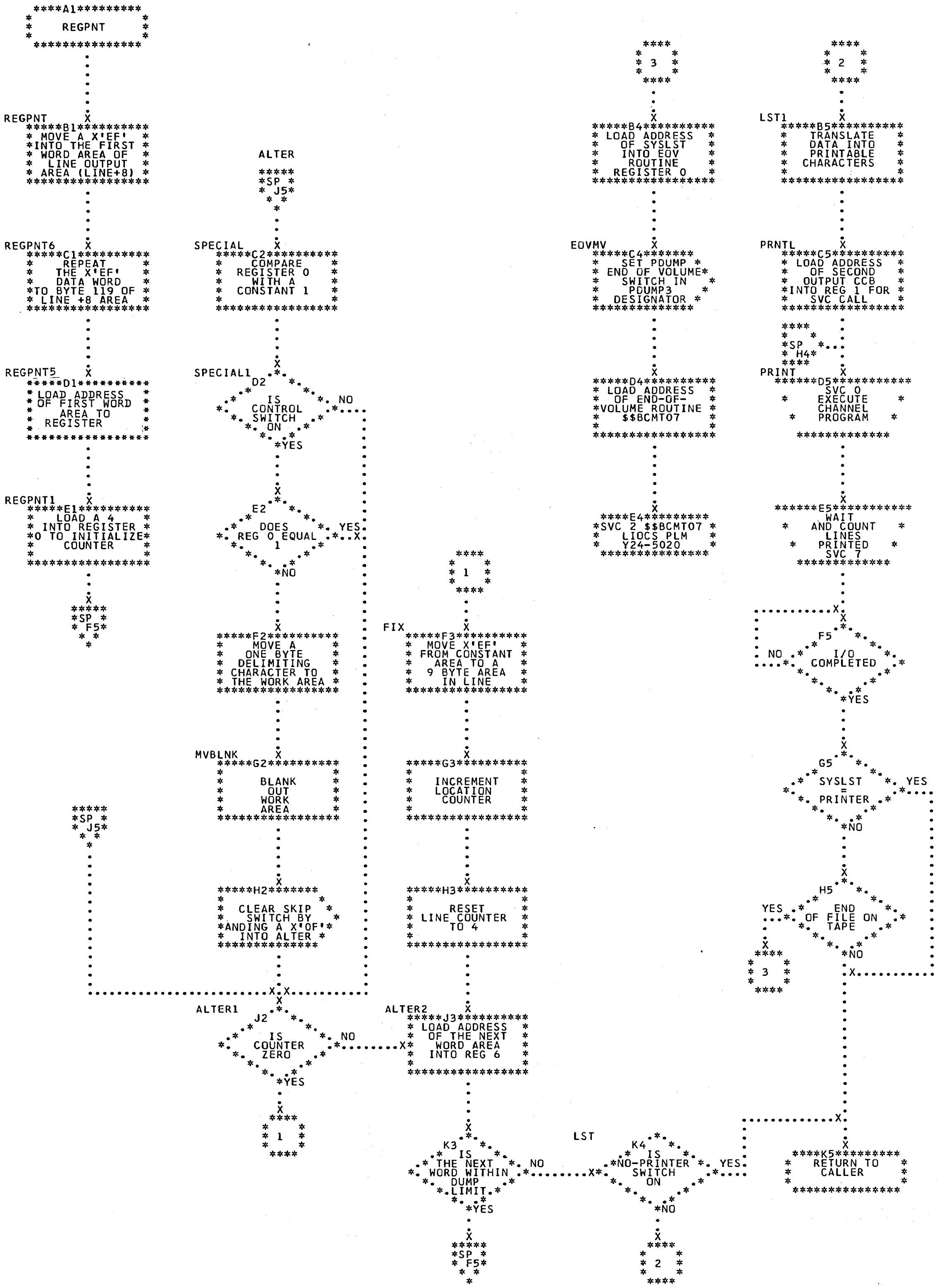
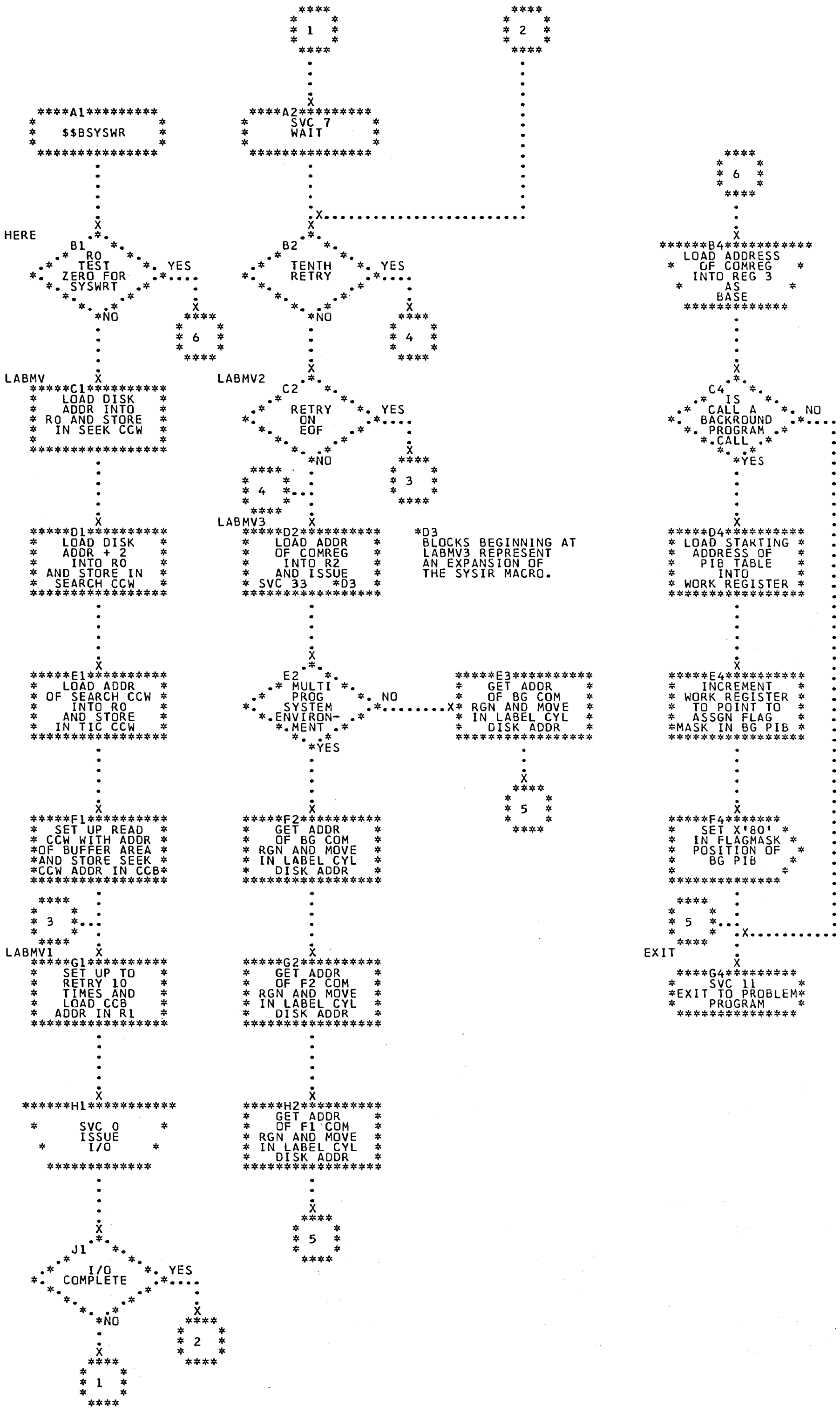

A1
REGPNT
REGPNT
B1
MOVE A X'EF' INTO THE FIRST WORD AREA OF LINE OUTPUT AREA (LINE+8)
REGPNT6
C1
REPEAT THE X'EF' DATA WORD TO BYTE 119 OF LINE +8 AREA
REGPNT5
D1
LOAD ADDRESS OF FIRST WORD AREA TO REGISTER
REGPNT1
E1
LOAD A 4 INTO REGISTER 0 TO INITIALIZE COUNTER
SP F5
SP J5
ALTER
SP J5
SPECIAL
C2
COMPARE REGISTER 0 WITH A CONSTANT 1
SPECIAL1
D2
IS CONTROL SWITCH ON
NO
YES
E2
DOES REG 0 EQUAL 1
YES
NO
F2
MOVE A ONE BYTE DELIMITING CHARACTER TO THE WORK AREA
MVBLNK
G2
BLANK OUT WORK AREA
H2
CLEAR SKIP SWITCH BY ANDING A X'0F' INTO ALTER
ALTER1
J2
IS COUNTER ZERO
NO
YES
1
1
FIX
F3
MOVE X'EF' FROM CONSTANT AREA TO A 9 BYTE AREA IN LINE
G3
INCREMENT LOCATION COUNTER
H3
RESET LINE COUNTER TO 4
ALTER2
J3
LOAD ADDRESS OF THE NEXT WORD AREA INTO REG 6
K3
IS THE NEXT WORD WITHIN DUMP LIMIT
NO
YES
SP F5
LST
K4
IS NO-PRINTER SWITCH ON
YES
NO
2
3
B4
LOAD ADDRESS OF SYSLST INTO EOV ROUTINE REGISTER 0
EOVMV
C4
SET PDUMP END OF VOLUME SWITCH IN PDUMP3 DESIGNATOR
D4
LOAD ADDRESS OF END-OF-VOLUME ROUTINE $$BCMT07
E4
SVC 2 $$BCMT07 LIOCS PLM Y24-5020
2
LST1
B5
TRANSLATE DATA INTO PRINTABLE CHARACTERS
PRNTL
C5
LOAD ADDRESS OF SECOND OUTPUT CCB INTO REG 1 FOR SVC CALL
SP H4
PRINT
D5
SVC 0 EXECUTE CHANNEL PROGRAM
E5
WAIT AND COUNT LINES PRINTED SVC 7
F5
NO
I/O COMPLETED
YES
G5
SYSLST = PRINTER
YES
NO
H5
YES
END OF FILE ON TAPE
NO
3
K5
RETURN TO CALLER

Chart TA. $$BSYSWR - Set up a Write on SYSRES Operation; Move Label Cylinder Address to COMREG



Label	Phase	Chart
A	$$BEOJ	RL
A01	$$ANERRG	LM
A02	$$ANERRG	LM
A03	$$ANERRJ	LS
A05	$$ANERRJ	LS
A051	$$ANERRJ	LS
A06	$$ANERRJ	LS
A2321	$$A$SUP1 (SGTCHS)	CG
AACT	$$ANERRU	MG
AACTION	$$ANERRP	LZ
ABTRANS	$$A$SUP1 (SGDFCH)	EA
ACTIONA	$$ANERRP	LZ
ADDLST	$$BTERM	RF

The pointer from FAVP byte, which was pointing to the first available JIB before this terminating phase began, is put in the chain byte of the last-dequeued JIB (using register 8 as an intermediate storage). The second byte of the LUB has a pointer to the first JIB associated with that LUB; this pointer is now put in the FAVP byte.

ADDRLP	$$BATTNH	NT
AFTTIO	$$A$SUP1 (SGUNCK)	DF
ALLBND	$$A$SUP1 (FOPT)	BC
ALLOC	$$BATTNE	NM
ALT	$$BATTNI	NW
ALTER	$$BDUMPF	RZ
	$$BDUMPB	SE
	$$BDUMPD	SJ
	$$BPDUM1	SP

Switch to enter or bypass SPECIAL routine that blanks printing of the first two storage data words. To illustrate the use of this SPECIAL routine, consider the example where the beginning address of a problem program or parameter dump falls between 3F8 and 3FF. To begin print of the dump at the nearest lower double-word boundary, it is necessary to blank out data from 3F0 through 3F7.

In the case of a parameter dump, an additional calculation is made to determine the number of additional blanks needed, if the desired starting address is 3FC. This number is put in register 2 by the $$BPDUMP monitor phase and passed to the phase actually performing the dump. This switch is, therefore, a NOP only once (if needed) at the outset of the problem program portion of a dump or a parameter dump, and is normally set to a branch.

ALTER1	$$BDUMPF	RZ
	$$BDUMPB	SE
	$$BDUMPD	SJ
	$$BPDUM1	SQ

Routine that puts an extra 2 spaces between groups of 4 words, making a total of 3 spaces. This makes the dump easier to read, because storage locations such as 1B0, 1C0, 1D0, etc. stand out clearly. The word counter, register 0, used for this grouping function, is reset to 4.

ALTER2	$$BDUMPF	RZ
	$$BDUMPB	SE
	$$BDUMPD	SJ
	$$BPDUM1	SQ

This routine increments register 6, which points to locations along the print line where data information is being assembled. It is incremented by 9 for each new word to be printed: one for the space between words and 8 for the print positions of each unpacked word.

ALTER3	$$BDUMPF	RZ
	$$BDUMPB	SE
	$$BDUMPD	SJ

Switch to enter or bypass instructions that create 2 blank spaces between the location counter and first word of storage data. Switch is set to branch, except when preparing the first word of each new print line.

ANAERR	$$BATTNC	NG
ANNUL	$$ANERR9	MY
ANYERRS	$$ANERR9	MY
ARCANCEL	$$BEOJ	RA
ASGCHG	$$BATTNP	QN
ASGLST	$$BATTNJ	PG
ASGPUB	$$BATTNI	NW
ASSGN	$$BATTNI	NV
ASSGNLOG	$$BATTNG	NS

Label	Phase	Chart
ATABLE	$$BATTNE	*

Defines a set of internal allocation tables built by the ALLOC processor. Because of this label's physical placement, it also defines the internal allocation table for background programs. To establish the table, the ALLOC processor uses existing limit information, found in the PIB, and allocation information acquired from the ALLOC statement. After the allocation data has been validity checked and posted to the allocation table, it is used to update the appropriate PIB entry. The table expansion is:

1. A halfword of padding for proper alignment. (This field can contain a constant.)

2. A halfword containing the current number of 2K blocks.

3. A fullword containing the save area address.

4. A fullword containing the lower limit address.

5. A fullword containing the upper limit address.

ATHRTN	$$A$SUP1 (SGTCON)	HA
ATNCNL	$$A$SUP1 (SGSVC)	FE
ATTNH	$$BATTNH	NT
ATYPE	$$ANERRN	LX
	$$ANERRV	MJ
	$$ANERRX	MP
B	$$BEOJ2	RL
B01	$$ANERRG	LM
B02	$$ANERRG	LM
B050	$$ANERRI	LR
BAL	$$BPDUM1	SM
BAL1	$$BDUMP	RW
	$$BDUMPD	SG
	$$BPDUMP	SL

In $$BDUMP and $$BDUMPD: Routine that blanks out initializing instructions of this phase so this portion of storage can be used as an I/O output area.

In $$BPDUMP: Sense data is tested for file-protect condition if SYSLST is a tape drive. If it is protected, the dump cannot be taken, and this program phase returns to supervisor for selection of next task. If not protected, B-transient $$BPDUM1 is fetched to perform the actual parameter dump.

BALR14	$$BEOJ1	RK
	$$BPSW	RS
BATCH	$$BATTNG	NR
BGCALC	$$BATTNG	NS
BGJOB	$$BDUMPD	SG
BJFTST	$$ANERR1	MW
BLANKS	$$BDUMPB	SE
	$$BDUMPD	SJ

Blanks are used to blank out the unneeded high-order positions of the printline area when the registers and user's part of the communications region are printed.

BLNK2	$$BDUMPF	RZ
	$$BDUMPD	SJ
BLNKLD	$$BATTND	NL
BLNKST	$$BDUMPB	SE
	$$BDUMPD	SJ

Switch that determines if BLANKS instruction will be used. Switch set to branch except under conditions given in the BLANKS label.

BSOT	$$ANERRE	LF
BSOUT	$$ANERRW	ML
BSTTST	$$A$SUP1 (SGUNCK)	DA
BTLOOP	$$BATTNA	NB

Beginning of a table lookup, in the branch vector table, that finds the appropriate B-transient required for further processing.

BTOFWS	$$ANERRF	LJ
BTOTCL	$$ANERRF	LJ
BTSTCR	$$BDUMPF	SA
	$$BDUMPD	SK
	$$BPDUM1	SN

Branch and link to TSTCOR subroutine is followed by comparing characters of the next line to be printed with those of the line just printed. If the next line is identical, a switch is set to branch to the CLRLIN routine that suspends printing the identical line and prints---SAME---instead.

BUS	$$ANERRX	MP
BUS1	$$ANERRV	MJ
BUS2	$$ANERRV	MJ
BUSOUT	$$ANERRA	LB
	$$ANERRV	MJ
	$$ANERRW	MK
	$$ANERR9	MX
BYCNL	$$ANERRZ	MT
C	$$BEOJ2	RL
C01	$$ANERRG	LN
	$$ANERRH	LQ
	$$ANERRJ	LS
C0100	$$ANERRV	MJ
C02	$$ANERRV	MH
CALFET	$$A$SUP1 (SGSVC)	FD

*Listing only.

Label	Phase	Chart
CALLMW	$$ANERRX	MN
CALLSEC2	$$ANERRZ	MS
CALSEC3	$$ANERRY	MR
CALPH4	$$ANERRO	LY
CALPH5	$$ANERRP	LZ
CALPHS3	$$ANERRN	LW
CANCEL	$$ANERR1	MW
	$$BATTNC	NG
CANCLB	$$BATTNC	NG
CANCOD	$$ANERRS	MD
CANCON	$$ANERRX	MP
CANRTN	$$ANERRS	MD
CANTLP	$$ANERRY	MQ
CASERR	$$BATTNI	NV
	$$BATTNP	QM
CAUSE1	$$BEOJ1	RJ
	$$BEOJ2	RL
CCBSTR	$$ANERRM	LV
CCBUNAV	$$ANERRN	LW
CCBQED	$$A$SUP1 (SGUNCK)	DB
CDATACK	$$ANERR9	MX
CDC	$$ANERRW	ML
CEDETST	$$A$SUP1 (SGTCHS)	CK
CHAIN	$$BTERM	RF
CHAINCH	$$ANERRB	LC
CHDATCH	$$ANERRE	LF
CHDTCK	$$ANERRX	MN
CHECK	$$ANERRF	LH
CHEND	$$A$SUP1 (SGTCHS)	CK
CHEND1	$$A$SUP1 (SGTCHS)	CK
CHFAIL	$$A$SUP1 (SGUNCK)	DF
CHFAIL1	$$A$SUP1 (SGUNCK)	DF
CHGSTT	$$BATTNF	NQ
CHKADR	$$ANERRT	ME
	$$ANERRW	MK
CHKAM	$$ANERRB	LC
CHKASG	$$BATTNM	QD
CHKCNT	$$ANERRL	LL
	$$ANERRV	MJ
CHKDEV	$$ANERRP	LZ
CHKDISK	$$ANERRA	LA
CHKF1	$$BATTNJ	PE
CHKF2	$$BATTNJ	PE
CHKFGA	$$BATTNH	NU
CHKFUA	$$BATTNI	PD
CHKIS	$$ANERRE	LG
CHKJIB	$$BATTNI	NZ
CHKLTA	$$ANERRS	MD
CHKMOD	$$BATTNI	NV
CHKNXC	$$BATTNI	PB
	$$BATTNK	PL
CHKOWN	$$BATTNN	QE
	$$BATTNP	QM
CHKPRN	$$BATTNE	NM
CHKPUB	$$BATTNI	NY
CHKRNG	$$BATTNE	NN
	$$BATTNI	PB
	$$BATTNK	PL
CHKRNG1	$$BATTNK	PL
CHKRT	$$ANERRE	LF
CHKSLH	$$BATTNK	PK
CHKSTT	$$BATTNA	NB
CHKUA	$$BATTNJ	PE
CHNDRT	$$A$SUP1 (SGTCHS)	CM
CHNTST	$$A$SUP1 (SGTCHS)	CL
CHQOVFLW	$$ANERRY	MQ
CKBF12	$$BATTNC	NG
CKF1F2	$$BATTNB	NE
CKNDAR	$$BATTNF	NQ
CKNDCH	$$BATTNI	NW
CKNXJB	$$BATTNI	NZ

Exit point to the scan JIB subroutine, SCNJIB. The subroutine is entered to reset JBSLUB according to any JIB chained to the logical unit.

CKPBUA	$$BATTNJ	PE
CKSCST	$$BATTNE	NM
CLCEX	$$A$SUP1 (SGDFCH)	EA
CLCINS	$$A$SUP1 (SGTCHS)	CG
CLI	$$BEOJ3	RD
CLRCCB	$$ANERRS	MD
CLRLIN	$$BDUMPF	SA
	$$BDUMPB	SF
	$$BDUMPD	SK
	$$BPDUM1	SN
CLRTEB	$$A$SUP1 (SGTCHS)	CK
CMDCHN	$$A$SUP1 (SGUNCK)	DA
CMDREJ	$$ANERRV	MJ
	$$ANERR9	MY
CMDSEQ	$$ANERRV	MH
CMNWLM	$$BATTNF	NP
CMPBPT	$$BATTNI	NZ

Test for identical PUB pointers. Identical PUB pointers indicate that another LUB is assigned to the physical unit pointed to by the LUB just unassigned. (See label UNPA in this list.) If there is no other LUB with a matching PUB, the ownership flag of the PUB indicated by the LUB in LBSLUB is reset so that the PUB is not assigned to any LUB.

Label	Phase	Chart
CMPCOR	$$BDUMPF	SA
	$$BDUMPB	SF
	$$BDUMPD	DK
	$$BPDUM1	SP

Register 5 contains the highest storage location that prints for any single line. Register 5 is compared to register 8 (which contains the upper storage limit of the dump) to see if limit of dump will be exceeded should the entire line be printed. If register 5 is higher than register 8, the value in register 8 is then loaded into register 5 and the printing ceases at the dump limit.

Label	Phase	Chart
CMR	$$ANERRV	MJ
CNCL	$$A$SUP1 (FOPT)	BC
	$$ANERRU	MG
CNCLIN	$$BATTNC	NG
CNCLLOOP	$$ANERRZ	MT
CNCLME	$$BATTNC	NG
CNCLMSK	$$ANERRY	MR
CNCLSW	$$ANERRY	MR
CNCLTEST	$$BEOJ	RB
CNLRTN	$$BATTNC	NG
CNLSVE	$$A$SUP1 (FOPT)	BA
CNO	$$ANERRU	MF
CNTEXT	$$ANERRP	LZ
CNTREX	$$ANERRD	LE
CNTRTN	$$ANERRD	LE
CNTST	$$ANERRE	LF
	$$ANERRN	LW
	$$ANERRV	MH
CNUNCO	$$BATTNK	PN
	$$BATTNM	QD
CNUNCO1	$$BATTNK	PN
CNVBCD	$$BATTND	NL
COMBIN	$$ANERRA	LA
COMM	$$BDUMPB	SC
	$$BDUMPD	SG
COMPCLC	$$A$SUP1 (SGTCHS)	CA
COMR	$$ANERRX	MN
COMREJ	$$ANERRB	LC
COMRJ	$$ANERRW	MM
CONCAT	$$BATTNK	PN

Entry point to a subroutine used to:

1. Read the second half of a statement.

2. Join the first and second parts of a statement forming a single statement. (This operation is called concatenation.)

3. Reset the address of the operand in the I/O area named BUFFER.

4. Reset the length of the operand.

Label	Phase	Chart
CONT	$$ANERRV	MJ
	$$ANERRW	ML
CONTINUE	$$ANERR9	MY
CONTROL	$$BATTNA	NB
CONTROL1	$$BATTNA	NB
CONTSCAN	$$BTERM	RG
CONTX	$$ANERRE	LF
CORCHN	$$A$SUP1 (SGTCHS)	CJ
CORE	$$BDUMPF	RX
	$$BDUMPB	SC
	$$BDUMPD	SG

Register 7, containing the beginning storage address of the problem program area, is tested for proper boundary alignment. If register 7 is not on a boundary that is a multiple of 16, it is adjusted to a boundary such as 1B0, 1C0, 1D0, etc., and the switch at ALTER is set to NOP. See label ALTER.

Label	Phase	Chart
CORE1	$$BDUMPF	RX
	$$BDUMPB	SC
	$$BDUMPD	SG
CORE2	$$BDUMPF	RX
	$$BDUMPB	SC
CORE3	$$BDUMPF	RX
	$$BDUMPB	SC
	$$BDUMPD	SG
CORPUB	$$A$SUP1 (SGTCHS)	CJ
CORREC	$$A$SUP1 (SGTCHS)	CH
COUNTRG	$$ANERRP	LZ
	$$ANERRQ	MA
CQDSP	$$A$SUP1 (SGTCHS)	CN
CRC	$$ANERRF	LJ
CRTBLD	$$BATTNE	NM
CSWCHK	$$A$SUP1 (SGUNCK)	DF
CUAPNX	$$BATTNJ	PE
CUU	$$ANERRO	LY
CYLEND	$$A$SUP1 (SGDSK)	GB
D	$$BEOJ2	RL
DASD2321	$$A$SUP1 (SGTCHS)	CG
DAT	$$ANERRX	MN
DATAADDR	$$A$SUP1 (SGDFCH)	EC
DATACHK	$$ANERRB	LC
DATCHK	$$ANERRV	MJ
DATE	$$BPDUM1	SP
DCERP	$$ANERRA	LA
DEALSO	$$A$SUP1 (SGTCHS)	CM
DECCTR	$$ANERRQ	MB
DECHQ	$$A$SUP1 (SGTCHS)	CL
DEQUE	$$A$SUP1 (SGSVC)	FD
DEQUER	$$A$SUP1 (SGUNCK)	DE

Label	Phase	Chart
DEQUER1	$$A$SUP1 (SGUNCK)	DE
DEQUEUE	$$BTERM	RF

The JIB pointer from the LUB is temporarily stored at label FRLSTBEG. The JIB pointed at by the LUB is addressed, and its first 3 bytes are zeroed. The chain byte (4th byte) of the JIB is checked for additional JIBs in the chain; if there are any, the first 3 bytes of these JIBs are zeroed until the end of the chain is reached.

DEQUEUED	$$BTERM	RF
DISKRT	$$BDUMP	RV
	$$BPDUMP	SL
DISWHY	$$A$SUP1 (SGTCHS)	CJ
DKTYPE	$$BEOJ	RC
	$$BEOJ1	RK
	$$BEOJ2	RM
	$$BILSVC	RP
	$$BPSW	RS
	$$BPCHK	RT
DLAB	$$BATTNL	PS
DLBL	$$BATTNK	PQ
DLBOUT	$$BATTNK	PQ
	$$BATTNL	PS
DNEERR	$$BATTNI	NY
DOERG	$$ANERRL	LL
DONE	$$BTERM	RE
DOP34	$$BATTNL	PT
DOSVC	$$ANERRD	LE
	$$ANERRE	LF
	$$ANERRF	LK
	$$ANERRL	LL
DOSVCA	$$ANERRL	LL
DSKTST	$$A$SUP1 (SGDSK)	GA
DTCH	$$ANERRB	LC
DTCHAT	$$BATTNA	NA
DTCHSZ	$$BATTNA	NA
DTCK	$$ANERRL	LL
DTCRTN	$$ANERRD	LD
DTINUN	$$BATTNA	NC
DTSTCR	$$BDUMPB	SF
DUMP	$$BDUMPD	SG
E050	$$ANERRI	LR
E051	$$ANERRI	LR
E06	$$ANERRI	LR
E062	$$ANERRI	LR
EDRDHA	$$A$SUP1 (SGDSK)	GB
EDRDHA1	$$A$SUP1 (SGDSK)	GB
EDTIC	$$A$SUP1 (SGDSK)	GB
EDTIC1	$$A$SUP1 (SGDSK)	GB
END	$$ANERR0	MV
	$$ANERR1	MW
ENDLIN	$$BDUMPF	SA
	$$BDUMPB	SF
	$$BDUMPD	SK
	$$BPDUM1	SN
ENDLIN1	$$BDUMPF	SA
	$$BDUMPD	SK
	$$BPDUM1	SN
ENDPUB	$$BTERM	RE
ENDPUBS1	$$ANERRZ	MS
ENDPUBS3	$$ANERRZ	MT
ENT1	$$BEOJ	RB
ENT2	$$BEOJ	RB
ENT3	$$BEOJ	RB
ENTEXT	$$A$SUP1 (FOPT)	BD
	$$A$SUP1 (SGSVC)	FJ
ENTIO	$$A$SUP1 (FOPT)	BD
ENTPCK	$$A$SUP1 (FOPT)	BA
	$$A$SUP1 (SGSVC)	FG,FH
ENTSVC	$$A$SUP1 (SGSVC)	FA
EOJS1	$$BEOJ	RB
EOJSTEP	$$BEOJ	RB
EOVMV	$$BPDUM1	SQ
EQPCHK	$$ANERRU	MG
	$$ANERRV	MH
EQUCHK	$$ANERRX	MP
EQUIP	$$ANERRA	LA
EQUIPXIT	$$ANERR9	MX
EREV	$$ANERRD	LE
	$$ANERRF	LK
	$$ANERRL	LL
ERG	$$ANERRD	LE
ERGERR	$$ANERRL	LL
ERGRET	$$ANERRF	LH
ERR	$$ANERRD	LE
	$$ANERRE	LG
	$$ANERRF	LK
	$$ANERRL	LL
ERR20	$$ANERRA	LA
ERRGO	$$A$SUP1 (FOPT)	BB
ERROVL	$$A$SUP1 (SGUNCK)	DB
ERRPRT	$$A$SUP1 (SGUNCK)	DD
ERRRTN	$$BATTNA	NC
ERRRTN1	$$BATTNH	NU
ERRSEN	$$A$SUP1 (SGUNCK)	DB
ERRSET	$$A$SUP1 (FOPT)	BB
ERRSETO	$$A$SUP1 (FOPT)	BB
ERRSIO	$$A$SUP1 (SGUNCK)	DF
ERRSIO2	$$A$SUP1 (SGUNCK)	DF
ERRTYP	$$ANERRP	LZ
EXCAN	$$A$SUP1 (SGSVC)	FE

Label	Phase	Chart
EXCP1	$$A$SUP1 (SGTCHS)	CA,CB
EXCP10	$$A$SUP1 (SGTCHS)	CA
EXCP2	$$A$SUP1 (SGTCHS)	CB
EXCP3	$$A$SUP1 (SGTCHS)	CC
EXCP4	$$A$SUP1 (SGTCHS)	CA
EXCP5	$$A$SUP1 (SGTCHS)	CA,CB
EXCP6	$$A$SUP1 (SGTCHS)	CA,CB
EXCP7	$$A$SUP1 (SGTCHS)	CA,CB
EXCP8	$$A$SUP1 (SGTCHS)	CA
EXCPIGN	$$A$SUP1 (SGTCHS)	CA,CB
EXCPRG	$$BATTNA	NC
EXEC	$$BATTNM	PY

Entry point to the execute (EXEC) processor. This phase is the last processing phase of the foreground initiator. The foreground program will be loaded when this phase has finished executing and when the foreground program has been chosen by the task selection mechanism of the supervisor.

Label	Phase	Chart
EXEC1	$$BATTNM	PZ
EXIGN	$$A$SUP1 (SGUNCK)	DD
EXINTR	$$ANERRR	MC
EXIT	$$ANERRD	LE
	$$ANERRN	LX
	$$ANERRS	MD
	$$ANERRZ	MS
	$$ANERR1	MW
	$$BPDUMP	SL
	$$BPDUM1	SN
	$$BSYSWR	TA
EXIT1	$$A$SUP1 (SGDFCH)	ED
	$$ANERR0	MV
EXIT2	$$ANERR0	MV
EXITA	$$ANERRA	LB
	$$ANERRL	LL
	$$ANERRS	MD
EXITAB	$$ANERRB	LC
EXITB	$$ANERRA	LB
	$$ANERRS	MD
EXITBB	$$ANERRB	LC
EXITC	$$ANERRA	LB
EXPAND	$$A$SUP1 (SGDFCH)	EB
EXRTY	$$A$SUP1 (SGUNCK)	DD
EXT01	$$A$SUP1 (FOPT)	BC
EXT02	$$A$SUP1 (FOPT)	BC
EXT03	$$A$SUP1 (FOPT)	BC
EXT04	$$A$SUP1 (FOPT)	BC
EXT1	$$A$SUP1 (SGSVC)	FJ
EXT2	$$A$SUP1 (SGSVC)	FJ
EXTEOJ	$$A$SUP1 (FOPT)	BC
EXTIN1	$$BATTNB	NE
EXTNT	$$BATTNO	QF
EXTRAN	$$A$SUP1 (SGUNCK)	DC
EXTRT1	$$A$SUP1 (SGSVC)	FG,FL
EXWHY	$$A$SUP1 (SGUNCK)	DD
F01	$$ANERRG	LN
F010	$$ANERRG	LN
F02	$$ANERRG	LN
F03	$$ANERRG	LN
F1CALC	$$BATTNG	NS
F1TBEN	$$BATTNA	*

Internal allocation table for foreground 1 programs. (See label ATABLE.)

Label	Phase	Chart
F2CALC	$$BATTNG	NS
F2CALC1	$$BATTNG	NS
F2CH	$$BDUMPD	SG
F2TBEN	$$BATTNA	*

Internal allocation table for foreground 2 programs. (See label ATABLE.)

Label	Phase	Chart
FCH3	$$A$SUP1 (SGSVC)	FD
FCHOVL	$$A$SUP1 (SGUNCK)	DC
FCHRT1	$$A$SUP1 (SGDFCH)	EA
FCHRT2	$$A$SUP1 (SGDFCH)	EA
FCHRT3	$$A$SUP1 (SGDFCH)	EA
FCHTP	$$ANERRA	LB
FDDKCODE	$$BATTNK	PQ
	$$BATTNL	PS
FDEOJ	$$BDUMPF	SA
FDK1	$$BATTNK	PQ
FDK2	$$BATTNK	PQ
FDKIJ1	$$BATTNK	PQ
FDKIJ2	$$BATTNK	PQ
FDKTDAT	$$BATTNK	PP
FDKTDAT1	$$BATTNK	PP
FDKTDAT2	$$BATTNK	PP
FDKTID	$$BATTNK	PP
FDKTV	$$BATTNK	PK
FDKTV1	$$BATTNK	PK
FDKTV2	$$BATTNK	PK

*Listing only.

Label	Phase	Chart
FDKTVNM	$$BATTNK	PK
FEBIN1	$$BATTNO	QJ
FESEQ	$$BATTNO	QJ
FESEQCK	$$BATTNO	QG
FESPLIT	$$BATTNO	QH
FESPLIT1	$$BATTNO	QH
FESPLIT2	$$BATTNO	QH
FESPLIT4	$$BATTNO	QH
FESYSX2	$$BATTNO	QF
FESYSX3	$$BATTNO	QF
FETCH	$$BDUMP	RW
FETINSR0	$$BATTNK	PM
FETINSR1	$$BATTNK	PM
FETINSRT	$$BATTNK	PM
FETK	$$BATTNO	QG
FETKNO	$$BATTNO	QG
FETKNO1	$$BATTNO	QH
FETKTK	$$BATTNO	QK
FETKTK1	$$BATTNO	QK
FETKTK2	$$BATTNO	QK
FETMSGW	$$ANERR9	MY
FETYPE	$$BATTNO	QF
FETYPE1	$$BATTNO	QG
FETYPED	$$BATTNO	QG
FETYPEI	$$BATTNO	QG
FETYPEI4	$$BATTNO	QG
FEVSER	$$BATTNO	QF
FEVSER1	$$BATTNO	QF
FEVSER3	$$BATTNO	QF
FEVSER4	$$BATTNO	QF
FG1	$$BTERM	RG
FGHED	$$BPDUM1	SP
FGJOB	$$BEOJ1 $$BEOJ2 $$BILSVC $$BPSW $$BPCHK $$BDUMP	RJ RM RN RR RT RW
FGLST	$$BEOJ $$BEOJ1 $$BEOJ2 $$BILSVC $$BPSW $$BPCHK	RC RK RM RP RS RU
FGNAME	$$BPDUM1	SP
FGTAPE	$$BILSVC	RP
FGTST	$$BPDUM1	SM
FILEPR	$$ANERRB	LC
FILPR	$$ANERRL	LL
FINDPUB1	$$ANERRZ	MS
FINISH	$$BATTNM	QB
FIX	$$BPDUM1	SQ

When word counter reaches zero, two extra blanks are inserted between words so that locations such as 1B0, 1C0, 1D0, etc., will stand out, thus making the dump easier to read.

FIX7	$$BPDUMP	SL
FLGBYO	$$BATTNA	*

Defines a byte of program switches:
Bit 0 = 1: Input is from SYSRDR.
Bit 1 = 1: A read has been issued.
Bits 2 - 7: Not used.

FLPTR	$$A$SUP1 (SGTCHS)	CC
FNDQUE	$$A$SUP1 (SGTCHS)	CC
FNDQUE1	$$A$SUP1 (SGTCHS)	CC
FONTERR	$$ANERR9	MY
FPSW	$$BDUMPF $$BDUMPB	RX SC
FREDEV1	$$A$SUP1 (SGTCHS)	CL
FSCAN	$$BATTNK $$BATTNO	PM QJ
FSCAN1	$$BATTNK $$BATTNO	PM QJ
FTCH2	$$ANERRD	LD
FTCH3	$$ANERRD	LE
FTEND	$$BATTNK	PM
FTHMSGW	$$ANERRU	MG
FTMON	$$ANERRE	LG
FTMSW	$$ANERRE	LF
G01	$$ANERRG	LN
GARY	$$BEOJ3	RD
GEN1	$$A$SUP1 (FOPT)	BD
GEN2	$$A$SUP1 (FOPT)	BD
GENENT	$$A$SUP1 (FOPT)	BD
GETBYTE	$$BTERM	RG
GETCHQ	$$A$SUP1 (SGTCHS)	CJ
GETENTRY	$$A$SUP1 (SGDFCH)	EB
GETJIB	$$A$SUP1 (SGTCHS)	CG
GETKEY	$$BATTNI	PD
GETMPSST	$$BDUMPB	SC
GETMSG	$$ANERRV	MJ
GETNXT	$$BTERM	RG
GETPIB	$$A$SUP1 (SGTCHS)	CN
GETPIB1	$$A$SUP1 (SGTCHS)	CN
GETSEN	$$A$SUP1 (SGUNCK)	DB
GIOADR	$$A$SUP1 (SGTCHS)	CE,CF
GO	$$BEOJ3	RD
GOBCK	$$ANERRE	LG
GPR	$$BDUMPB	SC
GTNXJB	$$BATTNI	NZ

Continues search for LUBs with a PUB pointer that matches the pointer in LBSLUB. However, the search is within the JIB table.

*Listing only.

Label	Phase	Chart
GTNXLB	$$BATTNI	NZ

Start of a repetitive sequence of code to get each LUB of a given class and compare its PUB pointer with the PUB pointer of the LUB in LBSLUB.

Label	Phase	Chart
GTNXOP	$$BATTNE	NM
H01	$$ANERRG	LN
	$$ANERRV	MJ
HALT	$$A$SUP1 (SGTCHS)	CD
HALTIO	$$BEOJ3	RD
HARDWT	$$A$SUP1 (SGUNCK)	DC
HERE	$$BSYSWR	TA
HEXCON	$$BATTNI	PB
HOLD	$$BATTNP	QN

Entry point to the HOLD processor. This routine sets a switch in the appropriate PIB assign flag. This switch can be interrogated later by the job control program.

Label	Phase	Chart
HOLDQUE	$$ANERRY	MQ
IDSERR	$$BATTNI	NY
IGEXIT	$$ANERRS	MD
	$$ANERRT	ME
	$$ANERRW	MK,ML
IGNORE	$$BATTNC	NH
IGON	$$ANERRU	MF
IJBEJ22	$$BEOJ	RA
IJBER920	$$ANERR9	MX
IJBERA23	$$ANERRA	LA
IJBERB20	$$ANERRB	LC
IJBERC23	$$ANERRC	LC
IJBERD20	$$ANERRD	LD
IJBERE20	$$ANERRE	LF
IJBERF24	$$ANERRF	LH
IJBERG20	$$ANERRG	LM
IJBERH20	$$ANERRH	LQ
IJBERI20	$$ANERRI	LR
IJBERJ24	$$ANERRJ	LS
IJBERK20	$$ANERRK	LU
IJBERL20	$$ANERRL	LL
IJBERN24	$$ANERRN	LW
IJBERO20	$$ANERRO	LY
IJBERP20	$$ANERRP	LZ
IJBERQ24	$$ANERRQ	MA
IJBERR20	$$ANERRR	MC
IJBERS22	$$ANERRS	MD
IJBERT20	$$ANERRT	ME
IJBERU24	$$ANERRU	MF
IJBERV20	$$ANERRV	MH
IJBMVC20	$$ANERR1	MW
IJBPA120	$$ANERRZ	MS
IJBPA224	$$ANERRY	MQ
IJBPA320	$$ANERR0	MU
INDIB	$$A$SUP1 (SGTCHS)	CH
INDSEQ	$$BATTNL	PU
INHWRITE	$$A$SUP1 (SGTCHS)	CG
INIT	$$BDUMPD	SG
INITCHN1	$$A$SUP1 (SGTCHS)	CL
INITL	$$BATTNM	QC
INITOPR	$$BATTNG	NS
INITRG	$$A$SUP1 (SGTCHS)	CL
INITSIO	$$A$SUP1 (SGTCHS)	CL
INNXEN	$$BATTNE	NM
INRQ	$$ANERRW	ML
INSERT	$$ANERRB	LC
INST	$$ANERRP	LZ
INTERR	$$BEOJ	RA
INTERV	$$ANERR9	MX
INTPUBSC	$$A$SUP1 (SGTCHS)	CJ
INTREQ	$$ANERRU	MG
	$$ANERRV	MH
INTRTN	$$A$SUP1 (SGTCHS)	CJ
INTVEN	$$ANERRA	LB
INVAL	$$ANERRY	MR
IOCOMP	$$ANERRQ	MB
IODONE	$$ANERRP	LZ
IOERR	$$ANERRH	LQ
	$$ANERRI	LR
	$$ANERRJ	LT
	$$ANERRP	LZ
IOINTR	$$ANERRR	MC
IONOP	$$A$SUP1 (SGUNCK)	DA
IOPSET	$$A$SUP1 (SGSVC)	FD
IORTN	$$ANERRI	LR
	$$ANERRJ	LT
IREXIT	$$ANERRT	ME
	$$ANERRW	MK
ISCKSQ	$$BATTNL	PU
ISTYP4	$$BATTNL	PU
ITBRC	$$A$SUP1 (SGTCON)	HA
ITERATE	$$BEOJ3	RD
JBSLUB	$$BATTNI	*
	$$BATTNJ	*
	$$BATTNP	*

A halfword work area primarily used with the scan JIB or scan LUB subroutines. When used with the scan JIB subroutine, this area contains the LUB image information (PUB pointer and JIB pointer) found within a JIB entry. When used with the scan LUB subroutine, this area contains a true LUB entry used within the unassign routine for comparisons.

Label	Phase	Chart
JIBTYP	$$A$SUP1 (SGTCHS)	CH
K01	$$ANERRG	LN
KEYINT	$$ANERRR	MC
KRTY	$$ANERRS	MD
L01	$$ANERRG	LP
	$$ANERRI	LR

*Listing only.

Label	Phase	Chart
LA	$$BILSVC	RP
	$$BPCHK	RU
LABMV	$$BSYSWR	TA
LABMV1	$$BSYSWR	TA
LABMV2	$$BSYSWR	TA
LABMV3	$$BSYSWR	TA
LACSW	$$ANERRO	LY
LANXJB	$$BATTNJ	PG
LASPH	$$ANERRA	LA
LASPHAS	$$ANERRA	LA
LASTPUB	$$BEOJ3	RD
LAXERR	$$BATTNK	PR
	$$BATTNL	PU
	$$BATTNM	PZ
	$$BATTNO	QF
LBLADR	$$BATTNA through $$BATTNP	*

Defines an area that contains the address of the temporary label storage area (Foreground origin plus 1728).

LBLOUT	$$BATTNK	PR

Entry point to the subroutine used to output the label information that has been accumulated in the I/O area, BUFFER. The subroutine:

1. Sets length information in the write and verify CCWs.
2. Determines if space is available on the label track within SYSRES.
3. Updates the disk address if necessary.
4. Checks to ensure label area extents on SYSRES are not exceeded.
5. Sets up the seek address and CCB.
6. Branches to the I/O subroutine (EXCPRG) to write and verify the label information on SYSRES. See Appendix D for format of labels or SYSRES.

LBLOUT1	$$BATTNK	PR
	$$BATTNM	PZ
LBLTYP	$$BATTNO	QL
LBSLUB	$$BATTNI	*
	$$BATTNJ	*
	$$BATTNP	*

Defines an area that contains the LUB entry found by a scan of the LUB table. This label can be described as a parameter passing area.

LBTOUT	$$BATTNO	QL
LDPSW	$$ANERRR	MC
LDREGS	$$A$SUP1 (SGTCHS)	CJ
LGD	$$A$SUP1 (SGTCHS)	CG
LGD1	$$A$SUP1 (SGTCHS)	CG
LGDD	$$A$SUP1 (SGTCHS)	CG
LISTIO	$$BATTNJ	PE
LMERA	$$A$SUP1 (FOPT)	BC
LOAD	$$BDUMPD	SH
LOADPSW	$$ANERRR	MC
LOG	$$BATTNC	NH
	$$BTERM	RH
LOG1	$$ANERRY	MQ
LOG2	$$ANERRY	MQ
LOGENT	$$ANERRP	LZ
	$$ANERRQ	MA
LOGERP	$$ANERRU	MG
LOGEXT	$$BATTNH	NU
LOGGER	$$BATTNH	NU
	$$BATTNJ	PF
	$$BEOJ1	RJ
	$$BEOJ2	RM
	$$BPSW	RR
LOGLIST	$$BEOJ	RC
LOGPRC	$$A$SUP1 (SGTCHS)	CH
LOGPRC1	$$A$SUP1 (SGTCHS)	CH
LOGRTN	$$ANERRY	MQ
LOGTST	$$ANERRU	MG
	$$ANERRY	MQ
LOGWAIT	$$ANERRY	MQ
LOOKUP	$$A$SUP1 (SGDFCH)	EB
LST	$$BDUMPF	RZ
	$$BDUMPB	SE
	$$BDUMPD	SJ
	$$BPDUM1	SQ

Switch used to return from REGPNT subroutine, when last word of a printline has been unpacked and printed, to prepare the next line. For printing of registers and user's communications region, LST is a no-op that permits entry to a routine that blanks out unneeded high-order positions of the printline.

LST1	$$BPDUM1	SQ
LSTASG	$$BATTNJ	PJ
LSTAUN	$$BATTNJ	PG

Entry point to the subroutine that lists the assignments for either F1 or F2 system class and programmer class units. The subroutine sets up primary and secondary headers, calls the LUB scanning subroutine and the JIB scanning subroutine, and calls the final output subroutine.

LSTBG	$$BATTNJ	PF

*Listing only.

Label	Phase	Chart
LSTBGF	$$BATTNJ	PF
LSTBUN	$$BATTNJ	PG
LSTBUN1A	$$BATTNJ	PG
LSTBUN2	$$BATTNJ	PG
LSTDIB	$$A$SUP1 (SGTCHS)	CH
LSTF1	$$BATTNJ	PF
LSTF2	$$BATTNJ	PF
LSTLN	$$BDUMPF	SA
	$$BDUMPB	SF
	$$BDUMPD	SK
	$$BPDUM1	SN

The location counter, register 7, is set and translated to identify the storage locations being printed on each line of the dump. This label is also used to enter the PRNTLN subroutine on a last line condition, thereby bypassing the TSTCOR subroutine.

LSTPRG	$$BATTNJ	PH
LSTPRG1	$$BATTNJ	PH
LSTSTD	$$BATTNJ	PH
LSTUA	$$BATTNJ	PJ

Program switch set to NOP when a UA operand is found. The switch is reset to a branch when the header 'UNASSIGNED' has been printed.

LTA	$$A$SUP1 (SEND)	KA
LTABSY	$$A$SUP1 (SGSVC)	FC
LTACNL	$$A$SUP1 (FOPT)	BB
LTKHLD	$$ANERRZ	MT
M01	$$ANERRG	LP
	$$ANERRI	LR
	$$ANERRK	LU
MACHEK	$$A$SUP1 (SGUNCK)	DF
MAIN1	$$BPDUMP	SL

The starting address for the parameter dump, entered in register 6, is shifted right double logical 4 positions so that any value not a multiple of 16 is now in register 7. If value in register 7 is now zero, it indicates that the starting value in register 6 is on a double-word boundary. Register 6 is then restored by shifting left to the next lower double-word boundary nearest the value specified by the dump parameter (label FIXT). If register 7 was not zero when tested, the value now in it is used to calculate the number of blank print positions needed so printout starts at desired starting byte.

MAIN2	$$BPDUMP	SL

The upper parameter address is incremented by a word length and tested against system's main storage capacity to see if requested dump is a valid address within core. If not, the upper storage limit is put in register 8 to impose a valid dump end limit.

MAINRT	$$ANERRD	LD
	$$ANERRE	LF
MAP	$$BATTND	NJ
MARK	$$BEOJ	RA
MESG	$$ANERRB	LC
MICR	$$BEOJ	RA
MICR1	$$BEOJ	RA
MICRDEV	$$BEOJ	RA
MKASGN	$$BATTNI	NW

Entry point to a routine that makes the actual assignment during ASSGN processing. The assignment is made by:

1. Establishing the PUB pointer in the LUB.

2. Setting the ownership byte in the PUB.

3. Setting the mode byte in the PUB. (For tape devices only.)

MOD	$$BTERM	RE
MODCCW	$$ANERRX	MP
MODRST	$$BATTNI	NV
MOVE	$$BDUMPD	SH

Current address taken from the Disk Information Block (DIB) for the appropriate symbolic disk device is put in output area to serve as the count ID information when count, key, and data are written. The current address record number is then reduced by 1 and put in the search CCW for writing the first dump record.

MOVLOP	$$BATTNM	QB

Start of a repetitive sequence of code to move the last two routines of the EXEC processor to the main storage area occupied by the root phase, $$BATTNA. The root phase resides in the logical transient area of main storage. The two routines are moved 256 bytes at a time. The last time the move is executed, the remaining bytes (less then 256) are moved to the logical transient area.

Label	Phase	Chart
MOVRTN	$$BATTNM	QB

Entry point to the subroutine that:

1. Moves any label information from the temporary label storage area to the label storage area.

2. Clears the remainder of main storage to initialize it for the foreground program being initiated.

MPSTST	$$ANERRN	LW
	$$ANERRQ	MA
MSG	$$BATTNA	NE
MSG2	$$ANERRM	LV
MSG3	$$ANERRM	LV
MSG4	$$ANERRM	LV
MSGPRT	$$ANERRS	MD
MSGWTR	$$ANERRL	LL
	$$ANERRG	LM
	$$ANERRH	LQ
	$$ANERRI	LR
	$$ANERRJ	LS
	$$ANERRK	LU
	$$ANERRT	ME
	$$ANERRV	MJ
	$$ANERRW	MK,ML,MM
MTAPE	$$BTERM	RG

The device type from the PUB table entry for the device is examined. If the device is not a tape drive, the PUB scan proceeds to the next entry in the table; if it is a tape drive, the Tape Error Block (TEB) for that particular drive is addressed and checked for any record of tape errors. If this tape drive has had no errors, the PUB scan resumes, and the next device in the PUB table is investigated.

MTRSVD	$$BATTNK	*
	$$BATTNL	*
	$$BATTNO	*

A one-byte switch used when the file type is sequential disk (SD):

Bit 0 = 1: Look-ahead flag for LIOCS.

Bit 1 = 1: Last extent for file.

Bit 2: Not used.

Bit 3 = 1: No extent dequeue.

Bit 4 = 1: Extent limits omitted.

Bit 5 = 1: Extent limits converted to address.

Bits 6, 7: Not used.

MVBLNK	$$BDUMPD	SJ
	$$BPDUM1	SQ
MVC	$$ANERR1	MW
	$$BEOJ2	RL
	$$BILSVC	RP
	$$BPCHK	RU
MVCCW	$$A$SUP1 (SGTCHS)	CC
MVCLRT	$$BATTNM	QB
MVI	$$BTERM	RE

A detach flag is posted in the PIB for the terminated program. The portion of core occupied by this program is now available for overlay. An End-of-Termination switch is set in the PIBPUBAS flag byte, an SVC 22 releases control of the system from this program, and an SVC 11 returns the system to the Task Selection routine of the supervisor.

MVMSG	$$ANERRV	MH
MVZ	$$BEOJ	RC
MVZEX	$$A$SUP1 (SGUNCK)	DB
N01	$$ANERRG	LP
	$$ANERRJ	LS
N1114	$$A$SUP1 (SGTCHS)	CM
NAMED	$$BEOJ1	RJ
NASERR	$$BATTNI	NW
NDCHFD	$$BATTNI	NW
NDSCAN	$$BATTNA	ND
	$$BATTNJ	PH
	$$BATTNK	PP
NDSCAN1	$$BATTNK	PP
NDTERR	$$BATTNI	NW
	$$BATTNM	QC
	$$BATTNP	QM
NEWXTN	$$BATTNL	PW
	$$BATTNO	QH
NJPERR	$$BATTNI	NW
NLSERR	$$BATTNL	PS
	$$BATTNO	QJ
NLUERR	$$BATTNI	NX
	$$BATTNK	PN
	$$BATTNM	QD
NOCCB	$$ANERRO	LY
	$$ANERRX	MP
NOCCB1	$$A$SUP1 (SGUNCK)	DA
NOCHNG	$$BDUMP	RW

Routine used, when a foreground program is to be dumped, to identify the physical I/O device associated with SYSLST. The type of device determines which B-transient dump program will be fetched to perform the actual dump.

NOCOMP	$$ANERRE	LG
NODCUX	$$BATTNL	PW
	$$BATTNO	QH

*Listing only.

Label	Phase	Chart
NOFG	$$BEOJ	RC
NOLBPR	$$BATTNM	QA
NOLOADAD	$$A$SUP1 (SGDFCH)	EB
NOLOG	$$ANERRN	LX
	$$ANERRP	LZ
	$$ANERRQ	MB
	$$BATTNC	NH
NOLTA	$$ANERRN	LW
NONREC	$$ANERR9	MX
NOP	$$BTERM	RG
	$$BILSVC	RP
	$$BPCHK	RU

In $$ETERM: Switch to enter or bypass the routine that prints headings prior to logging the Tape Error Block (TEB) statistics. Because only one set of headings is needed, this routine is used only for the first TEB statistics logged. Thereafter, this routine is bypassed by making this switch an unconditional branch.

In $$BILSVC and $$BPCHK: After first line of message has been output, this switch is set to branch. The next time through, the second line of the message is output and the branch causes the transient $$BDUMP to be fetched.

NOPBR	$$ANERRQ	MA
NOPINS	$$A$SUP1 (SGTCHS)	CG
NOPINSTR	$$A$SUP1 (SGTCHS)	CG
NOQUIS	$$A$SUP1 (SGUNCK)	DG
NORCD	$$A$SUP1 (SGDSK)	GA
NORCFND	$$ANERRA	LB
NOSO1	$$A$SUP1 (SGTCHS)	CL
NOSTART	$$A$SUP1 (SGUNCK)	DA
NOTASG	$$BATTNJ	PJ
NOTBG	$$BPCHK	RT
NOTBSY	$$A$SUP1 (SGTCHS)	CK
NOTEST	$$BDUMPF	RX
	$$BDUMPB	SB
	$$BDUMPD	SG

An area of storage used for phase initialization instructions is blanked out to be used as an output area for the dump. If needed, a branch is taken past the end of the cleared area to the next instruction.

NOTIC	$$A$SUP1 (SGTCHS)	CG
NOTPD	$$A$SUP1 (SGSVC)	FA
NOTPD1	$$A$SUP1 (SGSVC)	FA
NOTPD2	$$A$SUP1 (SGSVC)	FA
NOTSEQ	$$BATTNL	PU
NOTZERO	$$BDUMPB	SD
NOUNIT	$$BATTNJ	PH
NRFRTY	$$ANERRJ	LS
NULLOG	$$BATTNA	NC
NUMCON	$$BATTNI	PB
	$$BATTNK	PN
	$$BATTNO	QK
NUMLOP	$$BATTNI	PB
NUMSYS	$$BATTNI	PB
NVAERR	$$BATTNF	NP
NVSERR	$$BATTNA	NC
NWPBPT	$$BATTNI	NW
NXPBNT	$$BATTNF	NQ
NXTLUB	$$BATTNJ	PG
OCTEST	$$BEOJ	RA
OK	$$ANERR1	MW
ONERET	$$ANERRG	LM
ONERTRY	$$ANERR9	MX
ONLIST	$$BEOJ	RC
	$$BEOJ1	RJ
	$$BEOJ2	RM
	$$BILSVC	RN
	$$BPSW	RR
	$$BPCHK	RT
OPCLOSE	$$A$SUP1 (SGDFCH)	EA
OPFLAG	$$ANERRN	LX
OPNUMB	$$BATTNA	NC
OPRFL	$$ANERRE	LF
OPRSNT	$$BATTNC	NG
OPTRT1	$$A$SUP1 (SGSVC)	FK
OPTRT2	$$A$SUP1 (SGSVC)	FK
OPVNT	$$ANERRW	ML
OR1287	$$ANERR9	MX
ORERP	$$ANERRA	LA
OTHERS	$$BEOJ	RB
OTSERR	$$BATTNK	PL
	$$BATTNL	PS
	$$BATTNO	QG
OURSIO	$$A$SUP1 (SGUNCK)	DF
OUT	$$BDUMPF	SA
	$$BDUMPB	SF
	$$BDUMPD	SK
	$$BPDUMP	SL

Switch made an NOP when the supervisor portion of dump is completed. During the problem program portion of the dump, the switch permits exit from the dump phase by fetching $$BEOJ when the dump limit is reached.

Label	Phase	Chart
OUT1	$$BDUMPF	SA
	$$BDUMPD	SK

If SYSLST is a tape drive, this switch is set to branch to write a tapemark following the record of the last line of the dump.

Label	Phase	Chart
OUT2	$$BDUMPF	SA
OUTAR	$$BTERM	RE

Entry point to this program phase. The output area address is loaded into a CCW. Register 13 is loaded as a link register to the unassign routine. The partition of the terminated program is identified as F2 or <u>not</u> F2. If F2, the ownership flags are reset in the PUB entries of devices owned by this program.

If the program is not an F2 program, it must be an F1, because the $$BTERM phase is called to terminate foreground programs. The PIB assign flag byte is checked to see if the cancel switch is on, which indicates cancel occurred while in a terminator phase due to an I/O malfunction. To prevent a repetitive cancel-within-cancel loop, a branch is set in the switch at label LOG to suppress further I/O operations.

Label	Phase	Chart
OUTLBL	$$BATTNL	PW
	$$BATTNO	QH
OUTPUT	$$BATTND	NL
	$$BATTNJ	PJ
OVERUN	$$ANERRB	LC
OVERRUN	$$ANERR9	MY
OVLAY	$$BILSVC	RN
OVRN	$$ANERRE	LF
OVRUN	$$ANERRV	MJ
OWNRSH	$$BATTNI	NV
PACKCG	$$BATTNL	PW
	$$BATTNO	QH
PAGHED	$$BDUMPF	RY
	$$BDUMPB	SD
	$$BDUMPD	SH
	$$BPDUM1	SP
PAR40	$$ANERR0	MU
PAR41	$$ANERR0	MU
PAR42	$$ANERR0	MU
PAR43	$$ANERR0	MU
PAR44	$$ANERR0	MU
PAR45	$$ANERR0	MU
PAT	$$ANERRA	LB
PAUSE	$$BATTNC	NH
PAUSE1	$$BATTNC	NH
PAUSE2	$$BATTNC	NH
PCHDIB	$$A$SUP1 (SGTCHS)	CH
PCHKSW	$$A$SUP1 (FOPT)	BA
	$$A$SUP1 (SGSVC)	FG,FH
PCITRT	$$A$SUP1 (SGSVC)	FK
PCTEST	$$BEOJ	RA
PCURTN	$$ANERRW	MK
PDABEX	$$A$SUP1 (SMICR)	JH
PDAGN	$$A$SUP1 (SMICR)	JF
PDALBD	$$A$SUP1 (FOPT)	BD
PDBOCK	$$A$SUP1 (SMICR)	JF
PDBUFL	$$A$SUP1 (SMICR)	JB
PDCCBT	$$A$SUP1 (SGSVC)	FB
PDCCBPT	$$A$SUP1 (SGSVC)	FB
PDCHAN	$$A$SUP1 (SMICR)	JE,JF
PDCLI	$$A$SUP1 (SMICR)	JH
PDCOMP	$$A$SUP1 (SMICR)	JE
PDCSWST	$$A$SUP1 (SMICR)	JD,JF
PDDFLT	$$A$SUP1 (SMICR)	JF
PDERXT1	$$A$SUP1 (SMICR)	JE,JF
PDERXT2	$$A$SUP1 (SMICR)	JE,JF
PDERXT3	$$A$SUP1 (SMICR)	JF
PDETIO	$$A$SUP1 (SMICR)	JD,JF
PDEXIT	$$A$SUP1 (SMICR)	JH
PDEXT1	$$A$SUP1 (SMICR)	JH
PDEXT2	$$A$SUP1 (SMICR)	JH
PDFSTM	$$A$SUP1 (SMICR)	JA
PDHERE	$$A$SUP1 (SMICR)	JB
PDIAGN	$$A$SUP1 (SMICR)	JA
PDIAGN1	$$A$SUP1 (SMICR)	JC
PDINTR	$$A$SUP1 (SMICR)	JA
PDKEY	$$A$SUP1 (SMICR)	JH
PDMOV	$$A$SUP1 (SMICR)	JE
PDMPI	$$A$SUP1 (SMICR)	JA
PDMVCSW	$$A$SUP1 (SMICR)	JE
PDNF	$$A$SUP1 (SMICR)	JB
PDNOB	$$A$SUP1 (SMICR)	JC

Label	Phase	Chart
PDNONOP	$$A$SUP1 (SMICR)	JA
PDNOPCK	$$A$SUP1 (FOPT) $$A$SUP1 (SGSVC)	BA FG,FH
PDNOSIO	$$A$SUP1 (SMICR)	JH
PDNOSS	$$A$SUP1 (SMICR)	JB
PDNOTSS	$$A$SUP1 (SMICR)	JG
PDNOTZR	$$A$SUP1 (SMICR)	JB
PDPDEQ	$$A$SUP1 (SMICR)	JE
PDPRCAN	$$A$SUP1 (SMICR)	JG
PDPROGCK	$$A$SUP1 (SMICR)	JG
PDREJDC	$$A$SUP1 (SMICR)	JF
PDREJT	$$A$SUP1 (SMICR)	JC
PDSCAW	$$A$SUP1 (SMICR)	JB
PSDERR	$$A$SUP1 (SMICR)	JD
PDSERR1	$$A$SUP1 (SMICR)	JD,JF
PDSERR2	$$A$SUP1 (SMICR)	JD
PDSERR3	$$A$SUP1 (SMICR)	JD
PDSIO	$$A$SUP1 (SMICR)	JC
PDSIORT	$$A$SUP1 (SMICR)	JD
PDSNO1	$$A$SUP1 (SMICR)	JE
PDSNO2	$$A$SUP1 (SMICR)	JD
PDSPK	$$A$SUP1 (SMICR)	JB
PDSS	$$A$SUP1 (SMICR)	JC
PDSUPEXT	$$A$SUP1 (SMICR)	JH
PDSUPV	$$A$SUP1 (FOPT) $$A$SUP1 (SGSVC)	BD FJ
PDSVC28	$$A$SUP1 (SMICR)	JC
PDTASK	$$A$SUP1 (SMICR)	JH
PDTERR1	$$A$SUP1 (SMICR)	JE,JF
PDTERR2	$$A$SUP1 (SMICR)	JE
PDTERR4	$$A$SUP1 (SMICR)	JE
PDTERR5	$$A$SUP1 (SMICR)	JE
PDTIMER	$$A$SUP1 (SMICR)	JH
PDTIO	$$A$SUP1 (SMICR)	JA
PDTIOERR	$$A$SUP1 (SMICR)	JE,JF
PDTIOK	$$A$SUP1 (SMICR)	JA
PDTKY	$$A$SUP1 (FOPT) $$A$SUP1 (SGSVC)	BD FJ
PDUMP	$$BPDUM1	SM
PDUMP1	$$BPDUM1	SM
PDUMP2	$$BDUMPD	SG

Switch set to branch if it is a parameter dump. Bypasses printout of all parts of core, except the area specified in the parameter limits.

Label	Phase	Chart
PDUPDT	$$A$SUP1 (SMICR)	JA
PDYES	$$A$SUP1 (SMICR)	JE
PHASE	$$A$SUP1 (SGDFCH)	EA
PHASE1	$$ANERRG $$ANERRM	LN LV
PHASEH	$$ANERRG	LP
PHASEJ	$$ANERRG	LM
PHASES	$$ANERRP $$ANERRQ $$ANERRR	LZ MB MC
PHASR	$$ANERRQ	MB
PHYSEIZE	$$BTERM	RE

Further I/O operations are disabled, and an SVC 22 is issued that disables multiprogramming and gives this program control over the system to complete its desired functions until another SVC 22 is issued to release control.

Label	Phase	Chart
PLCCB	$$ANERRW	MM
PNFORSVC	$$BILSVC	RN
PNPERR	$$BATTNM	PY
POSTCAN	$$ANERRY	MR
POSTCE	$$A$SUP1 (SGTCHS)	CM
PRCOMPL	$$BTERM	RE
PREFERED	$$A$SUP1 (SGDFCH)	EA
PRGUNT	$$BATTNM	QD
PRINT	$$BEOJ2 $$BILSVC $$BPSW $$BPCHK $$BDUMPF $$BDUMPB $$BDUMPD $$BPDUMP $$BPDUM1	RM RQ RR RU RY SD,SF SH SL SQ

Label	Phase	Chart
PRINT1	$$BDUMPD	SH
Routine that uses PIOCS to seek, search ID equal, write count, key and data, verify, and wait for completion of the I/O operation.		
PRINTER	$$BDUMP	RV
PRNTL	$$BDUMPF	RZ
	$$BDUMPD	SJ
	$$BPDUM1	SQ
PRNTLN	$$BDUMPF	SA
	$$BDUMPB	SF
	$$BDUMPD	SK
	$$BPDUM1	SN
PRNTLN1	$$BPDUM1	SN
PROC	$$ANERRT	ME
PROCED	$$BPDUMP	SL
PROG	$$BEOJ1	RJ
PROGCHK	$$BEOJ	RB
PROTCHK	$$ANERRA	LA
PROTECT	$$A$SUP1 (SGTCHS)	CG
PRTPRG	$$A$SUP1 (SGUNCK)	DB
PRTPRG1	$$A$SUP1 (SGUNCK)	DA
PSTBUF	$$ANERRW	ML
PSTEOF	$$A$SUP1 (SGTCHS)	CM
PSTERR	$$ANERRT	ME
	$$ANERRW	MK
PSWGET	$$ANERR1	MW
PTERP	$$ANERRA	LA
PUBDEQ	$$A$SUP1 (SGSVC)	FD
PUBSCN	$$ANERRZ	MT
PUBSCN3	$$ANERRZ	MT
PURGE	$$A$SUP1 (SGTCHS)	CK
PUT	$$BEOJ	RC
	$$BDUMP	RN
PUTUTR	$$ANERRU	MG
Q01	$$ANERRG	LP
QIDCK1	$$ANERRZ	MT
QISRT1	$$A$SUP1 (SGUNCK)	DG
QISRT2	$$A$SUP1 (SGUNCK)	DG
QISRT3	$$A$SUP1 (SGUNCK)	DG
QUEUE	$$BEOJ3	RD
QUEID	$$ANERRZ	MT
QUEST	$$BTERM	RG
QUISIO	$$A$SUP1 (SGUNCK)	DG
QUISIO2	$$A$SUP1 (SGUNCK)	DG
QUISIO3	$$A$SUP1 (SGUNCK)	DG
QUISIO4	$$A$SUP1 (SGUNCK)	DG
R01	$$ANERRG	LM
	$$ANERRI	LR
	$$ANERRJ	LS
RADLDS	$$A$SUP1 (SGTCHS)	CA
READBK	$$ANERRF	LJ
RCHAN	$$ANERRD	LE
	$$ANERRE	LG
	$$ANERRF	LK
	$$ANERRL	LL
RCHSC	$$ANERRD	LE
RCVERR	$$A$SUP1 (SGUNCK)	DC
RDDIR2	$$A$SUP1 (SGDFCH)	EB
RDHA9	$$A$SUP1 (SGDSK)	GA
RDRCD	$$ANERRH	LQ
RDSTMT	$$BATTNA	NC
RDTXT	$$A$SUP1 (SGDFCH)	EC
READ	$$BATTNP	QM
READUPDT	$$A$SUP1 (SGDFCH)	ED
RECAL	$$A$SUP1 (SGDSK)	GA
REDFOR	$$ANERRF	LJ
REGPNT	$$BDUMPF	RZ
	$$BDUMPB	SE
	$$BDUMPD	SJ
	$$BPDUM1	SQ
REGPNT1	$$BDUMPF	RZ
	$$BPDUM1	SQ
REGPNT5	$$BDUMPF	RZ
	$$BDUMPB	SE
	$$BDUMPD	SJ
	$$BPDUM1	SQ
REGPNT6	$$BPDUM1	SQ
RELO	$$ANERRL	LL
RELOC	$$ANERRP	LZ
	$$BILSVC	RN
	$$BDUMPB	SB
	$$BPDUM1	SM
RELOCF	$$BDUMPF	RX
RELSE	$$BATTNP	QN
REPBCK	$$ANERRE	LF
REPERR	$$ANERRQ	MB
REPFRW	$$ANERRE	LF
RESCHX	$$A$SUP1 (SGUNCK)	DB
RESERR	$$A$SUP1 (SGDSK)	GA
RESET	$$ANERRN	LX
RESPNS	$$ANERRR	MC
RESVC	$$A$SUP1 (SGSVC)	FC
RETOFF	$$ANERRD	LE
	$$ANERRE	LF
	$$ANERRF	LH
	$$ANERRL	LL
RETRY	$$ANERRD	LD
	$$ANERRF	LH
	$$ANERRR	MC
	$$ANERRV	MJ
	$$ANERRX	MP
	$$ANERR9	MY

Label	Phase	Chart
RETURN	$$BATTNI	NW
RETXIT	$$ANERRX	MP
REVSCN	$$BATTNH	NU
	$$BATTNJ	PF
RLCCB	$$BEOJ	RC
	$$BEOJ2	RM
	$$BILSVC	RP
	$$BPCHK	RT
RNGTOP	$$BATTNI	PB
	$$BATTNK	PL
ROVER	$$ANERRK	LU
RSCH	$$ANERRF	LK
RSETWAIT	$$A$SUP1 (SGTCHS)	CK
RSPPAE1	$$BATTNF	NQ
RSPPEA	$$BATTNF	NQ
RSTCHQ	$$ANERRY	MQ
RSTFLG	$$ANERRI	LR
	$$ANERRJ	LT
	$$ANERRP	LZ
	$$ANERRQ	MA
	$$ANERRU	MG
RSTOWN	$$BATTNI	NZ
RSTPUB	$$A$SUP1 (SGDFCH)	ED
RSTPUB1	$$A$SUP1 (SGDFCH)	ED
RSTQPT	$$ANERRH	LQ
	$$ANERRI	LR
	$$ANERRJ	LT
	$$ANERRP	LZ
	$$ANERRQ	MA
	$$ANERRU	MG
RSTREG	$$A$SUP1 (SGUNCK)	DE
RTY	$$ANERRV	MH
RTY1	$$A$SUP1 (SGDSK)	GA
RTY9	$$A$SUP1 (SGDSK)	GA
RTYCT	$$ANERRV	MJ
RTYONE	$$ANERRU	MG
RTYRTN	$$ANERRP	LZ
RVRSCN	$$BATTND	NL
S01	$$ANERRJ	LS
SAVLUS	$$BATTNJ	PG
SCANR1	$$BATTNA	ND
SCANR2	$$BATTNA	ND
SCANR3	$$BATTNA	ND
SCNJIB	$$BATTNI	PA

Entry point to a subroutine that:

- Initializes JBSLUB with the first and last bytes of the JIB chained to the current pseudo LUB entry of JBSLUB.
- Returns immediately to the calling sequence when an end-of-JIB-chain condition is found.

SCNLBS	$$BATTNI	NZ

Entry point to a subroutine that:

- Returns, sequentially, each LUB entry in a given class to the calling routine.
- Returns immediately to the calling routine when there are no more entries in a given class.

SCNLUB	$$BATTNI	PA
SCNRL1	$$BATTNA	ND
SCNRL2	$$BATTNA	ND
	$$BATTNK	PP
SCURTN	$$ANERRW	ML
SECTION1	$$ANERRZ	MS
SECTION3	$$ANERRZ	MT
SEEKTEST	$$A$SUP1 (SGTCHS)	CG
SEKCH	$$ANERRA	LB
SEKCHK	$$A$SUP1 (SGDSK)	GA
SEKCHK1	$$A$SUP1 (SGDSK)	GA
SELBMX	$$A$SUP1 (SGTCHS)	CL
SELECT	$$A$SUP1 (SGTCHS)	CL
SELERR	$$ANERRV	MH
SENSTSTS	$$ANERR9	MX
SETARON	$$ANERRY	MR
SETCNCL	$$ANERRQ	MA
SETCODE	$$ANERRS	MD
	$$BDUMP	RV
SETDIS	$$ANERRS	MD
SETEXT	$$BATTNE	NF
SETFLG	$$ANERRX	MN
SETLOGUN	$$BEOJ	RC
	$$BEOJ1	RK
	$$BEOJ2	RM
	$$BILSVC	RN
	$$BPSW	RR
	$$BPCHK	RT

Routine that sets logical unit address for SYSLST in CCB after determination of symbolic device to be used for message output.

SETLT1	$$A$SUP1 (SGSVC)	FC
SETLT2	$$A$SUP1 (FOPT)	BB
	$$A$SUP1 (SGSVC)	FC
SETLT2A	$$A$SUP1 (SGSVC)	FC
SETOP1	$$A$SUP1 (SGSVC)	FG,FL
SETOP2	$$A$SUP1 (SGSVC)	FG,FL
SETPOSTL	$$ANERRZ	MT
SETRTRY	$$ANERRQ	MB

Label	Phase	Chart
SETSKAD	$$A$SUP1 (SGDFCH)	EA
SETSVAR	$$A$SUP1 (SGDFCH)	EC
SETSVC	$$ANERRE	LG
SETUP	$$BTERM	RE
SIO	$$A$SUP1 (SGTCHS)	CF
SKCHK	$$ANERRA	LB
SKIP5	$$BDUMPF	RY
SKIPHDR	$$BTERM	RG
SKPLIN	$$BATTND	NL
SKSLI	$$ANERRB	LC
SNGCHG	$$BATTNP	QN
SNGUNA	$$BATTNI	PC
SNS	$$ANERRF	LJ
SPACE	$$BATTNJ	PH
SPECIAL	$$BDUMPF $$BDUMPD $$BPDUM1	RZ SJ SQ

See discussion of this label under ALTER.

Label	Phase	Chart
SPECIAL1	$$BPDUM1	SQ
SSCCB	$$ANERRW	MM
SSCOM	$$ANERRW	ML
SSELER	$$ANERRD	LD
SSMASK	$$ANERRQ	MA
START	$$BATTNG $$BPCHK $$BDUMPF $$BDUMPB $$BPDUM1	NR RT RX SB SM
START1	$$BDUMPF	RX
STARTF	$$BATTNF	NP
STDEXT	$$A$SUP1 (SGUNCK)	DC
STDSWH	$$BATTNJ	PG

Program switch set to branch when stored standard assignments are to be logged. The branch is taken at the end of the JIB table scan. The scan finds any stored standard assignments. The switch is reset at location LSTSTD, Chart PH.

Label	Phase	Chart
STEPLOOP	$$ANERRZ	MT
STEXCD	$$BATTNB	NF
STH	$$BEOJ1 $$BPSW	RK RS
STLLMT	$$BATTNF	NP
STMODE	$$A$SUP1 (SGTCHS)	CF
STORE	$$A$SUP1 (SGTCHS)	CG
STORKE	$$ANERRN	LW
STRTED	$$A$SUP1 (SGTCHS)	CE,CF
STRTIO	$$A$SUP1 (SGTCHS)	CE,CF
STRTIO1	$$A$SUP1 (SGTCHS)	CE,CF
STUBGL	$$BATTND	NK
STUCRL	$$BATTND	NK
STUF1U	$$BATTND	NK
STUSPC	$$BATTND	NK
SUPCNL	$$A$SUP1 (SGUNCK)	DC
SUPEXP	$$A$SUP1 (SGTCHS)	CA,CB
SUPEXT	$$A$SUP1 (FOPT)	BC
SUPPRIO	$$BEOJ	RB

This routine is entered if an abnormal end-of-job condition occurs while transient $$BTERM is executing. An I/O unrecoverable error would cause a cancel of $$BTERM itself, resulting in an unending loop. Therefore, I/O operation is bypassed and $$BTERM is recalled.

Label	Phase	Chart
SUPV	$$BDUMPF $$BDUMPD	RX SG
SUPV1	$$BDUMPF $$BDUMPB $$BDUMPD	RX SC,SF SG
SVC0	$$BEOJ	RC
SVC00	$$A$SUP1 (SGTCHS)	CA,CB
SVC01	$$A$SUP1 (SGSVC)	FB
SVC01A	$$A$SUP1 (SGSVC)	FB
SVC02	$$A$SUP1 (SGSVC)	FC
SVC02A	$$A$SUP1 (SGSVC)	FC
SVC03	$$A$SUP1 (SGSVC)	FD
SVC04	$$A$SUP1 (SGSVC)	FD
SVC05	$$A$SUP1 (SGSVC)	FB
SVC07	$$A$SUP1 (SGSVC)	FD
SVC08	$$A$SUP1 (SGSVC)	FE
SVC09	$$A$SUP1 (SGSVC)	FE
SVC10	$$A$SUP1 (SGSVC)	FE
SVC10A	$$A$SUP1 (SGSVC)	FE
SVC11	$$A$SUP1 (SGSVC)	FC
SVC11A	$$A$SUP1 (SGSVC)	FC
SVC12	$$A$SUP1 (SGSVC)	FB
SVC1213	$$A$SUP1 (SGSVC)	FB
SVC13	$$A$SUP1 (SGSVC)	FB
SVC15	$$A$SUP1 (SGTCHS)	CA,CB

Label	Phase	Chart
SVC18	$$A$SUP1 (SGSVC)	FL
SVC19	$$A$SUP1 (SGSVC)	FL
SVC2	$$BEOJ	RB
SVC22	$$A$SUP1 (SGSVC)	FF
SVC22A	$$A$SUP1 (SGSVC)	FF
SVC23	$$A$SUP1 (SGSVC)	FF
SVC24	$$A$SUP1 (SGSVC)	FF
SVC25	$$A$SUP1 (SGTCHS)	CA
SVC26	$$A$SUP1 (SGSVC)	FF
SVC27	$$A$SUP1 (SGTCHS)	CA
SVC29	$$A$SUP1 (SGSVC)	FB
SVC2DND	$$A$SUP1 (SGSVC)	FC
SVCALL	$$ANERRN	LX
SVCERR	$$BEOJ	RB
SVCRTN1	$$A$SUP1 (SGSVC)	FA
SVEREG	$$A$SUP1 (SGTCON)	HA
SVNWAD	$$A$SUP1 (SGTCHS)	CC
SWITCH	$$ANERRO	LY
SXTPOK	$$BATTNL	PU
SXTRT1	$$A$SUP1 (SGSVC)	FG,FL
SYCLAS	$$ANERRO	LY
SYSFILE	$$A$SUP1 (SGTCHS)	CH
SYSFILE1	$$A$SUP1 (SGTCHS)	CH
SYSFILE2	$$A$SUP1 (SGTCHS)	CH
SYSFILE3	$$A$SUP1 (SGTCHS)	CM
SYSFILE4	$$A$SUP1 (SGTCHS)	CH
SYSFILE5	$$A$SUP1 (SGTCHS)	CF
SYSIN	$$A$SUP1 (SGTCHS)	CH
SYSINOUT	$$A$SUP1 (SGTCHS)	CF
SYSMODM	$$BATTNI	NX
SYSRFG	$$BATTNI	NX
SYSSWH	$$BATTNJ	PG

Program switch set to branch when system units are to be logged. The switch is set to branch by the list BG routine (Chart PF). The switch is reset to NOP after the system class units have all been logged (Chart PH).

SYSTST	$$BDUMP	RV

Routine similar to the NOCHNG routine. Identifies the physical device assigned to SYSLST for a background program dump.

SYSUNT	$$BATTNJ	PG
SYSXN2	$$BATTNI	NX
SYSXXX	$$BATTNI	NX
T01	$$ANERRG	LN
T03	$$ANERRH	LQ
T06	$$ANERRH	LQ
TABLE	$$BATTNA	*

Defines the branch vector table that loads and executes B-transients required by either the nonresident attention routine or the single program initiator. If a START statement has been processed, indicating the B-transients are functioning as an initiator, the table entries starting at location INTABL are added to the branch vector table. Each table entry consists of:

- Operation field of the control statement.
- Phase identifier (an alphabetic character).
- Branch vector index factor used to get the first executable instruction of the processing phase.

TAPE1	$$BDUMP	RN

When SYSLST is found to be a tape drive, the CCB and CCW are modified accordingly to perform a sense operation for a file-protect condition. Register 12 signals the fetched dump program that a tape drive receives the storage dump.

TAPEMODE2	$$ANERR9	MY
TAPNOP	$$BDUMPF	RY
TAPRTN	$$BDUMPF	RX

Data address is stored in the tape CCW, and CCB is furnished with the CCW address. Switches at OUT1 and TAPNOP are set to branch to perform functions necessary for output on tape drive.

TAPSYS	$$BDUMPF	RY
TAPSYS1	$$BDUMPF	RY
TCAN2	$$ANERRY	MR
TEB	$$BTERM	RG
TEBVER	$$ANERRE	LF
	$$ANERRF	LH
TERM	$$BEOJ1	RK
	$$BILSVC	RP
	$$BPSW	RS
	$$BPCHK	RT
	$$BDUMP	RW

*Listing only.

Label	Phase	Chart
TEST	$$BTERM	RF
TEST1	$$BTERM	RF
TESTIGN	$$ANERRQ	MA
TESTSVC	$$A$SUP1 (SGDFCH)	EB
TFILL	$$BATTNK	PL
TIMER	$$BATTNN	QE
TIMLNK	$$BATTNN	QE
TLBL	$$BATTNK	PM
TM	$$BEOJ3	RD
TMEKEY	$$A$SUP1 (SGSVC)	FE
TMEKEY1	$$A$SUP1 (SGSVC)	FE
TMERT1	$$A$SUP1 (SGSVC)	FJ
TMFAVP	$$BATTNI	NZ
TMPAR1	$$BATTNA	*

Defines a doubleword save area used in conjunction with the general scan routine (Chart ND). The first word is loaded from register POINT1 with the address of a statement field (operation code or operand). The second word is loaded from register POINT2 with the remaining I/O area. Also used as an entry point during B-transient initialization.

Label	Phase	Chart
TNAERR	$$BATTNN	QE
TPBUSY	$$A$SUP1 (SGTCHS)	CD
TPBUSY1	$$A$SUP1 (SGTCHS)	CD
TPLAB	$$BATTNK	PL
TPLEND	$$BATTNK	PL
TPMARK	$$BDUMPF	SA
TPRTN	$$BDUMPB	SB
	$$BPDUM1	SM
TPTYPE	$$BEOJ	RC
	$$BEOJ1	RK
	$$BEOJ2	RM
	$$BILSVC	RP
	$$BPSW	RS
	$$BPCHK	RT
	$$BDUMP	RW
TPTYPE1	$$BEOJ	RC
	$$BEOJ1	RK
	$$BEOJ2	RM
	$$BILSVC	RP
	$$BPSW	RS
	$$BPCHK	RT
TRANS	$$BDUMPB	SE
TRKCHK	$$A$SUP1 (SGDSK)	GB
TRKEOC	$$A$SUP1 (SGDSK)	GB
TRNOFF	$$A$SUP1 (SGTCHS)	CK
TRYNXT	$$A$SUP1 (SGTCHS)	CG
TST03U	$$ANERRU	MF
TSTATD	$$ANERRR	MC
TSTATTN	$$A$SUP1 (SGTCHS)	CK
TSTBMX	$$A$SUP1 (SGTCHS)	CM
TSTCAN	$$ANERRR	MC
TSTCCB	$$ANERRN	LW
TSTCCC	$$ANERRU	MF
TSTCDC	$$ANERRU	MF
TSTCLN	$$ANERRD	LE
	$$ANERRF	LJ
TSTCOR	$$BDUMPF	SA
	$$BDUMPB	SF
	$$BDUMPD	SK
	$$BPDUM1	SP

Entry to subroutine that tests whether storage area to be printed on a line is in dump limits and whether the next line is the last line. Register 3 is a pointer to the storage address of the first byte of a line to be printed, and register 5 points to the last byte of the line. See CMPCOR label discussion, which is part of this subroutine.

Label	Phase	Chart
TSTDEV	$$A$SUP1 (SGTCHS)	CD
TSTEOJ	$$A$SUP1 (SGTCHS)	CE,CF
TSTERF	$$A$SUP1 (SGTCHS)	CK
TSTLST	$$BEOJ1	RJ
	$$BEOJ2	RM
	$$BILSVC	RN
	$$BPSW	RR
	$$BPCHK	RT
	$$BDUMPF	SA
	$$BDUMPB	SF
	$$BDUMPD	SK
	$$BPDUM1	SN
TSTLST1	$$BDUMPF	SA
	$$BDUMPB	SF
	$$BDUMPD	SK
	$$BPDUM1	SN

Switch that is set to a branch on last line of dump. If a portion of core is found that is identical to the previous line, this switch is set to no-op and the identical data is shown by printing a line with --SAME--.

Label	Phase	Chart
TSTNXT	$$A$SUP1 (SGTCHS)	CH
TSTPRT	$$BDUMPF	SA
	$$BDUMPB	SF
	$$BDUMPD	SK
	$$BPDUM1	SN
TSTQEF	$$A$SUP1 (SGTCHS)	CL
TSTRD	$$ANERRD	LE
	$$ANERRE	LG

*Listing only.

Label	Phase	Chart
TSTRCT	$$ANERRD	LD
TSTRTY	$$ANERRN	LW
TSTSSL	$$ANERRB	LC
TSTSVC	$$A$SUP1 (SGDFCH)	EB
TSTTRG	$$ANERRS	MD
TSTUCK	$$A$SUP1 (SGTCHS)	CJ
TXCUU	$$BATTNI	NY
UALNOT	$$BATTNJ	PJ
UAPSWH	$$BATTNJ	PE

Program switch set to branch after unit assignments have been listed. This switch is initialized in the NOP state. It is set to branch just before the 'UNASSIGNED' header is logged.

UC	$$ANERRE	LG
UCBPAR	$$ANERRV	MH
UCHK	$$ANERRF	LJ
UCS	$$BATTNM	QC
UCS1	$$BATTNM	QC
UCS2	$$BATTNM	QC
UCS3	$$BATTNM	QC
UCS4	$$BATTNM	QC
UCSDN	$$BATTNM	QC
UCSSCN	$$BATTNM	QC
UCSTST	$$ANERRV	MJ
UCUERR	$$BATTNI	PD
UNA	$$BATTNI	PC
UNAGO	$$BATTNI	PD
UNALOP	$$BATTNI	PD
UNANO	$$BATTNI	PD
UNARTN	$$BATTNI	PD
UNASSGN	$$BTERM	RE

Test for resetting symbolic device assignments, and if required, continue to Chart RF, where the LUBNDX from the PIB of this program (F1 or F2) is inserted in register 5. In the case of F1, for example, LUBNDX is equal to the sum of the LUBs assigned to devices owned by the system programs, the background program, and the foreground 2 program. This index is doubled because there are 2 bytes per LUB entry. The result is the displacement from the LUB table starting address, where this foreground program's LUBs begin. By adding this displacement to the LUB table starting address, the actual address for the first LUB is obtained in register 5.

The number of LUBs assigned to this type of foreground program is obtained from the NICL (Number-in-Class); this value is adjusted and doubled.

UNAV	$$BATTNG	NS
UNCLOG	$$BATTNH	NU
UNCOMMON	$$A$SUP1 (SGTCHS)	CE
UNDTR	$$ANERRT	ME
	$$ANERRW	MK
UNKN	$$ANERRB	LC
	$$ANERRE	LG
UNNORM	$$BEOJ	RA

Routine entered when abnormal end-of-job condition exists. Investigation of cause of cancel and type of program executing is made, to determine which B-transient of the terminating phases to call next.

UNPA	$$BATTNI	NZ

Entry point to a routine that unassigns currently assigned logical units. The subroutine saves, in location LBSLUB, the LUB entry of the LUB to be unassigned. It then unassigns the LUB in the LUB table. It checks the LUB table and JIB table for other LUBs that point to the same physical unit as that of the LUB just unassigned. It resets the ownership flag in the PUB if no other LUBs point to that physical unit. Any stored alternate assignments found in the JIB table are treated as LUBs (unassigned, then checked for matching PUB pointers).

UNPAN2	$$BATTNI	NZ
UNPCH	$$ANERRO	LY
UNPK	$$BDUMPF	RZ
	$$BDUMPB	SE
	$$BDUMPD	SJ
	$$BPDUM1	SP
UNPK1	$$BDUMPF	RZ
	$$BDUMPD	SJ
	$$BPDUM1	SP
UNPKSEN	$$ANERRO	LY
UNRCERP	$$ANERRA	LA
UNSUPTD	$$ANERR9	MY
UNTCK1	$$A$SUP1 (SGUNCK)	DA
UNTCK1A	$$A$SUP1 (SGUNCK)	DA
UNTCK2	$$A$SUP1 (SGUNCK)	DA
UNUSENS	$$ANERRU	MG
USREXT	$$A$SUP1 (SGUNCK)	DA
USRSEN	$$A$SUP1 (SGUNCK)	DB
USRUCK	$$A$SUP1 (SGUNCK)	DA
VOL	$$BATTNK	PK
VERIF	$$ANERRB	LC
VLDADR1	$$A$SUP1 (SGTCON)	HA
VLDADR2	$$A$SUP1 (SGTCON)	HA

Label	Phase	Chart
VLDADR3	$$A$SUP1 (SGTCON)	HA
WAIT	$$ANERRD	LE
	$$ANERRE	LG
	$$ANERRY	MQ
WAIT1	$$ANERR0	MU
WAITLOOP	$$A$SUP1 (SGDFCH)	ED
WFM	$$BATTNH	NT
WFMRES	$$BATTNH	NU
WRITE	$$ANERRF	LH
WRREP	$$ANERRF	LH
WRYT	$$ANERRD	LE
XTENT	$$BATTNL	PU
XTOP12	$$BATTNL	PX

Entry point to a subroutine that extracts and validity checks the first two operands (type and sequence number) of an XTENT statement. It converts the operand to binary, and stores it in the label area, DSECT (I/O area).

XTOP12A	$$BATTNO	QJ
XTOP12B	$$BATTNO	QJ
XTOP3	$$BATTNL	PV
XTOP34	$$BATTNL	PX

Entry point to a subroutine that extracts limit information from the XTENT statement, performs initial validity checks, converts the numeric EBCDIC limit data to binary, and puts the limits into the label area, DSECT (I/O area).

XTOP5	$$BATTNL	PV

Entry point to the routine that:

1. Gets and checks the serial number, and stores it in the label area DSECT (I/O area).

2. Converts the SYSXXX field of the extent to class and displacement.

3. Gets the B2 field of an extent, converts it to binary, and stores it in the label area, DSECT (I/O area).

XTOUT	$$BATTNL	PW
XTUNIT	$$BATTNL	PW
	$$BATTNO	QH
ZERTEB	$$ANERRS	MD
ZROREG	$$A$SUP1 (SGTCHS)	CN

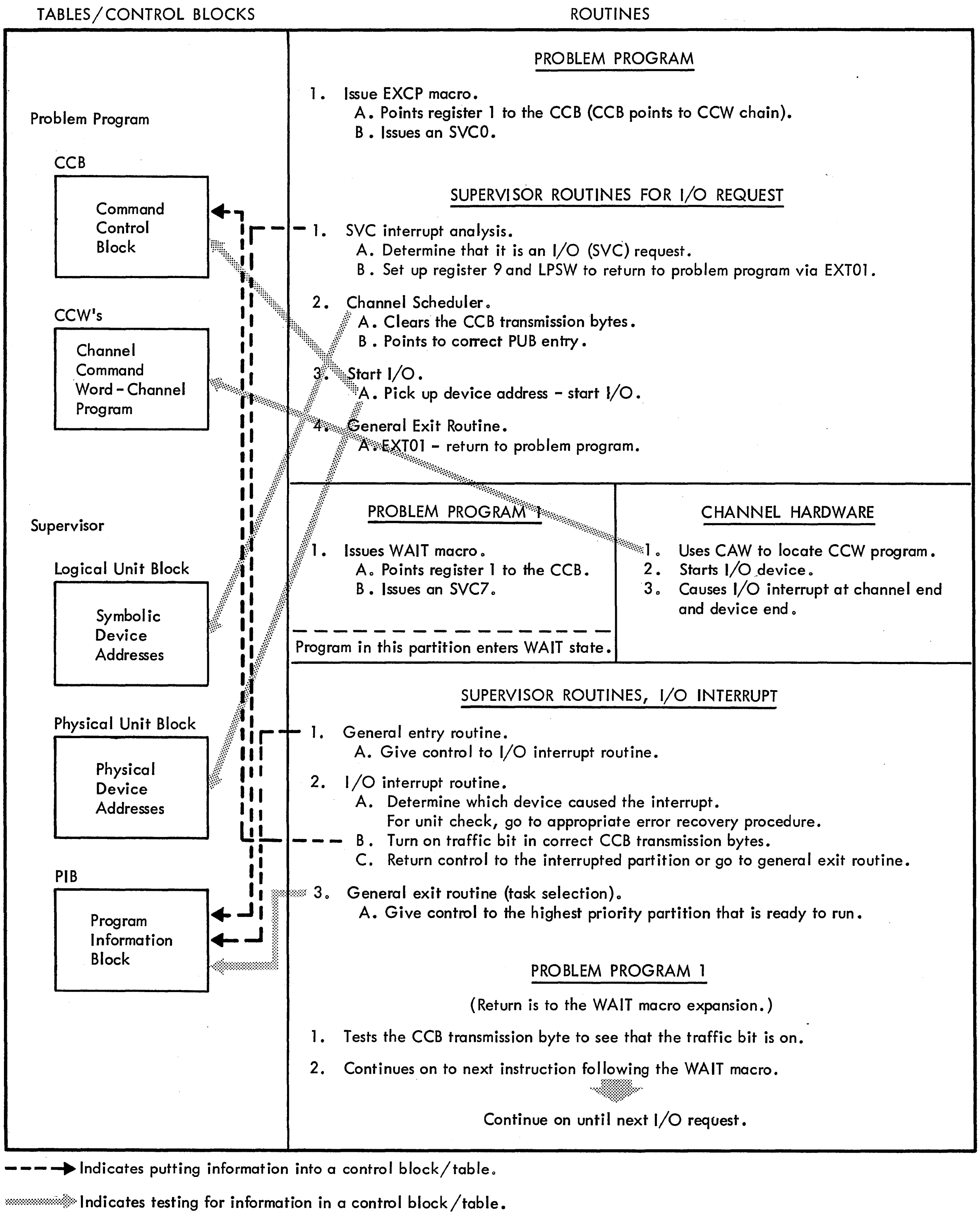
Message	Phase	Chart
0P08	$$A$SUP1 (Disk)	GA
	$$ANERRE (Tape)	LF
	$$ANERRG (Data Cell)	LN
	$$ANERRT (MICR)	ME
	$$ANERRU (Unit Record)	MG
	$$ANERRV (Unit Record)	MH
	$$ANERRW (MICR)	MK,ML
	$$ANERRX (Paper Tape)	MN
	$$ANERR9 (Optical Reader)	MX
0P09	$$A$SUP1 (Disk)	GA
	$$ANERRA (Disk)	LB
	$$ANERRE (Tape)	LF
	$$ANERRG (Data Cell)	LN
	$$ANERRT (MICR)	ME
	$$ANERRU (Unit Record)	MG
	$$ANERRV (Unit Record)	MJ
	$$ANERRW (MICR)	MK,ML
	$$ANERRX (Paper Tape)	PT
	$$ANERR9 (Optical Reader)	MX
0P10	$$A$SUP1 (Disk)	GA
	$$ANERRA (Disk)	LA
	$$ANERRD (Tape)	LD
	$$ANERRG (Data Cell)	LM
	$$ANERRU (Unit Record)	MG
	$$ANERRV (Unit Record)	MH
	$$ANERRX (Paper Tape)	MN
0P11	$$A$SUP1 (Disk)	GA
	$$ANERRB (Disk)	LC
	$$ANERRD (Tape)	LD
	$$ANERRL (Tape)	LL
	$$ANERRK (Data Cell)	LU
	$$ANERRV (Unit Record)	MJ
	$$ANERRX (Paper Tape)	MN
0P12	$$ANERRB (Disk)	LC
	$$ANERRK (Data Cell)	LU
0P13	$$ANERRB (Disk)	LC
	$$ANERRK (Data Cell)	LU
0P14	$$A$SUP1 (Disk)	GA
	$$ANERRB (Disk)	LC
	$$ANERRE (Tape)	LF
	$$ANERRG (Data Cell)	LN
	$$ANERRV (Unit Record)	MJ
	$$ANERR9 (Optical Reader)	MY
0P15	$$A$SUP1 (Disk)	GA
	$$ANERRB (Disk)	LB
	$$ANERRJ (Data Cell)	LS
0P16	$$ANERRB (Disk)	LC
	$$ANERRK (Data Cell)	LU
0P17	$$ANERRB (Disk)	LC
	$$ANERRG (Data Cell)	LP
	$$ANERRL (Tape)	LL
0P18	$$A$SUP1 (Disk)	GA
	$$ANERRB (Disk)	LC
	$$ANERRL (Tape)	LL
	$$ANERRG (Data Cell)	LP
	$$ANERRT (MICR)	ME
	$$ANERRU (Unit Record)	MG
	$$ANERRV (Unit Record)	MJ
	$$ANERRW (MICR)	MK,MM
	$$ANERRX (Paper Tape)	MN
	$$ANERR9 (Optical Reader)	MY
0P19	$$ANERRB (Disk)	LC
	$$ANERRE (Tape)	LG
	$$ANERRG (Data Cell)	LP
	$$ANERRT (MICR)	ME
	$$ANERRU (Unit Record)	MF,MG
	$$ANERRV (Unit Record)	MJ
	$$ANERRW (MICR)	MK,ML
	$$ANERRX (Paper Tape)	MN
	$$ANERR9 (Optical Reader)	MY
0P20	$$A$SUP1 (Disk)	DB,DF,GA, GB
	$$ANERRD (Tape)	LE
	$$ANERRE (Tape)	LG
	$$ANERRF (Tape)	LK
	$$ANERRL (Tape)	LL
	$$ANERRH (Data Cell)	LQ
	$$ANERRI (Data Cell)	LR
	$$ANERRJ (Data Cell)	LS
0P21	$$A$SUP1 (Disk)	GA
	$$ANERRJ (Data Cell)	LS
0P22	$$ANERRG (Data Cell)	LM
0P23	$$ANERRJ (Data Cell)	LS
0P24	$$ANERRA (Disk)	LA
0P25	$$ANERRA (Disk)	LA
0P26	$$ANERRA (Disk)	LB
	$$ANERRG (Data Cell)	LM
0P27	$$ANERRA (Disk)	LA
	$$ANERRC ----	LC
	$$ANERRU (Unit Record)	ME
0P28	$$A$SUP1 (Disk)	GA
	$$ANERRA (Disk)	LA
	$$ANERRE (Tape)	LF
	$$ANERRG (Data Cell)	LM

Message	Phase	Chart
	$$ANERRT (MICR)	ME
	$$ANERRU (Unit Record)	MF
	$$ANERRW (MICR)	MK,ML
	$$ANERRX (Paper Tape)	MN
	$$ANERR9 (Optical Reader)	MX
0P29	$$ANERRE (Tape)	LF
0P30	$$ANERRE (Tape)	LF
0P31	$$ANERRA (Disk)	LA
	$$ANERRD (Tape)	LD
0P32	$$ANERRE (Tape)	LG
0P33	$$ANERRV (Unit Record)	MH
0P34	$$ANERRW (MICR)	MM
0P35	$$ANERR9 (Optical Reader)	MX
0P36	$$A$SUP1 (Disk)	GA
	$$ANERRA (Disk)	LB
	$$ANERRJ (Data Cell)	LS
0P37	$$ANERRT (MICR)	ME
	$$ANERRW (MICR)	MK
	$$ANERR9 (Optical Reader)	MY
0P60D	$$ANERRY	MR
0P70I	$$BEOJ2	RL*
0P71I	$$BEOJ2	RL*
0P72I	$$BEOJ2	RL*
0P73I	$$BEOJ2	RL*
0P74I	$$BEOJ2	RL*
0P75I	$$BEOJ2	RL*
0P76I	$$BEOJ2	RL*
0P77I	$$BEOJ2	RL*
0P78I	$$BEOJ2	RL*
0P79I	$$BEOJ2	RL*
0S00I	$$BPCHK	RT
	$$BILSVC	RN
0S01I	$$BEOJ2	RL*
0S02I	$$BEOJ2	RL*
0S03I	$$BPCHK	RT*
0S04I	$$BILSVC	RN*
0S05I	$$BILSVC	RN*
0S06I	$$BEOJ1	RJ*
0S07I	$$BPSW	RR
0S08I	$$BEOJ	RA
0S09I	$$BEOJ1	RJ*
0S10I	$$BTERM	RE
0S11I	$$BEOJ1	RJ*
1A00D	$$BATTNI	NW
1A10D	$$BATTNI	NV
	$$BATTNP	QM
1A20D	$$BATTNI	NV,NW
	$$BATTNM	QC
	$$BATTNP	QM
1A30D	$$BATTNI	NW
1A40D	$$BATTNI	NX
	$$BATTNK	PN
	$$BATTNM	QC,QD
1A50D	$$BATTNI	NY
1A60D	$$BATTNI	PD
1A70D	$$BATTNI	NY
1C20D	$$BATTNH	NU
1C30A	$$BATTNM	PY,QC
1C40I	$$BATTNB	NF
1C50I	$$BATTNB	NF
1C60D	$$BATTNN	QE
1I30D	$$BATTNC	NG
1I40D	$$ANERR0	MU
1I60A	$$BATTNA	NA
1I80I	$$BTERM	RE
1L00D	$$BATTNL	PS,PU,PV,PX
	$$BATTNO	QG,QH,QJ,QK
1L10D	$$BATTNK	PR
	$$BATTNL	PU
	$$BATTNM	PZ
	$$BATTNO	QF
1P00D	$$BATTNF	NP

*Also refer to Figures 25 and 38.

Message	Phase	Chart
1P10D	$$BATTNG	NS
1S00D	$$BATTNA	NB,NC
	$$BATTNB	NE
	$$BATTNC	NG
	$$BATTNE	NM,NN
	$$BATTNG	NS
	$$BATTNI	NV,NX,NY, PB,PC,PD
	$$BATTNJ	PE
	$$BATTNK	PK,PL,PM, PN,PP,PQ
	$$BATTNL	PS,PT,PV
	$$BATTNM	PY,QC,QD
	$$BATTNN	QE
	$$BATTNO	QF,QG,QH, QJ,QK,QL
	$$BATTNP	QM,QN
1S10D	$$BATTNK	PK,PL
	$$BATTNL	PS,PU
	$$BATTNM	PY
	$$BATTNO	QF,QG

APPENDIX C: I/O REQUEST-I/O INTERRUPT SEQUENCE



TABLES/CONTROL BLOCKS

Problem Program

CCB

Command Control Block

CCW's

Channel Command Word - Channel Program

Supervisor

Logical Unit Block

Symbolic Device Addresses

Physical Unit Block

Physical Device Addresses

PIB

Program Information Block

ROUTINES

PROBLEM PROGRAM

1. Issue EXCP macro.
   A. Points register 1 to the CCB (CCB points to CCW chain).
   B . Issues an SVC0.

SUPERVISOR ROUTINES FOR I/O REQUEST

1. SVC interrupt analysis.
   A. Determine that it is an I/O (SVC) request.
   B . Set up register 9 and LPSW to return to problem program via EXT01.
2. Channel Scheduler.
   A. Clears the CCB transmission bytes.
   B . Points to correct PUB entry.
3. Start I/O.
   A. Pick up device address - start I/O.
4. General Exit Routine.
   A. EXT01 - return to problem program.

PROBLEM PROGRAM 1

1. Issues WAIT macro.
   A. Points register 1 to the CCB.
   B . Issues an SVC7.

Program in this partition enters WAIT state.

CHANNEL HARDWARE

1. Uses CAW to locate CCW program.
2. Starts I/O device.
3. Causes I/O interrupt at channel end and device end.

SUPERVISOR ROUTINES, I/O INTERRUPT

1. General entry routine.
   A. Give control to I/O interrupt routine.
2. I/O interrupt routine.
   A. Determine which device caused the interrupt.
      For unit check, go to appropriate error recovery procedure.
   B . Turn on traffic bit in correct CCB transmission bytes.
   C. Return control to the interrupted partition or go to general exit routine.
3. General exit routine (task selection).
   A. Give control to the highest priority partition that is ready to run.

PROBLEM PROGRAM 1

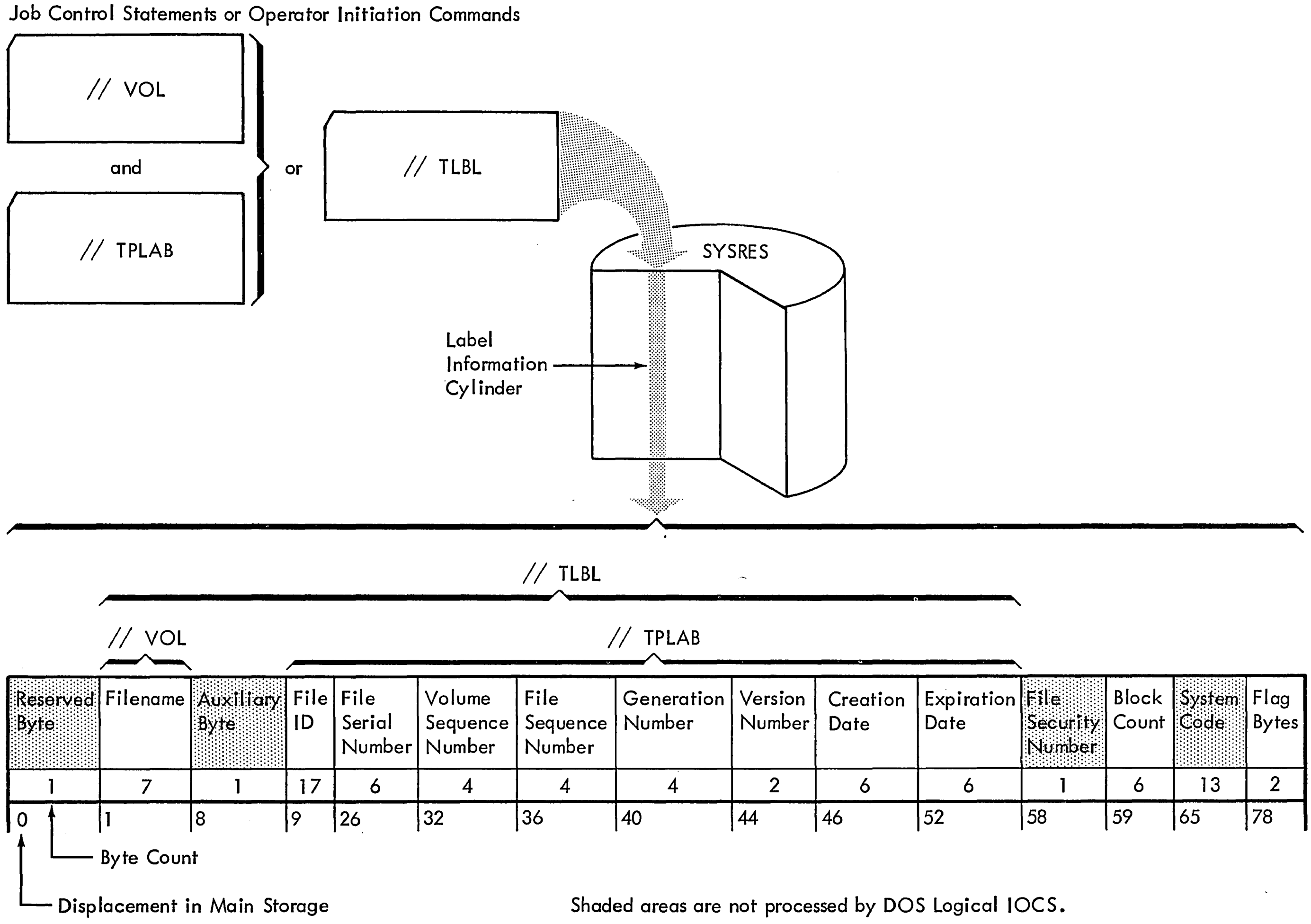
(Return is to the WAIT macro expansion.)

1. Tests the CCB transmission byte to see that the traffic bit is on.
2. Continues on to next instruction following the WAIT macro.

Continue on until next I/O request.

----→ Indicates putting information into a control block/table.

▨⇒ Indicates testing for information in a control block/table.


Job Control Statements or Operator Initiation Commands
// VOL
and
// TPLAB
or
// TLBL
SYSRES
Label Information Cylinder
// TLBL
// VOL
// TPLAB
Byte Count
Displacement in Main Storage

Reserved Byte	Filename	Auxiliary Byte	File ID	File Serial Number	Volume Sequence Number	File Sequence Number	Generation Number	Version Number	Creation Date	Expiration Date	File Security Number	Block Count	System Code	Flag Bytes
1	7	1	17	6	4	4	4	2	6	6	1	6	13	2
0	1	8	9	26	32	36	40	44	46	52	58	59	65	78

Shaded areas are not processed by DOS Logical IOCS.

Figure 42. Format of SYSRES Tape Label Information

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18		19		Field
DLBL - EXTENT Indicator	Filename	DA/IS Switch	File ID	Format ID	File Serial Number	Volume Sequence Number	Creation Date	Expiration Date	Reserved	Open Code	System Code	Volume Serial Number	EXTENT Type	EXTENT Sequence Number	EXTENT Lower Limit	EXTENT Upper Limit	System Unit Class	System Unit Order	2321 Lower Cell	2321 Upper Cell	Another EXTENT if DA or ISFMS
1	7	1	44	1	6	2	3	3	2	1	13	6	1	1	4	4	1	1	1	1	Bytes
0	1	8	9	53	54	60	62	65	68	70	71	84	90	91	92	96	100	101	102	103	Displacement

Field	Name	Description
1.	DLBL - EXTENT Indicator	X'80' = Next EXTENT on new pack. X'40' = Last EXTENT X'20' = Bypass EXTENT (SD), or number of EXTENTS (DA or ISFMS). X'10' = New volume on same unit. X'08' = EXTENT limits omitted. X'04' = EXTENT converted to DASD address.
2.	Filename	
3.	DA/IS Switch	Same as field 1 except that only bits 4 and 5 are used for DA or ISFMS.
4.	File ID	File identifier including generation and version numbers. If field is missing on DLBL card, filename padded with blanks is inserted.
5.	Format ID	Numeric 1 is inserted.
6.	File Serial Number	Volume serial number from first EXTENT.
7.	Volume Sequence Number	Always initialized to X'0001'.
8.	Creation Date	Initialized with 3 bytes of X'00'.
9.	Expiration Date	If date is in the form YYDDD, it is converted to YDD. If date is in retention period form, 1 to 4 characters, the field is padded with binary zeros.
10.	Reserved	The retention period, if specified, is converted to a 2-byte number and inserted in this field.

11.	Open Code	DLBL type:
		S = Sequential
		D = Direct Access
		C or E = Indexed Sequential File Management System

12.	System Code	Initialized to contain: DOS/360 VER 3. This field is not processed by DOS.
13.	Volume Serial Number	Volume serial number for EXTENT.
14.	EXTENT Type	Same codes as in Format 1 label: X'00' = Next three fields do not indicate any extent. X'01' = Prime area (ISFMS) or consecutive area, etc., (i.e., the extent containing the user's data records). X'02' = Overflow area of an ISFMS file. X'04' = Cylinder index or master index of an ISFMS file. X'40' = User label track area. X'8n' = Shared cylinder indicator, where n=1,2, or 4.
15.	EXTENT Sequence Number	Number of extents as determined by the EXTENT card sequence.
16.	EXTENT Lower Limit	Relative extent converted to the form HHnnT for // DLBL job control statement, or CCHH from // DLAB job control statement.
17.	EXTENT Upper Limit	Same as field 16, but for upper limit.
18.	System Unit Class System Unit Order	Device class and unit numbers.
19.	2321 Lower Cell 2321 Upper Cell	2321 EXTENT lower and upper limit Bin numbers.

Note: For Sequential Disk files, a complete 104-byte block is repeated for each new EXTENT. For Direct Access and ISFMS files, only fields 13 through 18 are repeated for each EXTENT.

Figure 43. Format of SYSRES DASD Label Information

PIK (Program Interrupt Key)

The PIK is a halfword in length and consists of a zero value in the high-order byte and the key value in the low-order byte. The key value is the key of the program that was last enabled for interrupts.

When an interrupt occurs, the value in the PIK indicates to the supervisor which program was interrupted. It can also be used by transient programs and problem programs to determine if they are running as BG, F1, or F2.

The value of the PIK equals the displacement from the beginning of the PIB table to the PIB entry for the program (task). For BG, F2, and F1 tasks, this value equals the storage protect key multiplied by 16.

Task	PIK Value
All Bound*	X'00'
BG	X'10'
F2*	X'20'
F1*	X'30'
AR	X'40'
Quiese I/O	X'50'
Supervisor	X'60'

*Multiprogramming generation option only.

The PIK is set by task selection within the general exit routine. The fetch routine sets the PIK to X'60', because it enables itself for interrupts and because it gets control directly from the SVC interrupt routines. The SVC interrupt routines, like other completely disabled supervisor routines, do not change the PIK from the value it had when the interrupt occurred that transferred control.

LTK (Logical Transient Key)

The LTK has the same value as the PIK when the logical transient area is in use. When the transient area is free, the LTK equals zero. The SVC 2 routine sets the LTK, and the SVC 11 routine resets it to zero.

RID (Requestor Identification)

See Figure 12, REQID (Item F).

RIK (Requestor I/O Key)

When a supervisor routine (fetch or physical transient) issues an SVC 0 or SVC 15, the routine puts the value to be used in the CAW storage protect key into the high-order digit of the second byte of the RIK halfword. When this value is zero, the low order digit can have these special meanings:

RIK	Meaning
X'01'	This is a SYSLOG I/O request. The channel scheduler is not to type a SYSLOG ID prefix.
X'02'	This has been a fetch I/O request. This special code is required by ERP to recognize fetch requests.

Fetch always sets a X'02' in the RIK. ERP transients put the key of the program requiring ERP into the RIK, when the ERP is a retry of a user EXCP and the ERP transient requires control to return to itself.

Physical transients put a X'01' into the RIK when they are doing a SYSLOG I/O. The PIK for physical transients has a value of X'06', therefore the channel scheduler would type "SP" (supervisor ID) as the SYSLOG ID. The physical transients put the ID of the program referred to by the message into the message.

FIK (Fetch I/O Key)

Used by the fetch to validate the phase name address and load address. FIK has the following values:

1	Key of the problem program requestor.
2	0
3	0
4	0 if the transient issued the SVC 4. Key of the problem program if not a transient.

APPENDIX F: EXPLANATION OF FLOWCHART SYMBOLS

DESCRIPTION

```
*****A1**********
*               *
*               *
*   PROCESSING  *
*               *
*     *B2       *
*****************
```

A GROUP OF PROGRAM INSTRUCTIONS THAT PERFORM A PROCESSING FUNCTION OF THE PROGRAM. THE LABEL, IF ANY, IS SHOWN ABOVE THE BLOCK.

*B2
IF ANY ADDITIONAL EXPLANATION IS REQUIRED, ITS LOCATION ON THE CHART IS IDENTIFIED BY AN ASTERISK AND THE BLOCK ID.

```
*****C1**********
*LABEL1       BW*
*-*-*-*-*-*-*-*-*
*  SUBROUTINE   *
*               *
*               *
*****************
```

DESCRIPTION OR TITLE OF A ROUTINE THAT IS DETAILED ON ANOTHER FLOWCHART. THE STARTING LABEL OF THE ROUTINE AND THE FLOWCHART ID APPEAR ABOVE THE STRIPE.

```
*****D1*******
*             *
*   PROGRAM    *
* MODIFICATION  *
*             *
*            *
**************
```

AN INSTRUCTION, OR GROUP OF INSTRUCTIONS, THAT CHANGES PORTIONS OF A ROUTINE OR INITIALIZES A ROUTINE FOR GIVEN CONDITIONS.

```
   **E1*******
  *           *
 * PREDEFINED  *
*   PROCESS     *
 *             *
  *           *
   ***********
```

A GROUP OF OPERATIONS NOT DETAILED IN THE FLOWCHARTS IN THIS MANUAL, SUCH AS USER'S ROUTINES.

```
******F1***********

* INPUT/OUTPUT  *

 *            *

  *************
```

ANY FUNCTION OF AN INPUT/OUTPUT DEVICE OR PROGRAM, USUALLY BRANCHING TO AN I/O ROUTINE TO PERFORM THE BLOCK.

```
        .*.
   G1 .*   *.
    .*       *.
  *.  DECISION .*
    *.       .*
      *.   .*
        *.*
```

POINTS WHERE THE PROGRAM BRANCHES TO ALTERNATE PROCESSING, BASED UPON VARIABLE CONDITIONS SUCH AS PROGRAM SWITCH SETTINGS AND TEST RESULTS.

```
****H1*********
*             *
*   TERMINAL  *
*             *
***************
```

THE BEGINNING, END, OR POINT OF INTERRUPTION IN A PROGRAM.

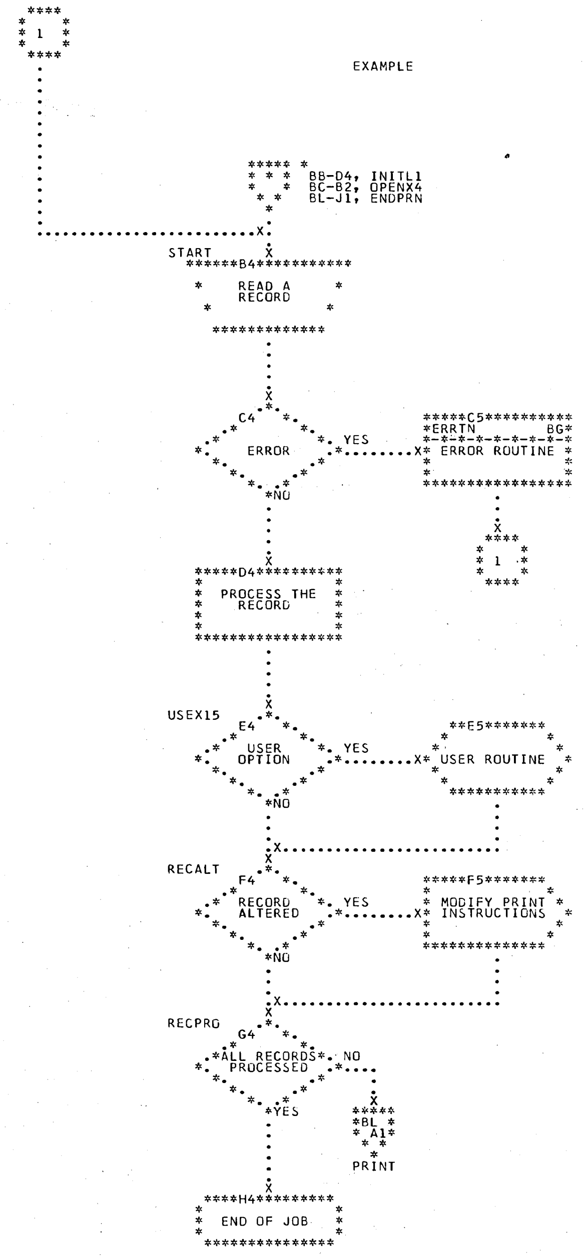
```
 ****
*    *
*  1 *
*    *
 ****
```

ON-PAGE CONNECTOR. AN ENTRY FROM, OR AN EXIT TO, ANOTHER FUNCTION ON THE SAME FLOWCHART. THE NUMBER IN THE CONNECTOR IDENTIFIES THE CORRESPONDING ENTRY OR EXIT ON THE CHART.

```
 *****
 *BD *
 * D4*
  * *
   *
 FILINP
```

OFF-PAGE CONNECTOR. AN ENTRY FROM, OR AN EXIT TO, A GIVEN POINT ON ANOTHER FLOWCHART. THE CHARACTERS IN THE CONNECTOR IDENTIFY THE CHART AND BLOCK. THE CORRESPONDING LABEL, IF ANY, IS PLACED OUTSIDE THE CONNECTOR. FOR MULTIPLE ENTRIES AND EXITS, AN ASTERISK APPEARS IN THE CONNECTOR AND THE CHARACTERS ARE LISTED NEARBY.

EXAMPLE



Reader's Comments Form

IBM System/360
Disk Operating System
Supervisor and Physical and Logical Transients **Y24-5084-0**

Your comments, accompanied by answers to the following questions, help us produce better publications for your use. If your answer to a question is "No" or requires qualification, please explain in the space provided below. Please give specific page and line references with your comments when appropriate. All comments will be handled on a non-confidential basis. Copies of this and other IBM publications can be obtained through IBM branch offices.

	Yes	*No*
● Does the publication meet your needs?	□	□
● Did you find the material:		
Easy to read and understand?	□	□
Organized for convenient use?	□	□
Complete?	□	□
Well illustrated?	□	□
Written for your technical level?	□	□

● What is your occupation? ____

● How do you use this publication:

As an introduction to the subject? □
For advanced knowledge of the subject? □
For information about operating procedures? □

As an instructor in a class? □
As a student in a class? □
As a reference manual? □

Your comments:

Thank you for your cooperation. No postage necessary if mailed in the U.S.A.

Staple

Your comments, please . . .

This publication is one of a series that serves as a reference source for systems analysts, programmers, and operators of IBM systems. Your answers to the questions on the back of this form, together with your comments, help us produce better publications for your use. Each reply is carefully reviewed by the persons responsible for writing and publishing this material. All comments and suggestions become the property of IBM.

Please note: Requests for copies of publications and for assistance in using your IBM system should be directed to your IBM representative or to the IBM sales office serving your locality.

Fold Fold

FIRST CLASS
PERMIT NO. 170
ENDICOTT, N. Y.

BUSINESS REPLY MAIL
NO POSTAGE NECESSARY IF MAILED IN THE UNITED STATES

POSTAGE WILL BE PAID BY . . .

IBM Corporation
P. O. Box 6
Endicott, N. Y. 13760

Attention: Programming Publications, Dept. 157

Cut Along Line

Fold Fold

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]

Additional Comments:

IBM S/360 Printed in U.S.A. Y24-5084-0

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]