Program Product

Information Management System/360 for the IBM System/360 Operations Manual Volume I - Systems Operation

Program Number 5736-CX3

Information Management System/360 is an Operating System/360 processing program designed to facilitate the implementation of medium to large common data bases in a multiapplication environment. This environment is created to accommodate both online message processing and conventional batch processing, either separately or concurrently. The system permits the evolutionary expansion of data processing applications from a batch-only to a teleprocessing environment.

This volume of the Operations Manual includes information on IMS/360 system distribution and handling, on planning the IMS/360 system, and on implementing the system. System examples and a sample problem are supplied; a chapter on status codes and completion codes is also included.

IBM

Second Edition (July 1970)

This edition applies to Version 1, Modification Level 1, of Program Product Information Management System/360 for the IBM System/360, 5736-CX3.

This is a major revision obsoleting H20-0635-0. Besides correcting errors, this edition contains additions and changes supporting Release 18 of the Operating System.

Here is a summary of the major new and changed items in this revision.

- Sample problem procedures and examples are expanded and clarified.
- The system definition example is replaced to correspond to APAR changes.
- The illustration of the system configuration is corrected.

Other changes to the text and small changes to illustrations are indicated by a vertical line to the left of the change; changed or added illustrations are denoted by the symbol ● to the left of the caption.

This edition applies to Release 18 of IBM System/360 Operating System and to all subsequent releases until otherwise indicated in new editions or Technical Newsletters.

Changes are continually made to the specifications herein. Therefore, before using this publication, consult the latest System/360 SRL Newsletter (GN20-0360) for the editions that are applicable and current.

Copies of this and other IBM publications can be obtained through IBM branch offices.

A form for readers' comments is provided at the back of this publication. If the form has been removed, address comments to: IBM Corporation, Technical Publications Department, 112 East Post Road, White Plains, New York 10601.

CONTENTS

The Systems Operation Manual is one of a set of manuals prepared to define the various functions and personnel relationships involved in the implementation and system operation of Information Management System/360 (IMS/360). It also includes the IMS/360 sample problem (Chapter 8). The other manuals in the set are:

IMS/360 Application Description Manual (GH20-0524)

IMS/360 Program Description Manual (SH20-0634)

IMS/360 Operations Manual, Volume II - Machine Operations (SH20-0636)

IMS/360 System Manual, Volume I - Program Logic (LY20-0431)

IMS/360 System Manual, Volume II - Flowcharts (LY20-0432)

This introductory chapter restates some of the same information found in the introductory chapter of the Program Description and Machine Operations Manuals.

The necessity for these manuals became apparent during the design phase of the IMS/360 system. The usual mix of data processing personnel normally provides for application programming, system programming, and machine operations functions. With the introduction of IMS/360, however, the need for a fourth function, a coordinating force in implementing, administering, and maintaining the system, became apparent. The function is the "heart" of the IMS/360 system and has been designated the "Systems Operation" function. The Systems Operation function and its interface with other functions are delineated in this manual (see Figure 1).

An understanding of the following is a prerequisite for a thorough comprehension of this manual:

IMS/360 Application Description Manual

IMS/360 Program Description Manual

IMS/360 Operations Manual, Volume II - Machine Operations

IMS/360 Application Directory

OS/360 COBOL or PL/I Language (GC28-6516 or GC28-8201)

OS/360 Supervisor and Data Management Services (GC28-6646)

OS/360 Supervisor and Data Management Macro Instructions (GC28-6647)

OS/360 Basic Telecommunication Access Method (GC30-2004)

OS/360 System Programmer's Guide (GC28-6550)

OS/360 System Generation (GC28-6554)

OS/360 Job Control Language (GC28-6539)

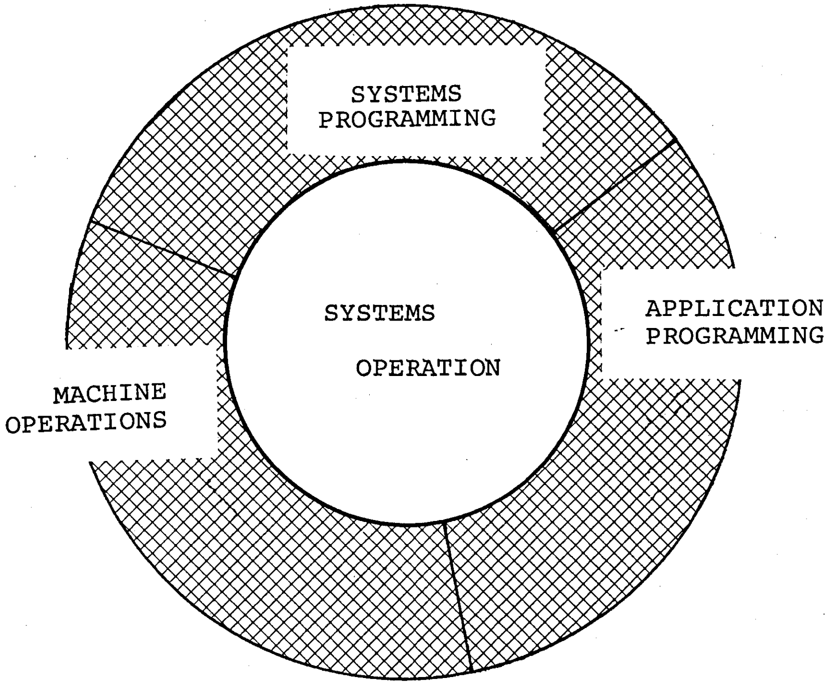


Figure 1. IMS/360 functional relationships

SYSTEMS OPERATION FUNCTION

The function of Systems Operation is:

- Configuration planning, for all purposes, of new applications so that communication lines, consoles, and software are available to support approved applications

- Responsibility for control over and approval of all new data base designs and descriptive control blocks

- Maintenance of the data bases under Data Language/I, including all control, allocation, and data base creation and reorganization

- Maintenance of a catalog of programs "certified" to operate as message processing programs under IMS/360, including related documentation, processing priorities, transaction codes, control blocks, etc.

- Responsibility to provide the capability for reconstruction and recovery of IMS/360 and its associated data bases when routine procedures known and understood by the Machine Operations function are insufficient for such recovery and reconstruction. The Systems Operation function also has the responsibility to be available to participate in such extraordinary operations whenever they are required.

- Responsibility for the utility programs that process the IMS/360 system log tapes and for causing these programs to process the log tapes and to yield accounting information, machine operations statistics, usage and data base statistics, and certain management reports on utilization and errors incurred. The function also has the responsibility for auditing these reports for quality and for assigning certain reports to other functions for analysis, as appropriate.

- Accounting and billing for IMS/360 and message programs and a background batch program in the IMS/360 environment. Statistics from the system log tape reflecting activity by system, transaction type, terminal, line, etc. are also distributed.

- Responsibility for IMS/360 system definition and modification

- Maintenance of all IMS/360 documentation

SYSTEMS PROGRAMMING FUNCTION

The functions of Systems Programming encompass the following:

- Assistance and participation in the hardware installation, test, and initial operations of any new equipment or changed configurations

- Consultation with IMS/360 application programmers in conjunction with the Systems Operation function to assist in the integration of applications with IMS/360

- Software maintenance and improvement of IMS/360 utility programs and modifications to Operating System/360

MACHINE OPERATIONS FUNCTION

In addition to the usual operational assignments, the Machine Operations function is responsible for:

- All master terminal capabilities in accordance with established procedures, with especially prepared instructions to cover extraordinary happenings

- Assisting terminal operators at remote terminals in the initial diagnoses of apparent problems, whether they are concerned with the remote terminal, the connecting communication line, the central hardware, the central software, or message processing application programs. After the initial diagnoses, the Machine Operations function should have accumulated sufficient information to determine whose assistance is required and to intelligently describe the problem, and can assist in determining the degree of emergency sustained.

APPLICATION PROGRAMMING FUNCTION

The Systems Operation function provides for applications planning, implementation, and audit. The application programming function must consider the following in its analysis of a proposed application:

- Configuration and storage device requirements for anticipated applications

- Data base structuring, storage device cost/performance tradeoffs, and commonality of data with existing data bases

- Program structuring, to include core storage requirements, duration of execution, overlay structure, and program chaining

- Message formats and length, transaction types, priorities, passwords, and logical terminal names

- Schedule of data base checkpoints, and checkpoint cost versus reconstruction cost

- Schedule of data base dumps and reorganization

SYSTEMS OPERATION CHECKLIST

A Systems Operation checklist is provided here as a further aid to the reader in understanding the tasks of the Systems Operation function.

The items or tasks in the checklist are enumerated in detail later in this manual. Implementation of these tasks is also described. Examples and possible error conditions in the performance of the Systems Operation function are also given.

This checklist is not ordered chronologically.

It is directed toward the information needed from a single application. Of course, under IMS/360, there is in all probability more than one application program proposed or in operation. Each application program must therefore be checked off against this list.

The following is an explanation of the columns of the checklist:

Column 1. The checklist item under consideration.

Column 2. Is a teleprocessing application program affected by this item? (X means YES.)

Column 3. Is a batch application program affected?

Column 4. Is the Systems Operation function affected by this item?

Column 5. Is IMS/360 DBD generation affected by this item? (X means YES; entry in Macro column indicates which macro; entry in Operand column indicates the operand of the macro.)

Column 6. Is IMS/360 PSB generation affected by this item? (X means YES; entry in Macro column indicates which macro; entry in Operand column indicates the operand of the macro.)

Column 7. Is IMS/360 system definition affected by this item? (X means YES; entry in Macro column indicates which macro; entry in Operand column indicates the operand of the macro.)

Column 8. Is the IMS/360 security maintenance program affected by this item? (X means YES; entry in Control column indicates which control statement; entry in Data column indicates which data statement.)

Column 9. In which manual can more details be found about this particular item?

Abbreviation	Full Title
SOM	IMS/360 Operations Manual, Volume I - Systems Operation
MOM	IMS/360 Operations Manual, Volume II - Machine Operations
PDM	IMS/360 Program Description Manual
SM	IMS/360 System Manual, Volume I - Program Logic
OS/360	Appropriate Operating System/360 Manual

SYSTEMS OPERATION CHECKLIST

①	②	③	④	⑤		⑥		⑦		⑧		⑨
				DBDGEN		PSBGEN		SYSTEM DEFINITION		SECURITY MAINTENANCE		DETAIL IN WHICH
ITEM	TELE-PROCESSING	BATCH	PLANNING	MACRO	OPERAND	MACRO	OPERAND	MACRO	OPERAND	CONTROL	DATA	MANUAL
1. When consulting with the application programming function about the application program structure, have the following been considered?												PDM, SOM
a. Core limits	X	X	X									SOM
b. Overlay structure	X	X	X									OS/360
c. Program chaining	X	X	X									PDM
d. IMS/360 restart	X	X	X									PDM, SOM
e. Storage devices	X	X	X									PDM, SOM
2. Select Type I programming systems (MVT, MFT-II, or PCP).	X	X	X					IMSCTRL APPLCTN	SYSTEM PGMTYPE			SOM
3. Select type of IMS/360 processing region (Type 1 or 2 or 3).	X	X	X					IMSCTRL	SYSTEM (INDIRECT-LY)			PDM, SOM
4. Select how many regions or partitions all applications will need at one time.			X					IMSCTRL	MAXREGN			PDM, SOM
a. Within those regions or partitions, how many requests are anticipated? (Terminal I/O, Message Queues, and DL/I data base requests)			X					IMSCTRL	MAXIO			PDM, SOM
5. Select application program name and check for duplication.	X	X				PSBGEN	PSBNAME	APPLCTN	PSB(name))	(PROGRAM	PASSWORD	PDM, SOM
6. Which application program language has been selected?	X	X	X			PSBGEN	LANG					PDM
7. Has enough information been provided about the selected telecommunications system for IMS/360 teleprocessing environment analysis?			X									PDM, SOM MOM
a. Terminal hardware and network (including lines)			X					LINEGRP	UNITYPE			PDM, SOM MOM
b. Specify transaction codes for application program.	X	X	X					TRANSACT	CODE	) (TRANSACT	PASSWORD TERMINAL	PDM, SOM MOM
(1) Specify priority for each transaction code.			X					TRANSACT	PRTY			PDM, SOM MOM
(2) Is this TRANSACT code an Inquiry type or not?			X					TRANSACT	INQUIRY			
c. Are messages to be entered at remote terminals single- or multiple-line?	X		X					TRANSACT	MSGTYPE			PDM,SOM MOM
(1) Select whether, after input of message, terminal is to continue input of other messages or wait until previous message has been processed.			X					TRANSACT	MSGTYPE			PDM,SOM MOM

SYSTEMS OPERATION CHECKLIST

① ITEM	② TELE-PROCESSING	③ BATCH	④ PLANNING	⑤ DBDGEN MACRO	DBDGEN OPERAND	⑥ PSBGEN MACRO	PSBGEN OPERAND	⑦ SYSTEM DEFINITION MACRO	SYSTEM DEFINITION OPERAND	⑧ SECURITY MAINTENANCE CONTROL	SECURITY MAINTENANCE DATA	⑨ DETAIL IN WHICH MANUAL
(2) Have message formats and length been reviewed?	X		X									PDM
d. Specify the length of time to process the message.	X		X					TRANSACT	PROCLIM			SOM
e. Specify the number of messages to be processed per application program load in a region.	X		X					TRANSACT	PROCLIM			SOM
f. Specify the line groups for the same terminal types.			X					LINEGRP LINE	DDNAME ---			SOM
(1) Specify whether line group is dialup (switched); if so, specify telephone number.								SUBPOOL LINEGRP LINE POOL	FEAT FEAT			SOM
g. If 1050 system, specify whether station control/ switched or station control/non-switched.			X					LINEGRP LINE	FEAT FEAT			SOM
(1) If station control/ switched, specify Autoanswer.			X					LINEGRP LINE	FEAT FEAT			SOM
(2) If station control/ nonswitched, option is Autopoll.			X					LINEGRP LINE	FEAT FEAT			SOM
h. If 2740 system, specify station control/non-switched or no station control (transmit control)/ switched.			X					LINEGRP LINE	FEAT FEAT			SOM
(1) If station control/ nonswitched, option is AUTOPOLL or POLL.			X					LINEGRP LINE	FEAT FEAT			SOM
(2) If no station control (transmit control)/ switched, option is AUTOANSWER.			X					LINEGRP LINE POOL	FEAT FEAT FEAT			SOM
i. Specify each communication line by number with physical terminals and logical terminal names, their features, their addresses, and their component addresses. Check for duplication of names.	X		X			PCB PCB PCB	TYPE=TP LTERM TYPE=DB	LINE TERMINAL NAME NAME	ADDR ADDR(PHY) ltermname COMP	) (TERMINAL (SEE 8)) (PTERM PASSWORD) (COMMAND	PASSWORD TERMINAL	SOM SM

SYSTEMS OPERATION CHECKLIST

① ITEM	② TELE-PROCESSING	③ BATCH	④ PLANNING	⑤ DBDGEN		⑥ PSBGEN		⑦ SYSTEM DEFINITION		⑧ SECURITY MAINTENANCE		⑨ DETAIL IN WHICH MANUAL
				MACRO	OPERAND	MACRO	OPERAND	MACRO	OPERAND	CONTROL	DATA	
j. Describe your input and output queue control record and message data sets desired. Also are reusable queues required?			X					IMSCNTRL MSGQUEUE MSGQUEUE MSGQUEUE MSGQUEUE MSGQUEUE	MSGBUFF QCRIN QCROUT MSGIN MSGOUT REUSE			SOM
k. Specify master terminal name after giving consideration to master terminal operation relationship. Check duplication.	X	X	X					MASTERM	logical name			APM SOM MOM
8. Specify password and terminal security.	X		X							) (PASSWORD) (TERMINAL	TERMINAL TRANSACT COMMAND DATABASE PROGRAM PTERM PASSWORD TRANSACT COMMAND	MOM SOM
9. Specify all data base names for each application program.	X	X	X	DBD	NAME	PCB	DBDNAME	DATABASE	DBD(name) (see item 10 below)	) (DATABASE	PASSWORD	PDM SOM
a. Specify what type of processing region.	X	X	X					IMSCNTRL	SYSTEM (indirectly)			PDM SOM
b. Specify how application program intends to use each data base (read-only, update, exclusive use).	X	X	X					DATABASE	INTENT			PDM SOM
(1) Also specify application program options (get, delete, insert, replace, load).	X	X	X			PCB	PROCOPT					PDM SOM
c. Has consideration been given to logging all segments against a data base for data base "backout" during emergency restart?								DATA BASE	LOG			PDM SOM

SYSTEMS OPERATION CHECKLIST

① ITEM	② TELE-PROCESSING	③ BATCH	④ PLANNING	⑤ DBDGEN MACRO	DBDGEN OPERAND	⑥ PSBGEN MACRO	PSBGEN OPERAND	⑦ SYSTEM DEFINITION MACRO	SYSTEM DEFINITION OPERAND	⑧ SECURITY MAINTENANCE CONTROL	SECURITY MAINTENANCE DATA	⑨ DETAIL IN WHICH MANUAL
10. Specify what access method is wanted for each data base; their data set names and their storage types.	X	X	X	DBD DMAN DMAN DMAN	ACCESS DD1 DD2 DEV1							PDM SOM
a. Is the organization of each data base the most efficient concerning: (1) Prime (2) Overflow (3) Logical record length (4) Blocking factor (5) Multiple data set Groups and single data set group (6) HISAM and HSAM (7) BISAM and QISAM (8) Variable-length data base record processing	X	X	X	X DMAN DMAN	 LRECL BLKFACT							SOM
11. Specify the application programs hierarchical (sensitive) segments and parent relationships.	X	X	X	SEGM SEGM SEGM SEGM	NAME PARENT BYTES FREQ	SENSEG	SENSEG-name parent-seg-name					PDM SOM
a. Describe in detail the (sensitive) segments.	X	X	X	FLDK & FLD	NAME TYPE BYTES START	PCB	KEYLEN					PDM SOM
b. Is an adequate history of these relationships being maintained for analysis of the statistics reports?			X									SOM
12. Check the entry point to the application program.	X	X										PDM
a. If PL/I, the load module ENTRY must be either IHESAPB or IHESAPD.			X									PDM
13. Plan and specify the statistics reports from IMS/360 system.	X	X	X									PDM SOM MOM
14. Plan the residence of MACLIB, RESLIB, PGMLIB, PSBLIB, DBDLIB, and PROCLIB.			X					MACLIB MACLIB MACLIB MACLIB RESLIB RESLIB RESLIB PGMLIB PGMLIB PGMLIB PSBLIB PSBLIB PSBLIB DBDLIB DBDLIB DBDLIB PROCLIB PROCLIB PROCLIB	UNIT VOLNO PDS COPY PDS UNIT VOLNO UNIT VOLNO PDS UNIT VOLNO PDS UNIT VOLNO PDS UNIT VOLNO PDS			SOM MOM SM

SYSTEMS OPERATION CHECKLIST

①	②	③	④	⑤		⑥		⑦		⑧		⑨
				DBDGEN		PSBGEN		SYSTEM DEFINITION		SECURITY MAINTENANCE		DETAIL IN WHICH
ITEM	TELE-PROCESSING	BATCH	PLANNING	MACRO	OPERAND	MACRO	OPERAND	MACRO	OPERAND	CONTROL	DATA	MANUAL
15. Are the libraries and procedures set up in accordance with this plan? (See 14)			X									
16. Specify data sets, volumes, I/O devices required for System Definition.			X					IMSGEN IMSGEN IMSGEN	UT1SDS ASMPRT LEPRT			SOM MOM SM
17. Specify numbers for OS/360 Type 1 SVC's (for inter-region communication).			X					IMSCNTRL	COMMSVC			SOM MOM SM
18. Specify OSAM channel end appendage load module member name-(IGG019xx).			X					IMSCNTRL	OCENDA			SOM MOM SM
19. Specify the user SVC number to be the OSAM type 2 SVC numbers.			X					IMSCNTRL	OSAMSVC			SOM SM
20. Has the DBD generation been executed? Items 9, 10, 10a, 11, 11a, and 20 must have been complete.	X	X	X	PRINT DBDGEN FINISH END	NOGEN -- -- --							PDM SOM
21. Has PSB generation been executed? Items 5,6,7i,9,9b(1),11, 11a,&21 must have been complete.	X	X	X			END	--					PDM SOM
22. Has System Definition been completed? Stage 1? Stage 2? Items 2,3,4,4a,5,7a,7b, 7b(1),7d,7e,7f,7f(1),7g, 7g(1) & (2),7h,7h(1) & (2),7i,7j,7k,9,9a,9b 14,16,17,18,19 must have been completed.	X	X	X					IMSGEN IMSGEN IMSGEN	UT1SDS ASMPRT LEPRT			SOM SM
23. Is the Security Maintenance Program required for this application? If yes, has SMP been completed? Items 5, 7b, 7i, 8 & 9.	X	X	X							) (PROGRAM - - -) (TRANSACT- - -) (TERMINAL- - -		SOM SM

SYSTEMS OPERATION CHECKLIST

①	②	③	④	⑤		⑥		⑦		⑧		⑨
				DBDGEN		PSBGEN		SYSTEM DEFINITION		SECURITY MAINTENANCE		DETAIL IN WHICH
ITEM	TELE-PROCESSING	BATCH	PLANNING	MACRO	OPERAND	MACRO	OPERAND	MACRO	OPERAND	CONTROL	DATA	MANUAL
24. Concerning the Master Terminal and the machine operation function: a. Have the instructions to the MT operator about the types of checkpoints been delineated?			X									SOM MOM
b. Have the instructions to the MT operator about the restart procedures been delineated?			X									SOM MOM
c. Has an operating plan been worked out between the MT operator and the computer console operator for the system log tapes?			X									SOM MOM
d. Are the types of system shutdown procedures described?			X									SOM MOM
e. Have the instructions for alternate master terminals been delineated?			X									SOM MOM
f. Is the remote terminal trouble procedure adequate?			X									PDM SOM MOM
g. Have adequate IPL instructions been delineated?			X									SOM MOM
h. Has a group of command language verbs been restricted to entry from the master terminal?										) (COMMAND	VERB	
25. Make necessary coordination to handle system ABENDS, error conditions, and trouble reports.			X									PDM SOM MOM
26. Have schedules been planned for system checkpoint, data base dumps, and reorganization?			X									SOM MOM
a. Has a number been specified that indicates a checkpoint log frequency?			X					IMSCTRL	CKPT			SOM MOM

SYSTEM DISTRIBUTION

The distribution of Information Management System/360 (IMS/360) is made on unlabeled, nine-track, 800-bpi or 1600-bpi magnetic tape, or unlabeled, seven-track, 800-cpi magnetic tape. The seven-track tape requires the data conversion feature. The distribution is composed of two tapes. The basic distribution tape includes two data sets:

- IMS/360 Macro-Definition Library (IMS.GENLIB)
- IMS/360 Load Module Library (IMS.LOAD)

The optional distribution tape includes one data set, which should be ordered:

- IMS/360 Source Module Library (IMS.SOURCE)

The nine-track tape distribution is recommended because two nine-track tapes are required for IMS/360 teleprocessing execution. The three data sets are unloaded copies of direct access partitioned data sets. They have been moved to tape using the IBM Operating System/360 IEHMOVE utility program. When the IMS/360 user receives the IMS/360 distribution tape(s), the IEHMOVE program should be employed to copy these data sets to direct access storage (Figure 2). The following Job Control Language statements and utility control cards should assist in the copy execution. The user should allocate each of the IMS/360 distribution libraries before the move to disk. See the IMS/360 Application Directory for recommended space allocation on direct access devices. The DCB attributes for the IMS.GENLIB and IMS.SOURCE data sets should be the same as SYS1.MACLIB. The DCB attributes for the IMS.LOAD data sets should be the same as SYS1.LINKLIB.

The IMS/360 sample problem as defined in Chapter 8 uses the same Job Control Language statements and utility control statements as listed here.

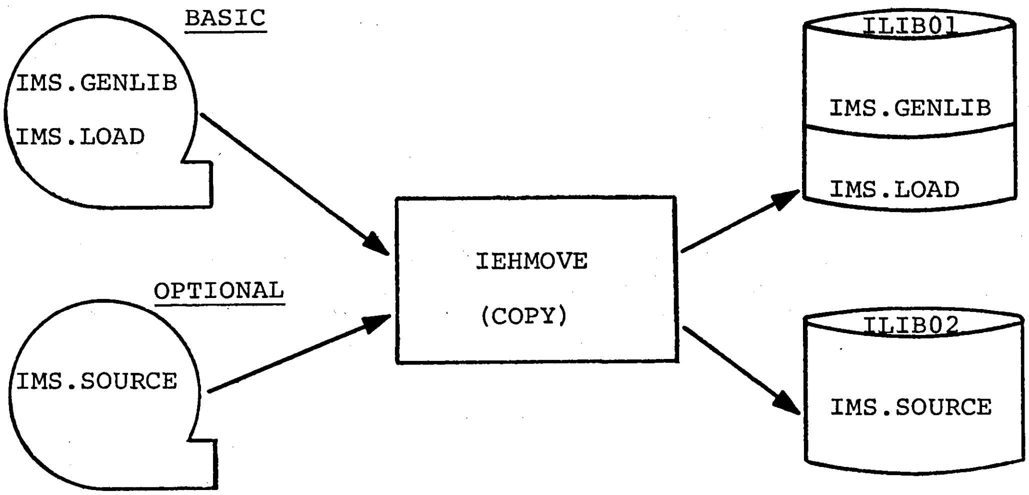


Figure 2. IEHMOVE to direct access storage device

Nine-Track Tape

```
//COPY        JOB    848,NAME,MSGLEVEL=1
//            EXEC   PGM=IEHMOVE,REGION=100K
//SYSPRINT    DD     SYSOUT=A
//SYSUT1      DD     UNIT=2311,DISP=OLD,VOLUME=SER=111111
//TAPE1       DD     UNIT=(2400-4,,DEFER),DISP=OLD,             X
//                   VOLUME=SER=SCRTCH,DCB=(LRECL=80,           X
//                   RECFM=FB,BLKSIZE=800,DEN=2),LABEL=(,NL)
//DISK1       DD     UNIT=2311,DISP=OLD,VOLUME=SER=ILIB01
//TAPE2       DD     UNIT=(2400-4,,DEFER),DISP=OLD,
//                   VOLUME=SER=SCRTCH,DCB=(LRECL=80,           X
//                   RECFM=FB,BLKSIZE=800,DEN=2),LABEL=(,NL)
//DISK2       DD     UNIT=2311,DISP=OLD,VOLUME=SER=ILIB02
//SYSIN       DD     *
              COPY   PDS=IMS.GENLIB,                            X
                     FROM=2400-4=(SCRTCH,1),                    X
                     TO=2311=ILIB01,FROMDD=TAPE1
              COPY   PDS=IMS.LOAD,FROM=2400-4=(SCRTCH,2),       X
                     TO=2311=ILIB01,FROMDD=TAPE1
              COPY   PDS=IMS.SOURCE,FROM=2400-4=(SCRTCH,1),     X
                     TO=2311=ILIB02,FROMDD=TAPE2
/*
```

Seven-Track Tape

```
  //COPY2        JOB     848,NAME,MSGLEVEL=1
  //             EXEC    PGM=IEHMOVE,REGION=100K
  //SYSPRINT     DD      SYSOUT=A
  //SYSUT1       DD      UNIT=2311,DISP=OLD,VOLUME=SER=111111
  //TAPE1        DD      UNIT=(2400-2,,DEFER),DISP=OLD,         X
  //                     LABEL=(,NL),                           X
  //                     VOLUME=SER=SCRTCH,DCB=(LRECL=80,       X
  //                     RECFM=FB,BLKSIZE=800,DEN=2,TRTCH=C)
  //DISK1        DD      UNIT=2311,DISP=OLD,VOLUME=SER=ILIB01
  //TAPE2        DD      UNIT=(2400-2,,DEFER),                  X
  //                     DISP=OLD,LABEL=(,NL),                  X
  //                     VOLUME=SER=SCRTCH,DCB=(LRECL=80,       X
  //                     RECFM=FB,BLKSIZE=800,DEN=2,TRTCH=C)
  //DISK2        DD      UNIT=2311,DISP=OLD,VOLUME=SER=ILIB02
  //SYSIN        DD      *
                 COPY    PDS=IMS.GENLIB,TO=2311=IBIL01,         X
                         FROM=2400-2=(SCRTCH,1),FROMDD=TAPE1
                 COPY    PDS=IMS.LOAD,TO=2311=ILIB01,           X
                         FROM=2400-2=(SCRTCH,2),FROMDD=TAPE1
                 COPY    PDS=IMS.SOURCE,TO=2311=ILIB02,         X
                         FROM=2400-2=(SCRTCH,1),FROMDD=TAPE2
/*
```

Those parameters which are underlined are user-specifiable (for example, 2314 rather than 2311). The region parameter is required only for Operating System/360 MVT execution. Generic name 2400-4 is nine-track at 800-bpi with DCB=(DEN=2); generic name 2400-2 is seven-track with data conversion at 800-bpi with DCB=(DEN=2).

If the SYS1.MACLIB data set DCB characteristics are not 80-character records, blocked 44, a preallocated IMS.GENLIB partitioned data set should be used in the move from tape to disk. The DCB characteristics for IMS.GENLIB should be equated to SYS1.MACLIB.

SYSTEM HANDLING

Once the IMS/360 libraries have been copied from the distribution tape(s) to direct access storage, the user can begin to tailor IMS/360 to his data processing environment. The tailoring of IMS/360 to a particular user's data processing environment is accomplished with the IMS/360 system definition macro-instructions contained within IMS.GENLIB. The IMS/360 system executes with a collection of control blocks that describe the user's data processing environment: application programs, data bases, communication lines and terminals, and other IMS/360 resources. The system definition process constructs these control blocks.

The IMS/360 user must prepare a control card input deck for IMS/360 system definition. The control card types and formats are described later in this manual. Once the control card deck has been prepared, it is appended to a package of Job Control Language for the macro-instruction assembly of system definition. System definition is required if either an online message processing and batch processing or a batch-only processing system is desired (Figure 3).

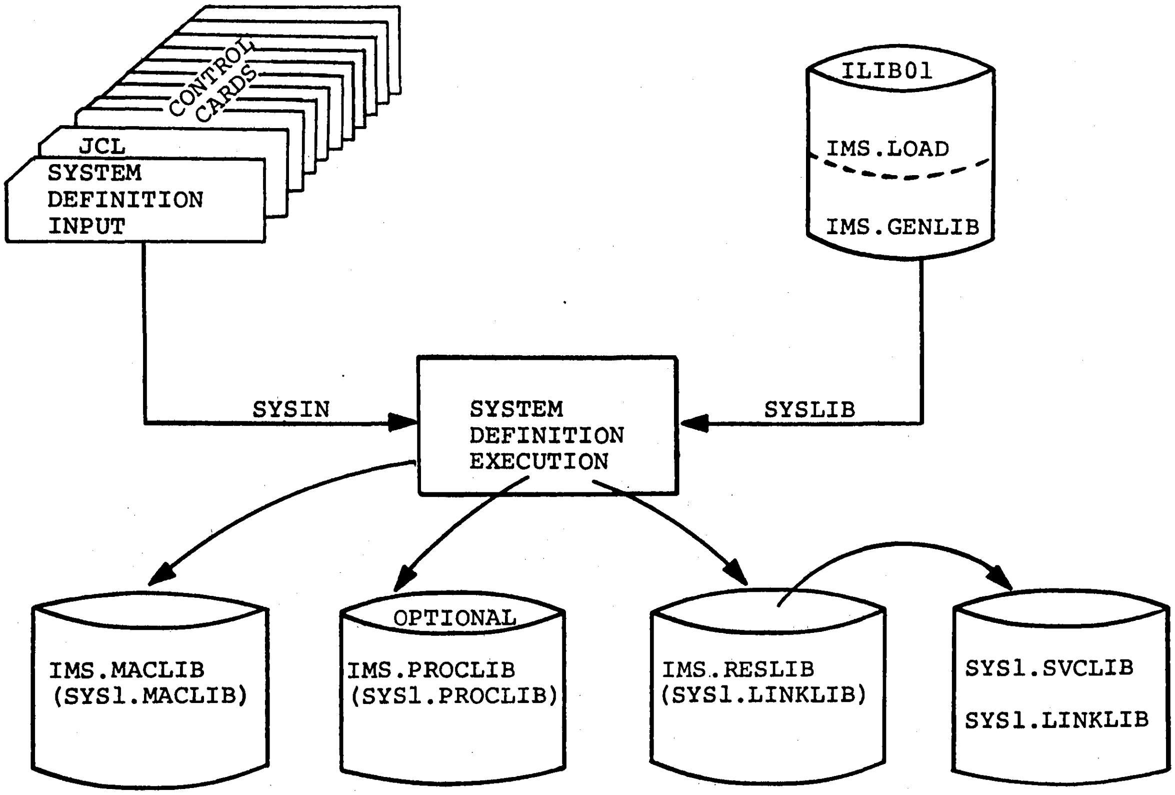


Figure 3. System definition handling

The output from IMS/360 system definition includes:

- Generation and placement in the user-named target load library of IMS/360 control program control blocks

- Generation of the IMS/360 control program nucleus into the user-named target load library

- Generation and placement in the user-named target load library of the Type 3 region Data Language/I batch processing nucleus

- The linkage edit of three user supervisor calls (SVC's), two of which are used for interregion communication and one for OSAM multivolume execution. These are placed in the RESLIB target library. The user can specify the desired SVC numbers.

- The naming and creation of the OSAM channel end appendage module in the RESLIB library. The user can specify the module name and must move the module to SYS1.SVCLIB.

- The moving of procedures to user or SYS1.PROCLIB. These procedures are used for data base description (DBD) and program specification block (PSB) generation, IMS/360 execution, message region execution, batch region execution, etc.

- The creation of IMS.MACLIB

Once IMS/360 system definition has been performed, the three SVC routines must be link-edited with the Operating System/360 nucleus.

The IMS/360 user must have provided for two Type 1 user SVC routines and one Type 2 user SVC routine in his Operating System/360 system generation. If the system definition is for Type 3 region batch execution only, the Type 2 user SVC is all that is required. This SVC is used by OSAM. The procedure for relink-editing the Operating System/360 nucleus with the user SVC routines is specified in Chapter 4 of this manual.

Once system definition and the SVC-Operating System/360 nucleus link-edit is performed, the user must allocate direct access space for the DBD and PSB libraries. In addition, if online processing is desired, space should be allocated for message queue data sets. All these data sets should be cataloged. Chapters 3 and 4 of this manual describe the execution of these functions.

Finally, the user is ready to create DBD's, PSB's, and application programs. Before any message processing may be performed, the required data bases must be created in the Type 3 region batch environment.

SYSTEM MAINTENANCE

Permanent modifications and corrections to problems encountered with the IMS/360 system will be provided with updated program modules through a new mod-level distribution of all these IMS/360 libraries. These mod-level distributions will be provided on a periodic basis. The IMS.SOURCE library will provide a vehicle for convenient, quick maintenance of PTF's (Program Temporary Fix). If the system user orders and maintains the source module library, corrections to erroneous IMS/360 modules can be distributed over the SECOM network. This maintenance distribution of PTF's through SECOM is in a format acceptable to the Operating System/360 utility program IEBUPDTE. The IMS/360 user will update the appropriate IMS/360 source or macro-definition member, assemble the affected modules, and link-edit the new copy of the module into IMS.LOAD. The IMS/360 system definition may have to be performed again if macro-definition statements have changed. For source-only update distributions, reprocessing of IMS/360 load modules by the linkage editor will ordinarily suffice.

IMS/360 PROCESSING REGIONS

This section of Chapter 3 describes the system flow of control within and between each type of IMS/360 system region or partition. Region types are described in the IMS/360 Program Description Manual. The communication between regions when necessary is supplied through two user-defined Type 1 supervisor call routines (SVC's). The control within the Type 0 IMS/360 control program region, where multiple input/output operations are occurring asynchronously, is provided by use of the Operating System/360 multiple wait capability. To assure | compatibility between Operating System/360 MFT and MVT, the IMS/360 control program does not use Operating System/360 multitasking to execute asynchronous input/output operations.

Type 0 and 1 Processing Regions System Flow

Once the Type 0 region containing the IMS/360 control program and one or more Type 1 regions to be utilized for the message processing have been initiated by the job management facilities of Operating System/360, the following system flow occurs ("Events" refer to Figure 4):

1. The control facility of the IMS/360 program region gives control to the telecommunications facility (Event 1) for communication with the master terminal. From the master terminal, commands (Event 20) are entered to enable communication with all user terminals and to restart the system (Event 24).

2. The restart facility, using a previous copy of the system log, restarts the system with current status and any outstanding messages from the previous system execution. The system can be started without a previous system log if the cold start option is taken.

3. The restart facility returns (Event 24) to the telecommunications facility, which enables the master terminal operator to initiate communication to all user terminals.

4. The telecommunications facility returns to the control facility (Event 1). When an input message or message line is received (Event 2), the telecommunications facility is again given control (Event 1). The common service facility is invoked (Event 3), the input message is queued and logged (Event 4), and control returns through the telecommunications facility to the control facility.

5. Steps 3 and 4 are repeated until an entire message is received. Upon receipt of an entire message, the telecommunications facility notifies the message scheduling facility of input available to scheduling for processing (Event 5).

6. The control facility in a message processing region notifies the control facility in the IMS/360 control program region (Event 6), through a resident SVC, that a message processing region is available for scheduling.

7. When there are input messages pending and a message region is available for scheduling, control is passed to the message scheduling facility (Event 7) to determine the application message processing program to be scheduled. Control is returned through the control facility by another resident SVC (Event 8) to

the message region. The application program is loaded and given control (Event 9).

8. The message processing application program subsequently makes requests for the input message, and possibly for data base references (Event 10), through the control facility (Event 6). The control facility passes control to the Data Language/I facility (Event 11) either for data base reference (Event 16) or for message reference (Event 13). The message reference is accomplished through the common service facility (Event 12).

9. Whether the data represents a message or a data base segment, it is communicated to the application program (Event 14), and control is returned to the application program (Events 11, 8, 15).

10. The application program uses the same routine of control (Events 10, 6, 11, 13) for sending output messages.

11. When the application program terminates, control is returned to the control facility (Event 26) in the IMS/360 control program region (Event 6).

12. The control facility passes control to the message scheduling facility (Event 7), which notifies the telecommunication facility (Event 27) of pending output.

13. Subsequently, control passes to the telecommunication facility (Event 1) to allow output messages to be transmitted. The common service facility is invoked (Event 3), and the message is retrieved from the message queue (Event 4) and transmitted (Event 8).

14. At periodic intervals, based upon either message volume or notification from the master terminal (Event 20), a checkpoint of the system occurs. The telecommunications facility gives control to the checkpoint facility (Event 21) for writing status on the system log. The common service facility is invoked for this purpose (Event 23).

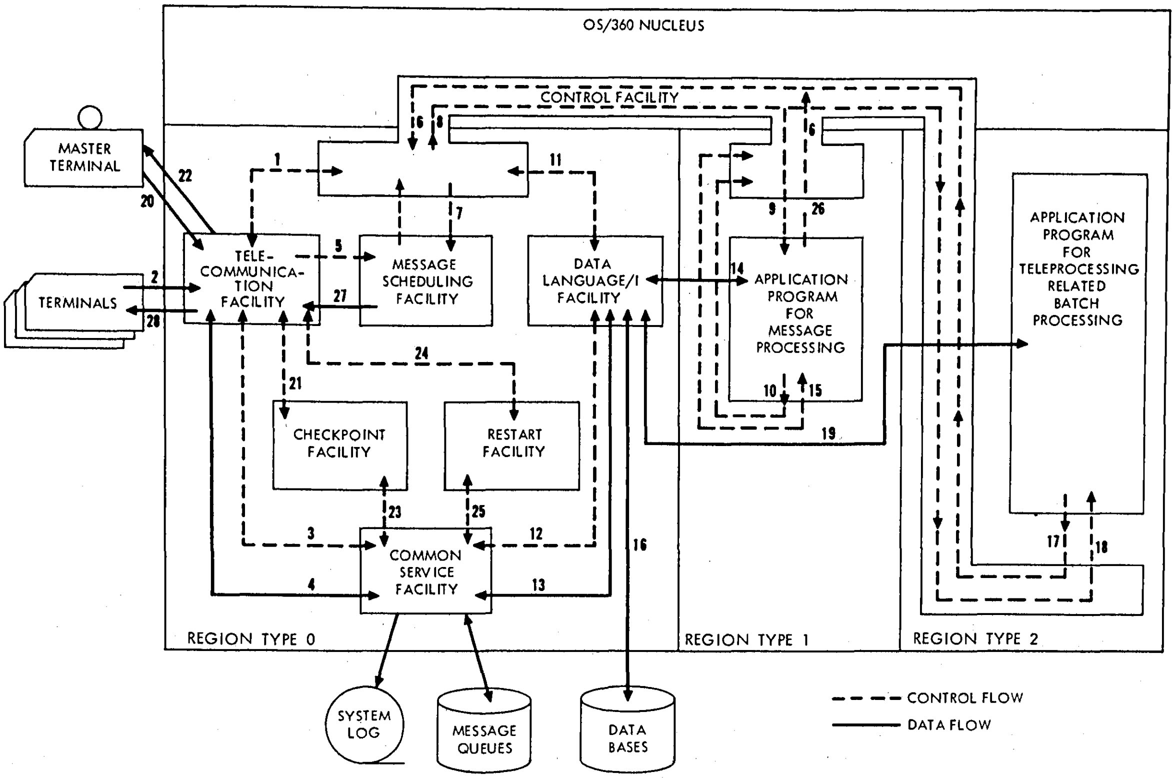


Figure 4. IMS/360 Type 1 and Type 2 processing region system flow

Type 2 Processing Region System Flow

Once the IMS/360 regions associated with teleprocessing have been initiated by Operating System/360, a Type 2 processing (batch) region can be initiated. This Type 2 processing region may contain an application program for processing against teleprocessing data bases in the batch manner. The application program in the batch region is scheduled by Operating System/360 job management, but may utilize the Data Language/I facility for teleprocessing data base reference (Figure 4). An application program executed in a Type 2 processing region can only access IMS/360 data bases which are defined in the IMS/360 Type 0 region ("Events" refer to Figure 4).

1. Any data reference is initiated by the batch application program (Event 17) through the control facility (Event 6).

2. Control is given to the control facility in the IMS/360 control program region. Control is then passed to the Data Language/I facility (Event 11) to reference the data bases (Event 16).

3. The data base segment requested is supplied to the application program (Event 19), and control returns through the control facility to the application program (Events 11, 8, 18). The data base request may be an addition, update, or deletion of a data base segment. The flow of control is identical in each case; however, the data base segment is supplied from the application program (Event 19) to the data base (Event 16).

A Type 2 processing region, in addition to being able to process data bases used for message processing, has access to input message queues and can output to the message queues. The access to the input message queues is provided by specifying a transaction type to which access is desired in the Job Control Language (JCL) for a Type 2 region. Access to the output message queues is provided by specifying output terminal PCB's in the PSB for the application program which executes in the Type 2 processing region. The IMS/360 Program Description Manual describes output terminal PCB's. The JCL used for a Type 2 processing region is specified in Chapter 4 of this manual.

Type 3 Processing Region System Flow

Whether or not the teleprocessing capabilities of IMS/360 exist among jobs executing under Operating System/360, the Data Language/I facility of IMS/360 can be used in a nonteleprocessing data base batch environment ("Events" refer to Figure 5).

1. The application program for Type 3 processing (nonteleprocessing data base batch processing) is initiated through the job management routines of Operating System/360 (Event 1).

2. Then the Data Language/I facility is invoked (Event 2). The highest level Data Language/I module analyzes the data base call request. Depending upon the I/O function requested in the call, the insert (Event 11), the retrieve (Event 3), the load (Event 15), the HSAM (Event 16), or the delete/replace (Event 12) module is invoked. These modules subsequently invoke either the ISAM modules (Event 4) or the OSAM modules (Event 5) to reference the data base.

3. The data segment is moved to or from the application program's I/O work area and the data I/O buffers used by the access method (Events 8, 9).

4. Also, the data segment is moved to or from the I/O buffers and the data sets representing the data base (Events 6, 7, 13, 14).

5. After the Data Language/I I/O request has been completed, control is returned from either ISAM or OSAM (Events 4, 5) to one of the Data Language/I modules. Subsequently, control is returned to the Data Language/I analyzer module (Events 3, 11, 12) and finally to the application program (Event 10).

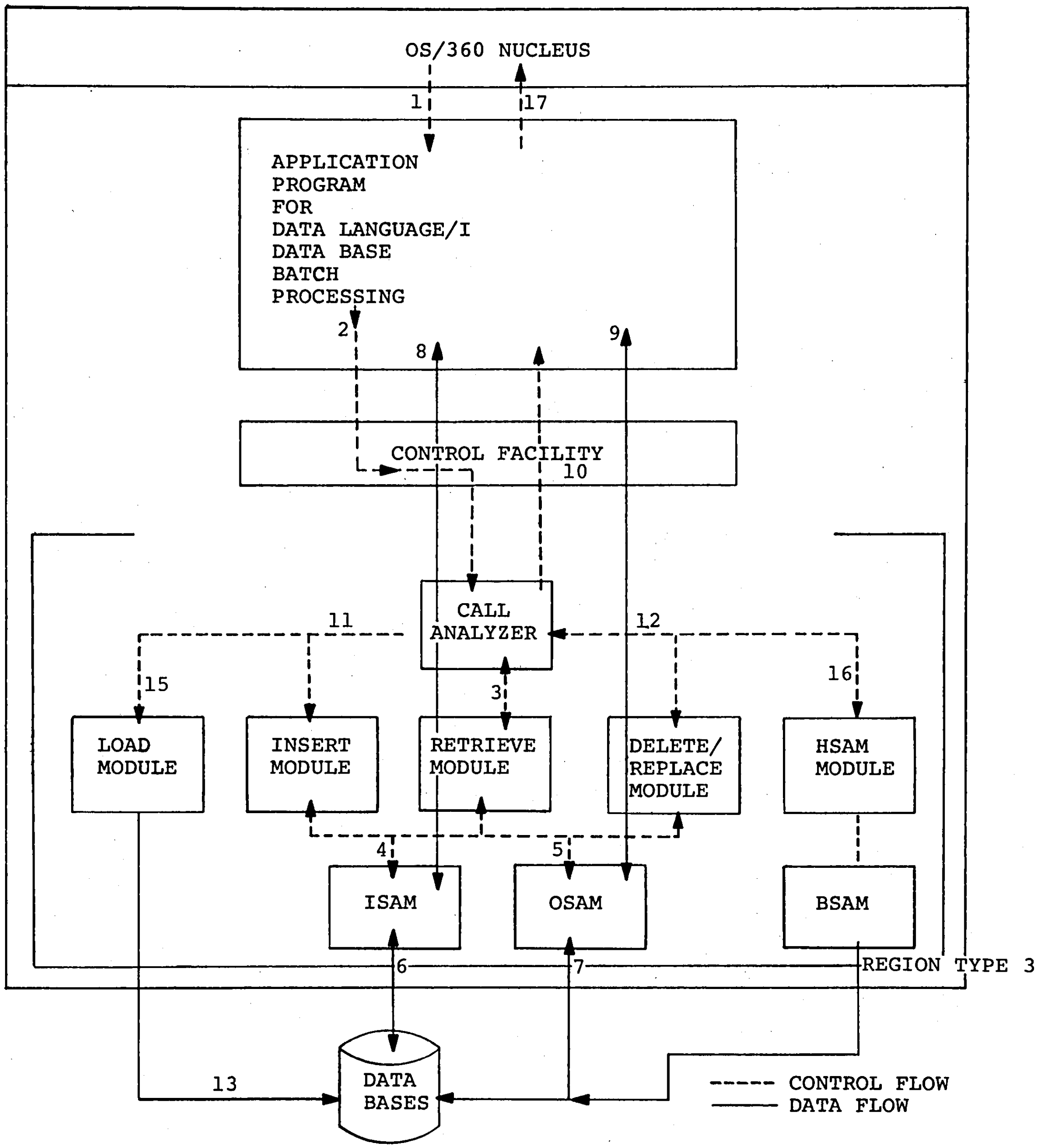


Figure 5. IMS/360 Type 3 processing region system flow

There is a capability in a Type 3 processing region to specify through Job Control Language (JCL) parameters a PSB and an application program with different names. Normally there is a one-to-one relationship between PSB and application program. However, the ability to specify different PSB's for one application program opens the door for the user of IMS/360 to create some general purpose "utility programs" which use multiple PSB's (one at a time). Chapter 4 of this manual describes the JCL available to assist in implementing this capability.

When the data bases normally used for message processing are not being used for that purpose, they may be processed in a Type 3 processing region. This is permitted when the IMS/360 control program is not operative as an Operating System/360 job.

SHARE – Both Jobs

R Only

Impact of MFT and MVT on IMS/360

Certain features in the MFT and MVT operating system options substantially affect the performance of IMS/360.

One of the major differences is the support of Operating System/360 in the area of storage management. Storage management for MVT is extended to include subpool management, whereas, under MFT, storage management resembles PCP storage management; that is, supervisor control blocks are not resident in system queue space, and modules of code are packed contiguously in an MFT partition. Under MVT, the storage management algorithm is based upon use of 2K blocks of storage as the minimum quantity that can be manipulated by the storage management facility. In addition, in MVT, FETCH= routines may make a different subpool call to storage management based upon module attributes. MFT is essentially transparent to the usability attribute of a load module. As a result, certain IMS/360 load modules will reside in protection key 0 storage within the problem program region in MVT, whereas under the MFT environment these modules will essentially be unprotected from modification. Since MVT does provide the added feature of protected code in the problem program region, it may be desirable at the time the system is defined that the IMS/360 resident library (that is, the library in which all IMS/360 executable modules will reside) be in fact either SYS1.LINKLIB or SYS1.SVCLIB. One of these two libraries should be specified, since these are the only two from which Fetch will store in protect code 0 core in the problem program region.

Another consideration in the differences between MFT and MVT is in system timing. In Type 0, 1, 2, and 3 regions, the TIME= parameter for job step timing is available, whereas in an MFT environment this capability is not implemented. This should not be considered a disadvantage of MFT; however, the user should be aware that, since the IMS/360 regions will be expected to persist for many hours of continuous operation, it is necessary to specify the TIME= parameter in the job card which initiates the various types of regions. If the TIME= parameter is not used and job step timing has been selected at Operating System/360 system definition time, the default time which appears in the reader procedure used to read the IMS/360 JCL will apply, and IMS/360 in any of its various type regions will be terminated abnormally by Operating System/360.

It appears that the starting of region Types 0 and 1 will normally be done through a procedure. In both MFT and MVT, reader procedures can be written which in turn invoke full job procedures from a user procedure library or SYS1.PROCLIB. However, in the case of Type 2 and Type 3 regions, the JCL that initiates these regions will normally be read in through the user's input stream; the responsibility, therefore, for placing a time parameter in the job card becomes the user's.

Another consideration is the dispatching priority of the running IMS/360 region type. In an MVT environment, the priority scheduler selects from a given class input queue the top priority job for initiation and creates a task control block which functions within the system at some relative dispatching priority related to the job selection priority. In MFT, however, dispatching priority is controlled by the partition number into which a job is scheduled. The normal MFT algorithm is that partition 0 has the highest priority, and partition n has the lowest priority. An IMS/360 Type 1 region operating in an MVT environment will CHAP itself below the dispatching level of the Type 0 region if possible. However, in an MFT environment, this is not possible. The IMS/360 Type 1 region will terminate with an abnormal completion code. So, in an MFT environment, the IMS/360 region Types 0 and 1 should have the same relative partition relationship; that is, IMS/360 should be in a partition number lower than Type 1 and 2 regions.

Another difference in IMS/360 operating characteristics under MFT and MVT is abnormal termination of Type 1 (message processing) and Type 2 (batch against online data bases) processing regions. Under MVT, the application program in either of these region types is ATTACHed as an Operating System/360 subtask. If the subtask abnormally terminates, the higher level controlling task communicates to the IMS/360 control program that abnormal termination has occurred. The resources used by the application program in the IMS/360 control program are released (that is, data base buffers, message type, the program, and the data bases). In the Type 1 region (message processing), the Operating System/360 job is not terminated, and scheduling of another message processing program occurs. Using MFT, the IMS/360 control modules and the application program in a Type 1 or Type 2 processing partition are part of the same Operating System/360 task. If the application program abnormally terminates, the entire Operating System/360 job terminates. The resources used by the Type 1 processing partition within the IMS/360 control program are not released until another message partition control module is started in the same partition. Through the use of MFT class initiations, the user should immediately initiate another message partition control module in the same partition. A good practice may be to "stack" another copy of the message partition control JCL in the input queue for that job class. The loss of the message partition will cause the next copy to be initiated in the same partition. The first call to the IMS/360 control program by the new message partition job causes the IMS/360 control program to release resources used by the ABENDed program.

ESTIMATING STORAGE AND MACHINE REQUIREMENTS

One of the things that must be planned for is the storage requirements of an Operating System/360-IMS/360 installation. Although the basic formula for estimating resident storage requirements of IMS/360 is contained in the _IMS/360 Application Description Manual_ (GH20-0524), further detail and examples are contained in this section. The techniques provided in this section allow the system user to obtain a more accurate storage estimate than that in the _IMS/360 Application Description Manual_. Do not attempt to mix partial estimates obtained from the following techniques and those obtained from the Application Description Manual.

Estimating IMS/360 Type 0 Region Main Storage Requirements

The main storage requirements for the Type 0 region are influenced by the number of data bases defined, the number of telecommunication lines and line groups, the type of telecommunication terminals to be supported, the buffer requirements to support defined data bases, and MAXIO specification in the IMSCTRL macro statement of system definition, performance requirements, etc. IMS/360 system definition supplies a procedure, IMS0, for the IMS/360 Type 0 region. The IMS0 procedure supplies default parameters in the EXEC card for REGION= and PARM= operands. These operands determine region size and region storage usage. The IMS0 procedure also provides definition of the positional characters in the PARM= operand. The following data should assist the user in determining an initial region or partition size. After proper analysis, the user may wish to change the default values in the IMS0 procedure. Chapter 4 of this manual provides an illustration of IMS0 procedures.

The storage requirements estimates are discussed under the following headings:

- Basic Storage Requirements
- BTAM Device Support

- IMS/360 Control Blocks
- IMS/360 Buffer Pools

Basic Storage Requirements

IMS resident nucleus (less control blocks)	60,000 bytes
Data Language/I action modules	12,800
OSAM access method modules	2,900
ISAM and IMS ISAM simulator modules	7,800
BTAM (less device support modules)	6,600
	90,100 bytes

BTAM Device Support

Add to the basic storage requirements, BTAM device modules for each line group described at IMS/360 system definition.

1050 nonswitched without autopoll	224 bytes
1050 nonswitched with autopoll	234
1050 switched	328
2740 with dial, transmit control, and checking	304
2740 with station control and checking	240
2740 with station control, checking, and autopoll	224
2260 remote	272

IMS/360 Control Blocks (Except Security)

Add to the basic storage requirements the size of the control block load module from the linkage editor output of Step 36 of Stage 2 of IMS/360 system definition. If the system user has not performed an IMS/360 system definition, the following should allow a control block estimate. The exact size of the control blocks may be calculated by referring to Chapter 11 of the IMS/360 System Manual.

Basic block requirement	12,000 bytes
Per message or Type 2 batch program	48
Per online data base	30
Per communication line	112
Per communication terminal	30
Per logical terminal (at least one per communication terminal)	64
Per transaction code	52
Per message or Type 2 batch region	280

Security Control Blocks

If the extended password and terminal security facilities are used, there are additional main storage requirements beyond those required by the resident control blocks generated during system definition. Security matrix bounds and main storage requirements may be determined using the following formula:

(I/8) * R = M = <32767

where:

M is the total main storage requirement in bytes.

I is the number of securing resources (passwords or logical terminals).

R is the number of unique combinations of secured resources.

The storage requirements for the password table may be determined using the following formula:

$$P = L * N$$

where:

P is the total main storage requirement in bytes.

L is the length in bytes of the longest password. L can vary from one to eight characters inclusive.

N is the number of passwords = < 32768.

Examples of security control blocks:

1. Terminal security

   Assume 200 logical terminals as securing resources, and

   100 unique combinations of secured resources (transactions, command verbs, etc.)

   $(200/8) * 100 = 2500 < 32768$

   Terminal security storage = 2500 bytes

2. Password security

   Assume 400 passwords as securing resources (maximum password length is four characters), and

   200 unique combinations of secured resources (transactions, command verbs, etc.)

   $(400/8) * 100 = 5000 < 32768$

Password table 400 * 4	1600 bytes
Password matrix	5000
Password security storage	6600 bytes

IMS/360 Buffer Pools

IMS/360 requires a set of buffer pools for communication line buffers, message queue buffers, data base control block buffers, and data base buffers. The storage requirement for these pools must be added to the basic storage requirement for the IMS/360 region size.

The following pools are required:

- Queue control record (QCR) and message buffer pool
- Data base buffer, output message buffer, and 2260 buffer pool
- PSB control block pool
- DMB control block pool

QCR and Message Buffer Pool. One queue control record buffer is required for each input 2740 and one for each 1050 telecommunication line to be serviced. Step 4 of Stage 2 of IMS/360 system definition adds the procedure to the user-defined procedure library named IMS0. The calculated number of QCR buffers required is defined in the DDD positions of the PARM= operand on the EXEC card. In that procedure, a number of message buffers are defined in the EEE positions of the PARM= operand for the system defined by the user. Message buffers are utilized for handling multiline messages.

The main storage requirements for these buffers may be determined by using the following sizes per buffer:

Queue control record buffer 176 bytes

One QCR buffer is required for each input communication line and each message region.

Message buffer 880 bytes

One message buffer is required for each simultaneous input or output request for a multisegment message.

Data Base Buffer Pool. When a program is scheduled into a message region or when a Type 2 batch program is scheduled, the buffers required for the data bases used by that program are obtained from a general data base buffer pool. When the application program terminates, the assigned buffers are returned to the pool. The output of each DBD generation Step 2 (linkage editor) includes a SETSSI value produced in Step 1. This value is the data base buffer byte requirement in hexadecimal required for use of the data base by an application program in an IMS/360 Type 1 or 2 region. If data base logging is specified for any data base referenced by a message or batch Type 2 program, the size of a message buffer (80 bytes) plus 72 bytes (152 bytes total) must be added to the total buffer requirements for that program. The pool of data base buffers provided for use in the IMS/360 Type 0 region must be at least as large as that required by the message program or batch program in a Type 2 region which uses the largest number of data base buffer bytes. As an example, assume that program X, the largest user of data base buffers, uses three data bases which require 5000 bytes of buffer storage each. The absolute minimum requirement for the data base buffer pool will be 15,000 bytes. If more than one Type 1 or 2 region is to execute concurrently, the data base buffer pool must be large enough to obtain the buffers for the data bases to be used in each message and Type 2 region concurrently. If the data base buffer requirement associated with an application program cannot be satisfied at the time the program is to be scheduled for execution, IMS/360 will wait until buffer space becomes available. The IMS0 procedure provides a default size for the data base buffer pool in the HHH positions of the PARM= operand of the EXEC card.

In addition to data base buffers, space in this pool is used to supply buffers during a /DBDUMP execution. These additional buffers are used for the HSAM data base image of the HISAM data base to be dumped. The total buffer space requirement is equal to twice the buffer requirement for the largest data base to be dumped.

2740/1050 Output Line Buffer Pool. Output messages to be transmitted via communication lines are read from queue control record and message buffers, edited to include line control characters, translated, and placed in output line buffers. Output line buffers are contained in the output line buffer pool. The size of this pool is defined by the HHH positions of the PARM= operand of the EXEC card in the IMS0 procedure. The value specified for the HHH positions of the PARM= operand should include the output line buffer pool requirements *in addition to* the data base buffer pool requirements. A reasonable pool size requirement might be 200 bytes per buffer and one buffer per three output 2740/1050 communication lines. However, if response is inadequate, the number of buffers should be increased.

2260 Line Buffer Pool. Buffers required for communication line control operations with a 2260 terminal are obtained from the 2260 line buffer pool. The size of this pool is defined by the HHH positions of the PARM= operand of the EXEC card in the IMS0 procedure. The value specified for the HHH positions of the PARM= operand should include the 2260 line buffer pool requirements in addition to the 1050/2740 output line buffer and data base buffer pools. One buffer of 1000 bytes should be included for each active 2260 line.

PSB and DMB Pool Sizes. A control block called a PSB exists on the PSB library for each message or batch Type 2 program. This block is loaded into core storage when the program is scheduled, and is retained as long as possible. The PSB is maintained in the PSB pool. A control block called the DBD also exists on the DBD library for each data base. This block is loaded into core storage and modified to create a DMB when the data base is initially used. The DMB is retained as long as possible in the DMB pool. The DMB contains DCB's for the data base in an OPEN state. The ability of IMS/360 to retain PSB's and DMB's in core depends upon their respective pool sizes.

The sizes of the PSB and DMB pools are determined by the values specified in the FFF and GGG positions of the IMS0 procedure PARM= operand. A default value is assigned by IMS/360 system definition to the FFF and GGG positions of the IMS0 procedure PARM= operand. These values must be at least large enough to contain the largest PSB as expanded during loading and all DMB's required by the application program that uses the largest number of data bases.

The size of a typical PSB is 500 to 1000 bytes. The size of a typical DMB is 600 bytes per data set group in the data base. The more PSB's and DMB's that remain resident, the greater the performance of the IMS/360 system. The maintenance of PSB's and DMB's in their respective pools is based upon frequency of use. The more often a program is scheduled and its data bases accessed, the greater the probability of retention of the block in core storage. The user can retain all PSB's and DMB's in core if the pools are defined large enough.

When a PSB is removed from the PSB pool to accommodate a new PSB, all DMB's used by that PSB and not concurrently used by other PSB's at time of PSB removal are removed from the DMB pool. This means the closing of the associated data sets. The user may avoid the removal of HISAM DMB's through the use of the second position in the PARM= operand on the EXEC card of the IMS0 or IMS1 procedure. However, the user must stipulate a DMB pool large enough to accommodate all HISAM DMB's used for online processing and his largest HSAM DMB or the HSAM DMB used in data base dump for the largest HISAM data base.

PSB Size Calculation. The exact main storage occupancy for a PSB may be calculated by adding the size of the PSB prefix (PSBPFX), the size of a teleprocessing program communication block (TPCBSZ), and the size of each data base program communication block (DPCBSZ). The formula for the calculation is:

```
PSBSIZE=PSBPFX + (TPCBSZ*n) + DPCBSZ,...+DPCBSZn
```

where:

PSBPFX = 44 bytes

TPCBSZ = 96 bytes if the PSB generation PSBGEN statement contains the keyword LANG = PL/I

= 40 bytes, if keyword LANG=COBOL or ASSEM

DPCBSZ = 112+(132*DSGn)+(44*SSEGn) + (LEVs*LEVn)+MKFBs + (12*FLDn)

where:

DSGn = number of data set groups defined; varies from 1 through 9

SSEGn = number of sensitive segment names defined; varies from 1 through 255

LEVs = 16 bytes plus the length of largest key field as defined by a FLDK statement in the associated data base definition, rounded up to the next fullword

LEVn = number of levels in the data base hierarchy; varies from 1 through 15

MKFBs = length of maximum concatenated key as specified in the KEYLEN = keyword of the PCB statement

FLDn = number of FLD statements in the associated data base definition

Note: If the DBDUMP command is to be used, the PSB pool must be large enough to contain a PSB that is twice the size of the largest possible data base PCB plus the PSBPFX size.

DMB Size Calculation. The exact storage occupancy for a DMB may be calculated by adding the size of the DMB prefix to the size of the DCB's required for each data set group (DSG) that comprises the data base and the number of segment types times a constant. The formula for size calculation is:

DMBSIZE = [(IDCBSZ + ODCBSZ + DSGPFX) * DSGN] + DMBNSZ + (NSEGT * 4)

which reduces to:

DMBSIZE = (488 * DSGN) + 8 + (NSEGT * 4)

where:

IDCBSZ = 252 bytes for an ISAM DCB
ODCBSZ = 228 bytes for an OSAM DCB
DSGPFX = eight bytes
DMBNSZ = eight bytes for DMB name
NSEGT = number of segment types, where 0<NSEGT< 256
DSGN = number of data set groups in the data base

Operating System/360 Requirements

Additional core storage is required in the IMS/360 Type 0 region for some Operating System/360 modules and control blocks. In addition, when MVT is used, core storage is required in the system queue space area.

| MFT:

Task I/O Table (TIOT) = 28 + 16n + 4d
n = number of DD cards
d = number of I/O devices

Program Request Blocks (PRB) (two required)	= 64 bytes
Supervisor Request Blocks (SVRB) (at least one required)	= 96 bytes
Loaded Program Request Blocks (LPRB) (38 required)	= 1520 bytes

Data Extent Blocks (DEB) - 160 bytes each

(one required per communication line group)

(five DEB's required for log and message queues)

(one DEB per ISAM data set in each data base)

Data Set Integrity (see MVT)

System Fetch Work Area	390 bytes
OPEN/CLOSE/EOV Work Area	1200 bytes
ABEND Work Area	1000 bytes
Input/Output Blocks (IOB)	136 bytes

(one IOB per communication line)

(two IOB's per data set)

MVT: The MVT requirements are divided between the space required in System Queue Space and the IMS/360 Type 0 region.

System Queue Space:

TIOT = (28 + 16n + 4d) bytes
n = number of DD cards
d = number of I/O devices

Main Storage Control Blocks and Misc.	500 bytes
PRB's (two required)	64 bytes
SVRB (one required)	96 bytes
LLE,XL, & CDE (40 required)	1760 bytes
DEB's - 160 bytes each	

(one per communication line group)

(one per ISAM data set)

(two per OSAM data set)

(five for log and message queues)

TCB's (three required)	504 bytes

Data Set Integrity Queue Blocks

sum of nd (14 + L) where nd is number of data sets allocated and L is number of bytes in the data set name including concatenation characters

Type 0 Region:

System Fetch Work Area	2000 bytes
OPEN/CLOSE/EOV Work Area	1200 bytes
ABEND Work Area	2000 bytes
IOB's - 136 bytes each	

(one per communication line)

(two per data set)

(ten for log and message queues)

Total IMS/360 Type 0 Storage Requirement

The sum of the basic requirement, device support, control blocks, message and queue control record buffers, data base buffers, 1050-2740 output message buffers, 2260 line control buffers, and PSB-DMB pool sizes represents a starting estimate of the region size value for IMS/360 execution.

IMS/360 Type 0 Region Estimates Example

Assumptions:

MFT

16 - 2740/1050 communication lines (16 terminals)
1 - 2260 line (four terminals)
5 - data bases concurrently open
10 - PSB's concurrently resident
18 - data sets allocated, all of which have eight-character data set names
2 - message regions
20 - message programs
40 - transaction codes
20 - logical terminals

IMS/360 Requirements

Basic Storage Requirement	90,100 bytes
2740 Station Control, Autopoll, Checking	224
2260	272
1050 - Nonswitched with Autopoll	192
IMS/360 Control Blocks (Step 36 system definition)	18,000

Pool Type	EXEC Card Parameter	Number of Buffers	Size of Buffers	Total
QCR Buffers	DDD=020	20	176	3,520
Message Buffers	EEE=005	5	880	4,400
PSB Pool	FFF=010	10	1000	10,000
DMB Pool	GGG=003	3	1000	3,000
General Pool*	HHH=020**			20,000

* The General Pool includes:

1. 1050/2740 Line Buffers	5	200
2. 2260 Line Buffers	1	1000
3. Data Base Buffers	18	1000

**The HHH EXEC card parameter specifies the size of the general pool in multiples of 1000 bytes.

Total IMS/360 Requirements 155,708

Operating System/360 Requirements

TIOT = 28 + 16n + 4d

17 - communication lines
4 - message queue data sets
2 - tape (log/data base recovery)
23

28 + (16 x 23) + (4 x 23) = 488 bytes

PRB's (2) 64

SVRB's (1) 96

LPRB's (38) 1520

DEB's
1 - 1050 line group
1 - 2740 line group
1 - 2260 line group
2 - tapes
4 - message queues
15 - data base, data sets

24 x 160 bytes 3840

Data Set Integrity

18 data sets
8 character names

(14 + 8) * 18= 396

Fetch Work Area 390

OPEN/CLOSE/EOV Work Area 1200

ABEND Work Area 1000

IOB's	1 - per communication line	=	17
	2 - per tape	=	4
	2 - per message queue	=	8
	4 - per data base	=	20
	TOTAL IOB's		49

49 x 136 bytes	6,664
Total Operating System/360 MFT Requirement	21,558
Total IMS/360 Requirement	155,708
Total Operating System/360 MFT Requirement	21,558
Total Partition Requirement	177,266

In addition to this total partition storage requirement, storage is necessary for message processing partitions, the basic MFT nucleus, system writer (optional), and basic RAM area.

Estimating IMS/360 Type 1 and Type 2 Region Main Storage Requirements

The size of an IMS/360 Type 1 or Type 2 processing region is determined primarily by the size of the system user's application programs. The only permanent requirement of IMS/360 is 2000 bytes in each message or Type 2 batch region. Prior to loading the user's application program, IMS/360 requires an additional 4000 bytes to initiate the processing region.

Estimating IMS/360 Type 3 Region Main Storage Requirements

The core storage requirement within an IMS/360 Type 3 region is primarily dependent upon:

- The size of the user's application program
- The number of data bases to be used and their buffer requirements
- The IMS/360-Data Language/I and Operating System/360 modules utilized

Basic Modules

The IMS/360 region control modules and the Data Language/I Type 3 region nucleus require about 10,000 bytes. For initialization, about 8000 additional bytes of work space are required prior to loading the user's application program. When the user's application program is to be loaded, these 8000 bytes of work space are available.

Control Blocks

Associated with each application program is a control block called a PSB, and associated with a data base is a control block called a DMB (data management block). The exact formula for calculating PSB storage requirements will be found in the section titled "Estimating IMS/360 Type 0 Region Main Storage Requirements". Assume 2000 bytes for a PSB in the following example. A DMB for a single data set group data base is 500 bytes; a DMB for a multiple data set group data base is 500 bytes per data set group. As an example, a program that uses two data bases, one of which has one data set group and one of which has three data set groups, would require the following control block space:

1 - PSB - 2000 bytes	2000 bytes
1 - Single data set group DMB	500 bytes
1 - Three data set groups DMB	1500 bytes
Total IMS/360 control block space	4000 bytes

Data Base Buffers

The core storage requirement for all data base buffers associated with all data bases used by an application program in an IMS/360 Type 3 region is obtained in that region. The creation of a data base description (DBD) associated with a data base is provided in a SETSSI statement. This SETSSI statement gives, in hexadecimal, the number of bytes required for data base buffers associated with the data base. The buffer requirements for a data base are equivalent to two times the ISAM data set block size for each ISAM data set plus four times the OSAM logical record length for each OSAM data set within the data base.

Data Base Organization Modules

In addition to the basic IMS/360 modules in a Type 3 region, various sets of data base organization modules are required. The need for these modules is dependent upon the type of data base organizations and processing options employed by the user's application program. The module sets and their respective core storage requirements are as follows:

Hierarchical Sequential data base with processing option of GET or LOAD	5,300 bytes
Hierarchical ISAM data base with processing option of LOAD	13,000 bytes
Hierarchical ISAM data base with processing option of GET	20,000 bytes
Hierarchical ISAM data base with processing option of ALL (GET, ISRT, DLET, REPL)	24,000 bytes

Regardless of the number of data bases used by an application program, the data base organization module sets are required only once in the IMS/360 Type 3 region for each data base organization.

Operating System/360 Control Block Requirements

As in the IMS/360 Type 0 region, it is necessary for core storage in a Type 3 region to contain some Operating System/360 control blocks. The following list indicates the types of control blocks, their storage requirements, and their need in the IMS/360 Type 3 region. Some of these control blocks are maintained in system queue space when operating with Operating System/360-MVT. The reader should refer to the section in this manual titled "Estimating IMS/360 Type 0 Region Main Storage Requirements" for obtaining Operating System/360 requirements in any Operating System/360 region.

Type 3 Region Storage Requirement Example

Assumptions for example:

1. User's application program requires 20,000 bytes.

2. Three data bases are utilized.

   1 -- Hierarchical Indexed Sequential data base - single data set group - processing option equals LOAD (L)

   2 -- Hierarchical Indexed Sequential data bases - two data set groups each - processing option equals ALL (A)

3. All ISAM block sizes equal 2000 bytes, and all OSAM block sizes equal 1000 bytes.

IMS/360 Storage Requirement

	Amount
Basic Modules	10,000 bytes
User's Application Program	20,000
Control Blocks	
PSB	3,000
3 - DMB's (Five data set groups)	2,500
HISAM (processing option equals L)	13,000
HISAM (processing option equals A)	24,000
Data Base Buffers (2 - ISAM, 4 - OSAM Block Size Buffers Per Data Set Group)	

2 buffers x 5 data set groups x 2000 bytes/buffer	20,000	
4 buffers x 5 data set groups x 1000 bytes/buffer	20,000	
Total IMS/360 Requirement	112,500	bytes
Operating System/360 Requirement		
TIOT = 28 + 16n + 4d		
Assume 5 I/O devices = d		
Assume 10 DD cards = n		
= 28 + 160 + 20	208	bytes
Program Request Blocks (2 required)	64	
Loaded Program Request Blocks (10 required)	400	
Supervisor Request Block (1 required)	96	
Data Extent Block (20 required)	3200	
System Fetch Work Area	390	
OPEN/CLOSE/EOV Work Area	1200	
ABEND Work Area	1000	
Input/Output Blocks (2 per data set) x 10 data sets	1360	
Total Operating System/360-MFT-II Requirement	7560	bytes
Total Type 3 Region Storage Requirement		
IMS Type 3 region requirement	112,500	bytes
Operating System/360 region requirement (MFT)	7,918	
Total Region Size	120,418	bytes

In addition to the IMS/360 Type 3 partition or region requirements, storage is necessary for the basic Operating System/360 nucleus, RAM or link pack area, and system writer (optional).

IMS/360 DATA BASE CONSIDERATIONS

Variable Length Data Base Record Processing

The data base creation and processing capabilities of Data Language/I allow an application to define a data base record structure in hierarchical segments. The actual number of segments within a particular data base record may and probably will vary significantly across all data base records within the data base. This creates the need for the ability to handle the physical storage of variable length application logical records or data base records. The degree of variable length capability must not be constrained by the physical attributes (track length) of an input/output device. An application logical record may be a partial track or may exceed a cylinder of direct access device space.

In order to provide this variable length physical storage capability, Data Language/I has adopted the following philosophy. All segments of a data base record may be stored in one physical record or in multiple physical records.

When multiple physical records are required, the first physical record points to the second by relative direct access device address, and the second to the third in a like manner. When the Hierarchical Indexed Sequential organization is used, the first physical record of any data base record is an ISAM logical record. Any subsequent physical records for the same data base record are OSAM physical records (Figure 6).

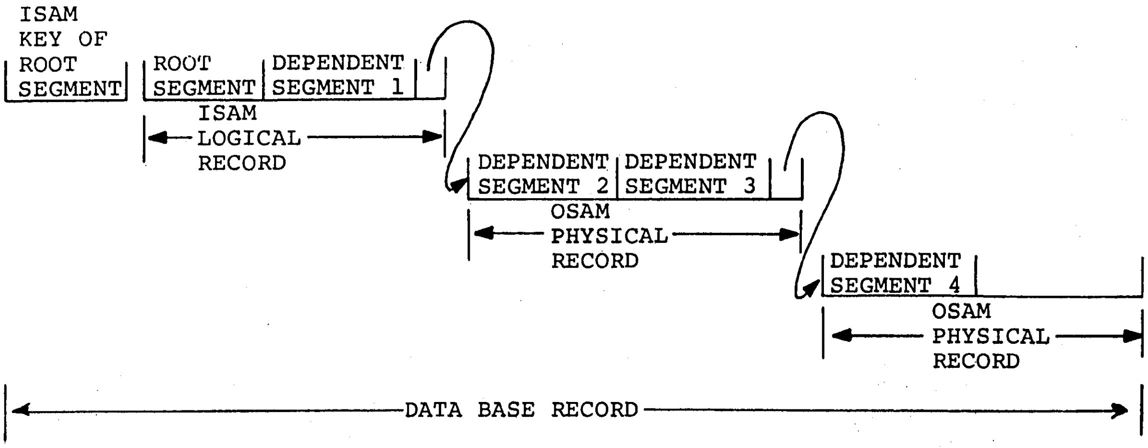


Figure 6. Multiple physical record example

This concept of variable length data base record support is provided through the use of OSAM. When an OSAM data set is opened in the Operating System/360 data management sense, it is used by Data Language/I for both reading and updating in-place existing segments of a data base record or for the addition of new segments of a data base record. An OSAM data set may have as many as 16 direct access device extents and may exist on up to five direct access device volumes. The physical records of an OSAM data set are the same length as the logical ISAM records of the same data set group within a data base. OSAM does not have a variable length physical record capability with a data set.

The OSAM capability has effectively extended the ability of Operating System/360 ISAM through Data Language/I to create, maintain, and process variable length application logical records.

The foregoing is summarized as follows:

Record	Data Set	Organization	Segment	Occurrence
1	Primary	ISAM	Root and Dependents	Initial Loading
2	Secondary	ISAM	2nd Level and Dependents	Initial Loading
3	Primary	OSAM	Root Segment Overflow	Insertion or Addition to Data Base
4	Secondary	OSAM	2nd Level Overflow Segments	Insertion or Addition to Data Base
5	Primary and Secondary	OSAM	All Dependent Segments	Both Initial Loading and Insertions

Data Language/I Record Format

For the Systems Operation function, it is felt that the actual Data Language/I record format is required for complete understanding of the organization of the data base. The explanation following assumes that this data base is of the Hierarchical Indexed Sequential organization and contains multiple data set groups. There are five logical record formats for a multiple data set group Hierarchical Indexed Sequential organization. The first three formats also pertain to the single data set group organization.

Figure 7 shows the relationship between the different types of logical record formats. The relationship differs according to the data set group, the ISAM or OSAM data set organization, and whether the segment is root or dependent. The type numbers in Figure 7 are explained in the next paragraphs.

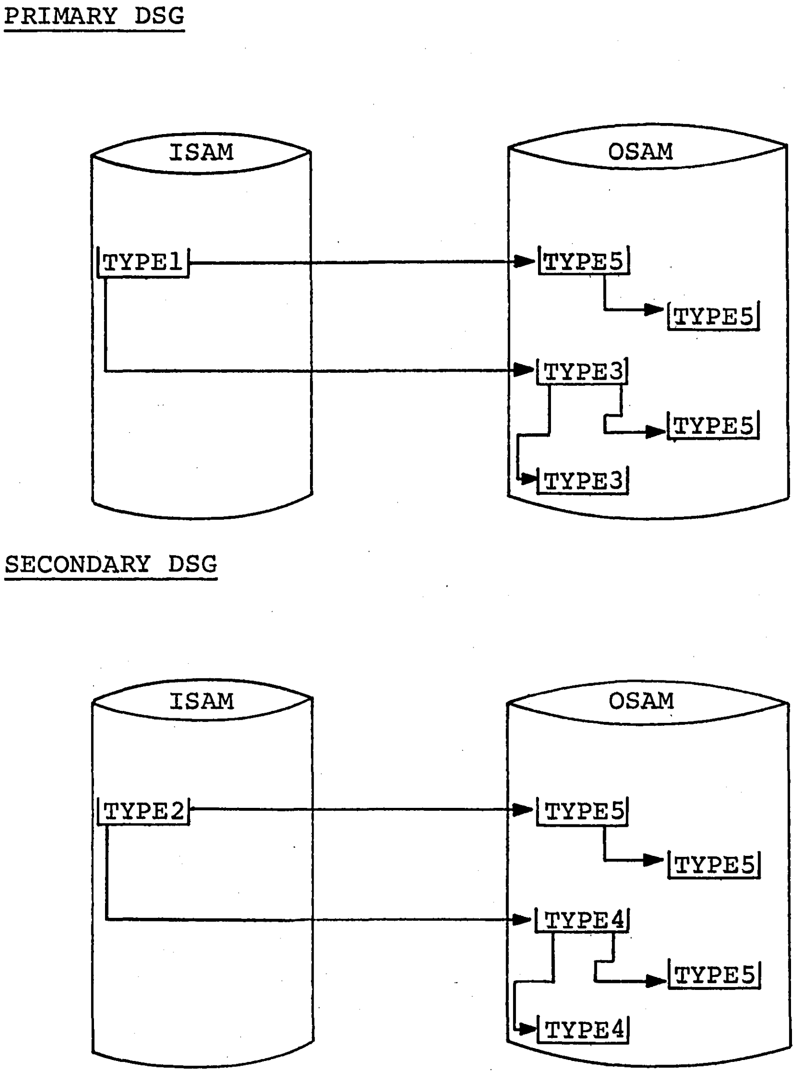
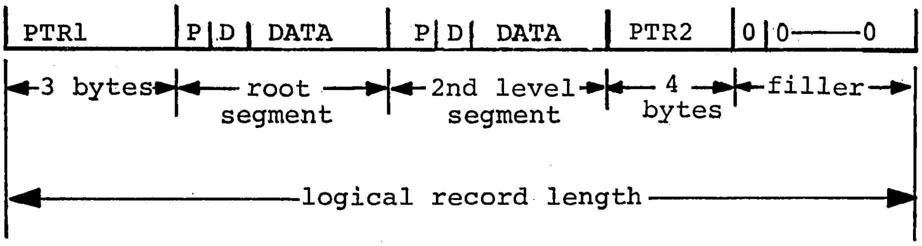


Figure 7. Logical record format relationship

Type 1 Record Format

The Type 1 logical record Data Language/I format is contained within the primary data set groups, is in the ISAM organization, pertains to root segments, and occurs primarily at initial loading of the data base. The format is:



where:

PTR1 = PTR pointer to next root segment (Type 3 format) in OSAM

P = one byte of overhead per segment for physical code of segment type

D = one byte of overhead per segment for delete code

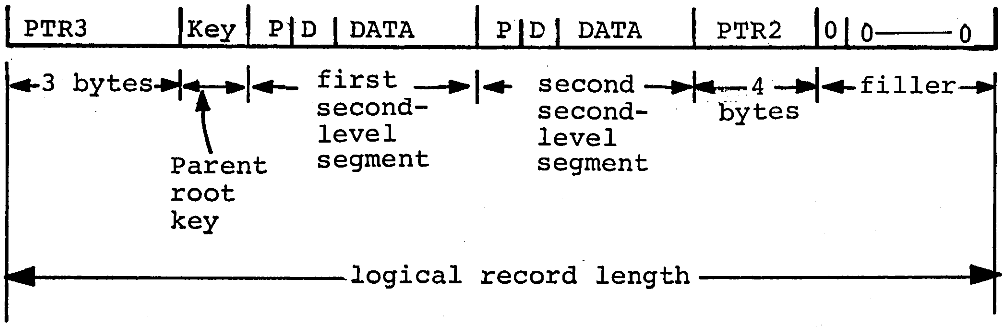
DATA = actual data for that segment, including its key

PTR2 = PTR pointer to next dependent segment (Type 5 format) in OSAM, if there is necessity for overflow. Otherwise, all four bytes are binary zeros.

filler = the remaining area not used by data. The PTR pointer is filled with binary zeros.

Type 2 Record Format

The Type 2 logical record Data Language/I format is contained within the secondary data set group, is in the ISAM organization, pertains to second-level segments (cannot start with root segments; must be no higher than second level), and occurs primarily at initial loading of the data base. The format is:



where:

PTR3 = PTR pointer to next second-level segment (Type 4 format) in OSAM

Key = The parent root key

P = one byte of overhead per segment for physical code of segment type

D = one byte of overhead per segment for delete code

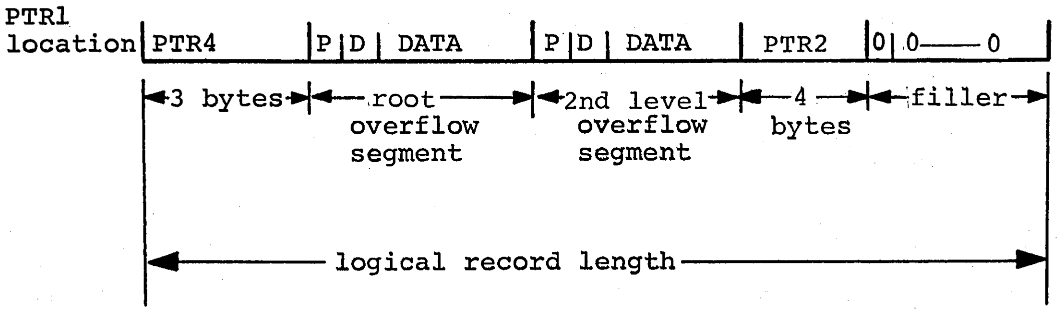
DATA = actual data for the segment, including its key

PTR2 = PTR pointer to next dependent segment (Type 5 format) in OSAM, if there is necessity for overflow. Otherwise, all four bytes are binary zeros.

filler = the remaining area not used by data. The PTR pointer is filled with binary zeros.

Type 3 Record Format

The Type 3 logical record Data Language/I format is contained within the primary data set group, is in the OSAM organization, pertains to root segment overflow, and occurs primarily at insertion or addition to the data base. The format is:



where:

PTR1 location = location of PTR1 pointer in Type 1 format

PTR4 = PTR pointer to next root segment in OSAM (Type 3 format)

P = one byte of overhead per segment for physical code of segment type

D = one byte of overhead per segment for delete code

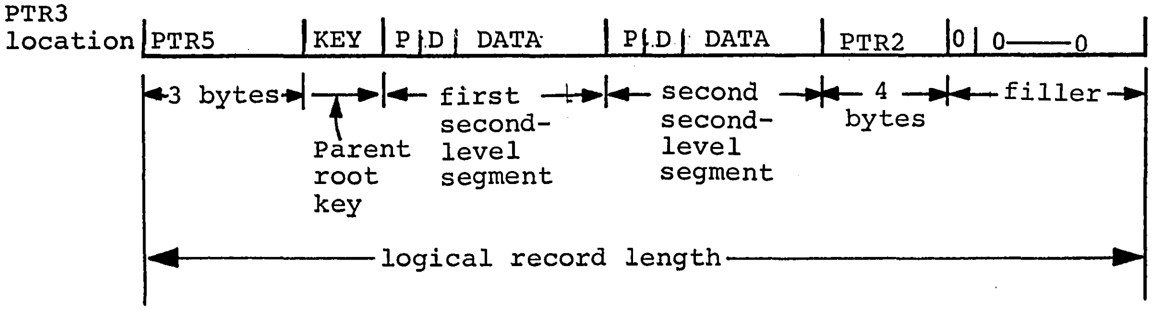
DATA = actual data for that segment, including its key

PTR2 = PTR pointer to next dependent segment (Type 5 format) in OSAM, if there is necessity for overflow. Otherwise, all four bytes are binary zeros.

filler = the remaining area not used by data. The PTR pointer is filled with binary zeros.

Type 4 Record Format

The Type 4 logical record Data Language/I format is contained within the secondary data set group, is in the OSAM organization, pertains to second level overflow segments, and occurs primarily at insertion or additions to data bases. The format is:



where:

PTR3 location = location of PTR3 pointer in Type 2 format location

PTR5 = PTR pointer to next second level overflow segment (Type 4 format)

KEY = The parent root key

P = one byte of overhead per segment for physical code of segment type

D = one byte of overhead per segment for delete code

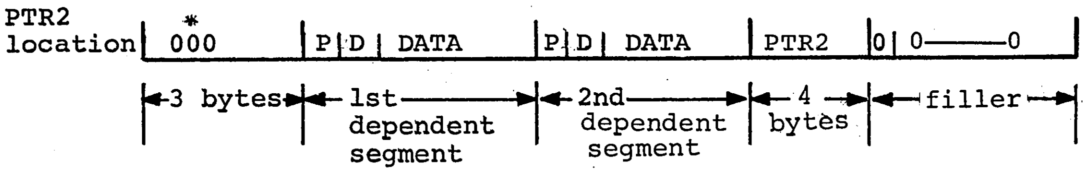
DATA = actual data for that segment, including its key

PTR2 = PTR pointer to next dependent segment (Type 5 format) in OSAM, if there is necessity for overflow. Otherwise, all four bytes are binary zeros.

filler = the remaining area not used by data. The PTR pointer is filled with binary zeros.

Type 5 Record Format

The Type 5 logical record Data Language/I format can be contained within the primary and secondary data set groups, is used as a part of the OSAM organization, pertains to all dependent segments, and can occur at both initial loading and insertions into a data base. The format is:



where:

PTR2
location = location of PTR2 pointer from either Type 1, 2, 3, 4, or 5 format

* = three bytes of binary zeros (reserved)

P = one byte of overhead per segment for physical code of segment type

D = one byte of overhead per segment for delete code

DATA = actual data for that segment, including its key

PTR2 = PTR pointer to another dependent segment record of the same type as this one

filler = The remaining area not used by data. The PTR pointer is filled with binary zeros.

Examples of Types of Data Language/I Logical Records

Figure 8 shows a data base structure that is used to explain, with examples, the different types of Data Language/I logical records. In Figure 8, the children (dependent segments) associated with root 1 are a number with the suffix 1. Such is the case also with roots 3 and 5. Root 2 and its children are noted in the same manner, but with dashes because they were not initially loaded into the data base.

The examples show these combinations:

Type 1 to Type 5
Type 2 to Type 5
Type 1 to Type 3
Type 2 to Type 4
Type 4 to Type 5
Type 5 to Type 5

Note that, for the purposes of example, a short logical record was constructed to force an overflow of OSAM.

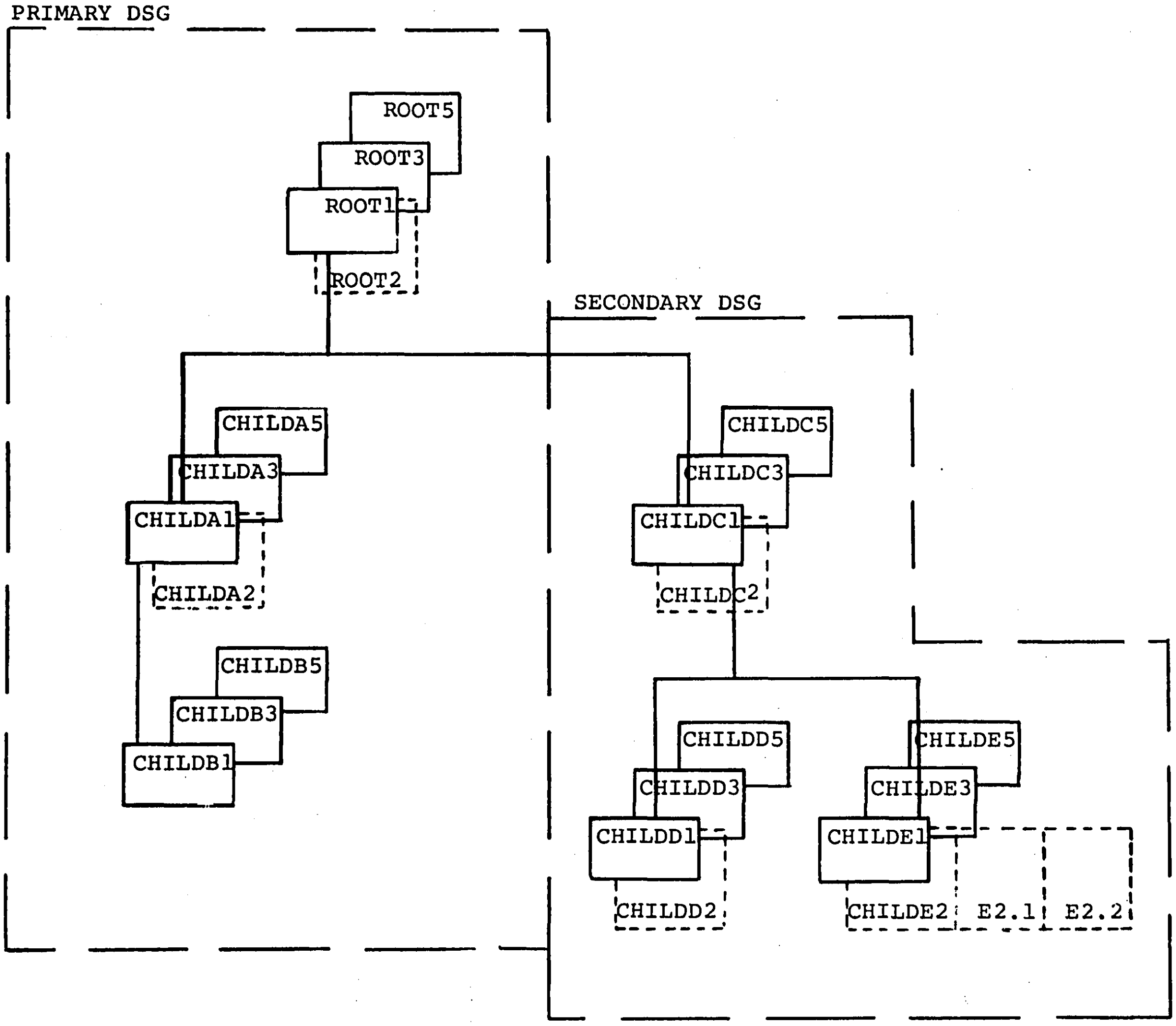
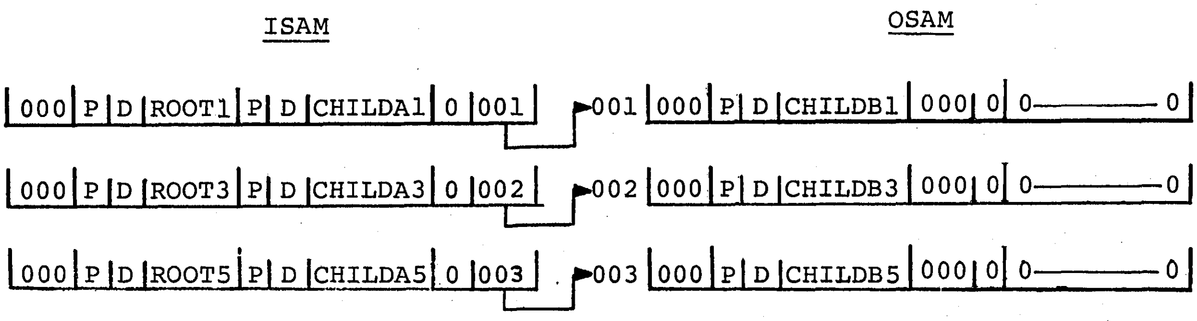


Figure 8. Types of Data Language/I logical records

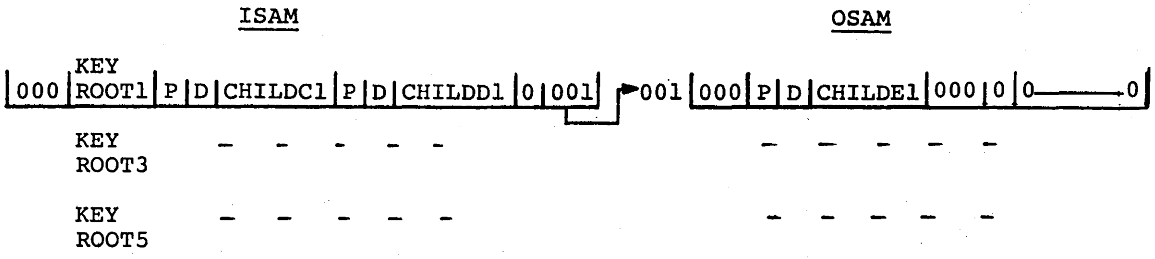
- **Example Type 1 to Type 5**

Primary DSG


ISAM
OSAM
000 | P | D | ROOT1 | P | D | CHILDA1 | 0 | 001
001 | 000 | P | D | CHILDB1 | 000 | 0 | 0 | 0
000 | P | D | ROOT3 | P | D | CHILDA3 | 0 | 002
002 | 000 | P | D | CHILDB3 | 000 | 0 | 0 | 0
000 | P | D | ROOT5 | P | D | CHILDA5 | 0 | 003
003 | 000 | P | D | CHILDB5 | 000 | 0 | 0 | 0

- **Example Type 2 to Type 5**

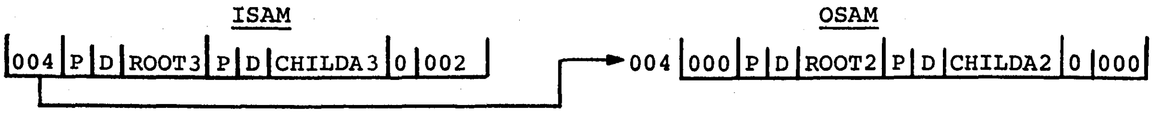
Secondary DSG


ISAM
OSAM
000 | KEY ROOT1 | P | D | CHILDC1 | P | D | CHILDD1 | 0 | 001
001 | 000 | P | D | CHILDE1 | 000 | 0 | 0 | 0
KEY ROOT3
KEY ROOT5

- **Example Type 1 to Type 3**

Adding root 2 to data base. See Figure 8.

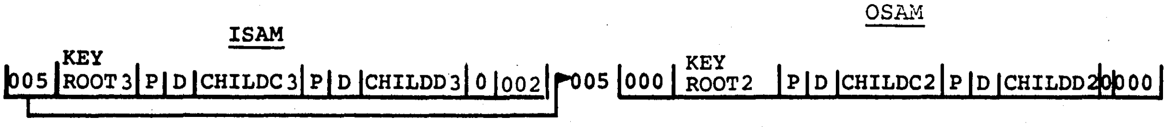
Primary DSG


ISAM
OSAM
004 | P | D | ROOT3 | P | D | CHILDA3 | 0 | 002
004 | 000 | P | D | ROOT2 | P | D | CHILDA2 | 0 | 000

- **Example Type 2 to Type 4**

Adding root 2's children to secondary data set group. See Figure 8.

Secondary DSG

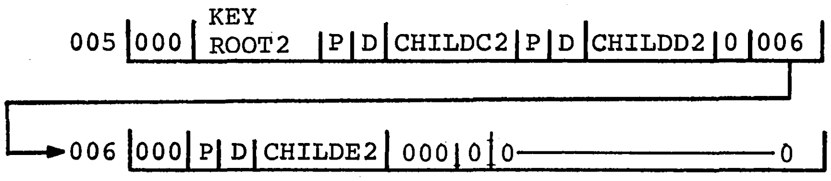

OSAM
ISAM
005 | KEY ROOT3 | P | D | CHILDC3 | P | D | CHILDD3 | 0 | 002
005 | 000 | KEY ROOT2 | P | D | CHILDC2 | P | D | CHILDD2 | 0 | 000

- Example Type 4 to Type 5

Secondary DSG

ISAM

OSAM

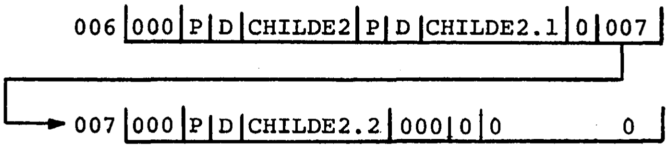


- Example Type 5 to Type 5

Secondary DSG

ISAM

OSAM



Data Base Creation

Initially, the Data Language/I data base must be loaded. Usually the data exists in a machine-readable form acceptable to COBOL, PL/I, or Assembler Language. If so, a user program (COBOL, PL/I, or Assembler Language) must read the data using conventional access methods and then issue Data Language/I insert calls to load the Data Language/I data base. Since the data was not previously organized in a hierarchical fashion, a certain amount of editing may be necessary before doing the Data Language/I load. Also, before the initial load, a DBD.GEN and PSB.GEN must have been done.

Data Base Creation DD Card Parameters

ISAM DCB Option Codes

The following option codes should be utilized and specified on the DD cards of the job for each data base creation.

```
DCB = (DSORG=IS,OPTCD=WM, [RECFM=FB])
```

where:

W = Write check

M = Master index creation (optional)

The user must not specify OPTCD=L, which indicates the presence of a delete byte in the ISAM logical record. The user should not specify OPTCD=I for ISAM independent overflow, because ISAM is not used to make additions to a data base.

The user may specify the RECFM, or it may be omitted. If RECFM is specified, it must state RECFM=FB.

OSAM DCB Option Codes

No DCB parameters need be specified on the DD card for the OSAM data set within a Data Language/I data base. However, the user may specify DCB =(DSORG=PS) if desired.

Single and Multiple Data Set Groups

Before creation of the data base, some consideration must be given to using single or multiple data set groups. (See "Definition of Multiple Data Set Groups" in Chapter 4 of IMS/360 Program Description Manual.) The DBD generation controls whether a data base is composed of single or multiple data set groups. The application program is insensitive to the number of data set groups; therefore, it is easy to experiment with different combinations until the optimum is found.

The advantages of a single data set group are:

- Only one ISAM index is needed; therefore, less storage space is used.

- On retrievals, using multiple data set groups, the ISAM index for the secondary data set group must be used to access segments in the secondary data set group. This may be more time-consuming than if a single data set group were used. Particularly on sequential retrieval of all dependent segments of a root, multiple data set groups would probably require more time than a single data set group.

- More core will be required for buffers using multiple data set groups.

The advantages of multiple data set groups are:

- The use of multiple data set groups is best indicated when a root has either so many or such long dependent segments that, for most roots, even with a large LRECL, all the dependent segments do not fit into the prime record.

- Under extreme situations, many OSAM blocks may be required to hold all the dependent segments for a single root. When this is the case, it is probably more efficient to use multiple data set groups, thus decreasing the references to OSAM.

- Using a single data set group, Data Language/I must go sequentially from the root through dependent segments to satisfy the call. It may be necessary, therefore, to pass over many dependent segments in order to satisfy a qualified Get Next or Get Unique type call for a second-level or lower segment. If all dependent segments of a root are contained in one block, this is a fast incore scan; but if the number of dependent segments requires multiple OSAM blocks, considerable time may be necessary to access these blocks sequentially in order to get the one containing the desired segment. If multiple data set groups are used, considerable scanning and possible OSAM access time can be saved because Data Language/I goes directly to the index of the data set group containing the requested segment. This consideration is Item 10a.(5) in the Systems Operation checklist.

Considerations of HISAM and HSAM

In deciding whether to use HISAM or HSAM, the HSAM restrictions must first be considered. Since HSAM is used to reference a sequential data set, data cannot be added, deleted, or replaced in an existing HSAM data set. Delete and replace calls are not valid for HSAM. Insert calls are invalid except when PCB processing options are equal to Load(L). Although HSAM is a sequential data set, it can be randomly processed within one volume. The data set will be scanned sequentially either forward or backward to satisfy the call. Therefore, to use an HSAM data set processed in a random fashion may be extremely inefficient. HSAM is not designed for random retrieval. Generally, when random processing is to be done, HISAM should be used. Exceptions to this may occur when the backward searches are very short or when all calls can be satisfied by proceeding forward through a data base.

Whereas a HISAM data set cannot be created online, an HSAM data set can. HSAM may be used to create an audit trail data base, in which case the time and date can be used as ascending order keys. This consideration is Item 10a.(6) in the Systems Operation checklist.

BISAM versus QISAM

The use of BISAM versus QISAM for access to Hierarchical Indexed Sequential data bases is determined by whether the input/output operation is executed from a Type 0 region or Type 3 region. BISAM is used for all ISAM reads and writes for update in a Type 0 region. It is more efficient than QISAM for direct record access and contains no embedded Operating System/360 wait instructions. This allows the IMS/360 control facility to control all Operating System/360 waits in the Type 0 region. QISAM is used in all Type 3 processing regions because it provides better sequential processing across one or more Data Language/I data base records accessed in a sequential manner. This consideration is Item 10a.(7) in the Systems Operation checklist.

Data Base Reorganization

Periodically, all data bases should be reorganized. This is necessary to delete those segments from the data base which have been marked "deleted", and to bring the added root segments which are placed in OSAM physical records during online processing back into ISAM prime records. It also decreases the amount of reprocessing necessary to recover from a hard error which requires a data base to be loaded and the update or additions to it to be reprocessed from the log tape.

The data base may be reorganized either by retrieving all segments from the HISAM data base and loading them to an HSAM data set, either on tape or direct access, or by retrieving them from the HISAM data base and loading them directly to a new HISAM data base. Going directly from HISAM to HISAM is the most efficient method, but this requires sufficient direct access data space to hold both copies of the data set. After the new one is loaded, the old HISAM packs can be held as the backup copy. Going to HSAM on tape as an intermediate step is not as efficient, but allows reloading of the HISAM data set on the same space previously allocated to it. It also may be convenient for any offline processing required and allows a tape rather than direct access space to hold the backup copy of the data base.

It should be noted that two DBD's are necessary if going directly from HISAM to HISAM. They will be similar but with different DBD names, and the DD1 and DLIOF entries on the DMAN card must be different. The DD1 and DLIOF entries specify the DD names on the DD cards. The DSNAMES may be the same on the old and the new data sets. The loading of the HISAM data base must be done in a Type 3 region.

Since the reorganization of a large data base requires a significant amount of time, a pertinent question is, "How can I tell when to reorganize my data base?" The answer depends on how volatile the data base is; that is, how many additions are made and how often these additions are being referenced.

The number of inserts is shown on the Application Accounting Report produced from the IMS/360 statistics programs. Another indicator for reorganization is the Transaction Response Report. When many of the segments being referenced are in OSAM, and/or the OSAM chains become long, the response times will become longer because of the increased amount of direct access arm movement required to respond to the call.

Care must be taken to ensure that there is always unused space available in the direct access space allocated to the OSAM data set. The IEHLIST utility program can be used to list the data set control blocks to monitor the amount of unused space available for OSAM additions. OSAM allows for a maximum of 16 extents across 5 volumes.

Description of Data Language/I Segment Insertion

Data Language/I segment insert logic for hierarchical indexed data bases is designed to handle (1) root segment insertion, and (2) dependent segment insertion. This section describes how Data Language/I implements segment insertion.

Root Segment Insertion

The logic for root segment insertion also includes the handling of second-level segment insertion on secondary data set groups. If the segment to be inserted is a root, Data Language/I proceeds to place the segment into OSAM and chain from ISAM of the primary data set group. For second-level segment insertion into secondary data set groups, the insert module calls upon the retrieve module for physical positioning within the proper data set and buffer as well as verification of the presence of a root segment. If a root exists for the second-level segment to be inserted, the retrieve module attempts to find an associated record on the secondary data set to prohibit duplicate dependent segments. When no record is found on a secondary data set, Data Language/I builds a new second-level segment and places the segment into OSAM with a chain from the ISAM data set of the secondary data set group.

The insertion operation in Data Language/I is performed by first searching the appropriate ISAM data set for a root segment key equal to or greater than the root segment or second-level segment key in a secondary data set group being inserted. If a segment is found with key equal, and the segment with a delete bit is turned on, Data Language/I inserts the new segment in place of the old segment. If a segment is found with key equal, and the segment with a delete bit is turned off, the insertion is rejected with the appropriate status code.

For a key-high condition, Data Language/I examines the first three bytes in the key-high record. This three-byte area signifies whether there is a chain of additional roots with keys less than the current key-high record. If the three bytes equal binary zero, Data Language/I inserts the next available OSAM block number into that three-byte area, writes the new root record out into OSAM, and, finally, writes back the ISAM key-high record, to include a pointer to the new OSAM block.

When the three-byte value is nonzero, Data Language/I reads the OSAM block that the three-byte value addresses and compares the newly read root key against the root key of the segment being inserted. If the newly read key is greater than the insert key, Data Language/I backs up to the previous record, moves its three-byte pointer into the new segment record, and writes out the new OSAM record. After completion of the successful OSAM write, Data Language/I updates the previous record by placing the newly written OSAM block number into the three-byte area and performing a write-back operation. If the key field of the segment in the OSAM block read is less than the key field of the segment to be inserted, the first three bytes of the OSAM block are tested equal or not equal to zero. If zero, the new segment to be inserted is written into the next available OSAM block, and the last-read OSAM block is updated to point to the new OSAM block. If nonzero, the next OSAM block chained to is read, and the key field of the segment in this block is tested against the key field of the segment inserted. At this point, the insert process will iterate through one of the above situations.

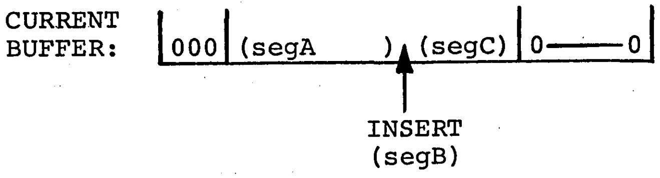
Dependent Segment Insertion

When a dependent segment is to be inserted into an existing data base record, the insert module of Data Language/I calls the retrieve module for the positioning action. After positioning action is complete, the insert module branches to one of four possible conditions, depending on how much slack space is available in the logical record.

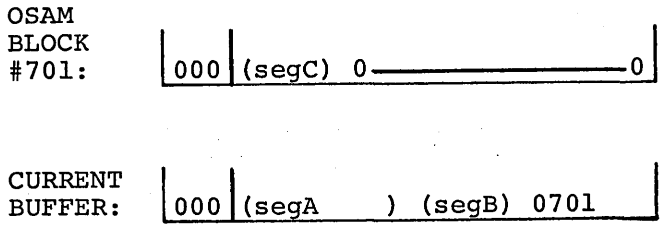
Condition 1. If there is enough space available to insert the new dependent segment into the existing physical record, this condition shifts any data that may exist to the right of the new dependent segment insert position. The new segment is then inserted and a "buffer pending" flag is turned on.

Condition 2. When the amount of slack is less than the insert segment length, a test is made to determine whether the new segment length plus an OSAM block pointer will fit into the area between the insert position and the end of the logical record. If so, any existing shift data is moved to a work area and is immediately written out to the OSAM data set. The new segment and just-written OSAM block number are moved into the current buffer, and the pending-flag is turned on.

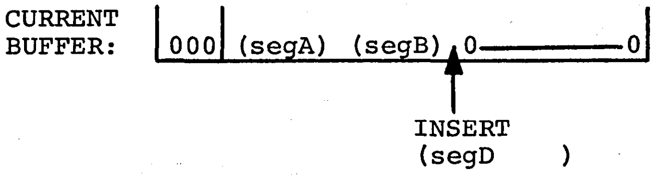
If the new segment plus the OSAM pointer do not fit, another test is required to determine whether enough space (four bytes) is available to hold an OSAM block pointer only. A YES is handled by Condition 3, and a NO by Condition 4.



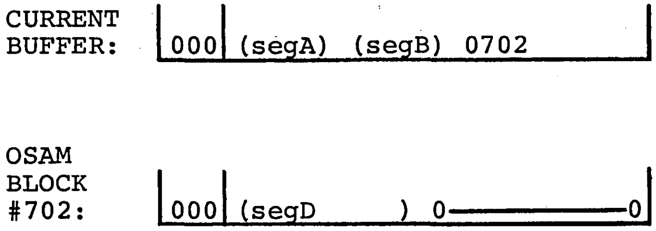
Current buffer before insert



Current buffer after insert

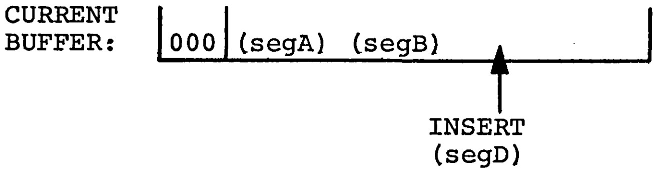


Current buffer before insert will not hold new segment, and slack is greater than 4 bytes

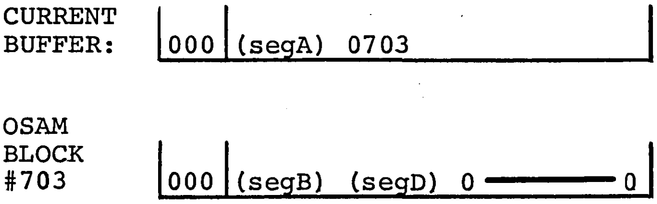


Current buffer after insert

Condition 3. This condition places any new segment and any existing shifted data into a work area and immediately writes out a new OSAM record. The just-written block number is moved into the current buffer, and the pending-flag is turned on.



Current buffer before insert has less than 4 bytes



Current buffer after insert

Condition 4. When less than four bytes of slack remain, the segment prior to insert position must be extracted to allow room for an OSAM block pointer. The previous segment, new segment, and any shift data are moved to a work area and are immediately written out to the OSAM data set. This newly written block number is then moved into the previous segment position, and the pending-flag is turned on.

Data Base Integrity Through the Use of OSAM

The modifications made to a Data Language/I data base by the Delete, Replace, and Insert functions create a need for internal capabilities in IMS/360 and Data Language/I to attempt to ensure the integrity of a data base. This is particularly true when operating in a message processing environment. The most complex Data Language/I input/output function of the three is Insert. Whenever a segment (root or dependent) is added to a Data Language/I data base of the Hierarchical Indexed Sequential organization, a new physical record may have to be generated. The OSAM data set(s) of that data base is used for all segment insertion. The Write-Key-New capability of ISAM is never utilized. The following diagrams illustrate a dependent segment insertion and are provided in the sequence of channel program operations. (See Figures 9, 10, and 11.)

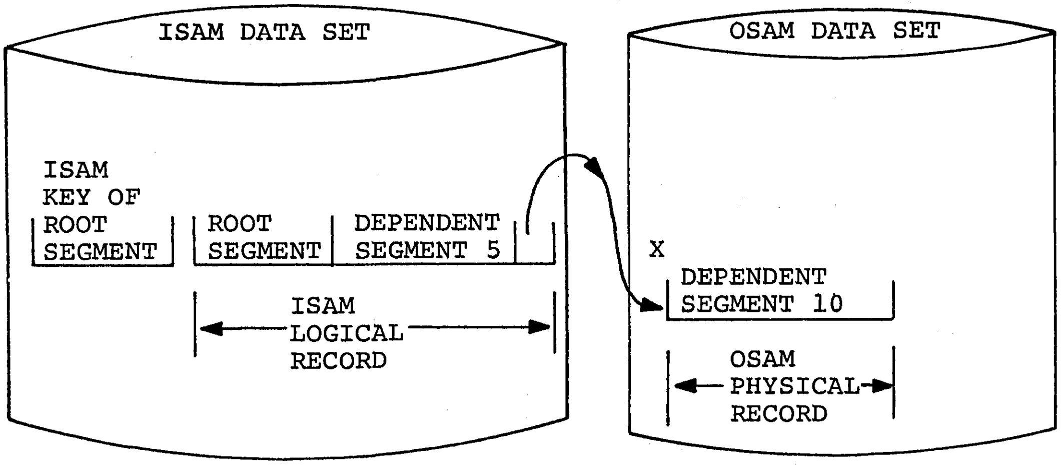


Figure 9. Data base prior to segment insertion

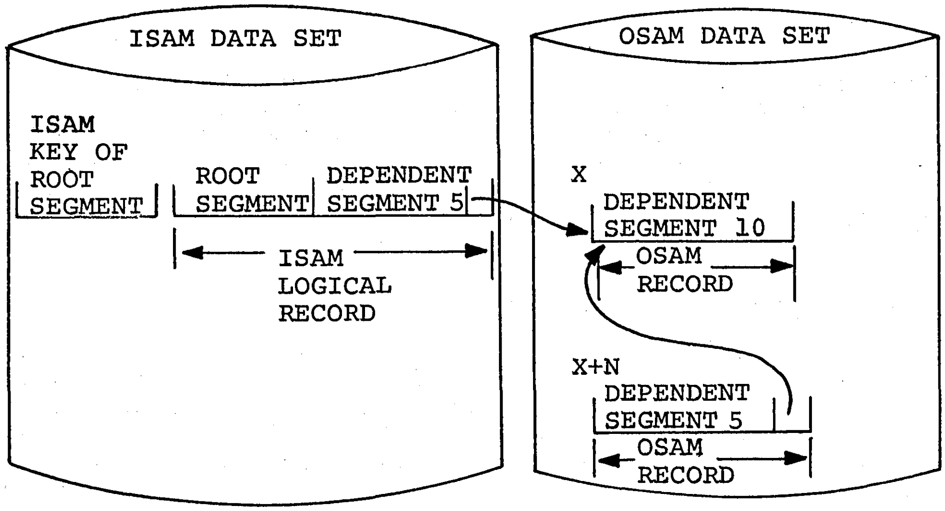


Figure 10. After channel program write and check of new OSAM physical record

The first OSAM physical record (prior to segment insertion) for the data base record is at direct access device address X. OSAM space within a data set is allocated sequentially. Assume that the next allocatable OSAM space is X+N. Assume that a dependent segment number 1 is to be inserted between the root segment and dependent segment number 5. There is no space available in the ISAM record to insert the additional segment, and the next available OSAM record address is X+N. A channel program reads the ISAM record, and Data Language/I finds that no space exists in the ISAM. Data Language/I also recognizes the existing direct access pointer to record X. Physical location X+N in the OSAM data set is allocated for the Insert. An OSAM record with dependent segment 5 is written at X+N and checked with a channel program (Figure 10).

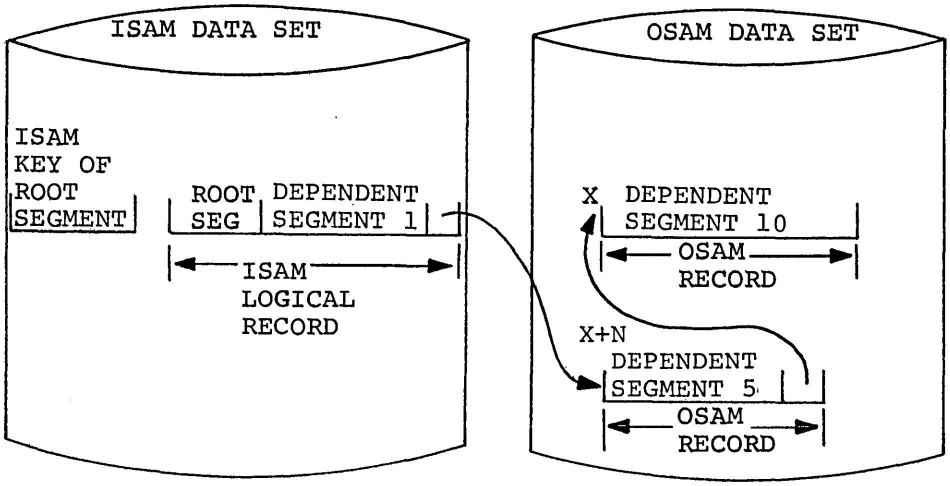


Figure 11. Data base after insertion

The OSAM record has a physical direct access address pointer to record X. Once the record at location X+N is successfully written, the ISAM record is updated, with the root segment and dependent segment 1 unchanged, using a channel program. The OSAM record pointer in the ISAM record is changed to X+N.

In order to maintain the integrity of the data base, the sequence of channel programs is important. The user should try an alternate sequence and consider the possibility of a system failure after start but prior to termination of the Insert operation.

Another critical consideration in the use of a sequentially allocated direct access space such as ISAM overflow areas, the sequential access methods, and OSAM, is the proper maintenance of the next allocatable direct access device address. All of the above access methods use a field in the data set control block (DSCB) to maintain this address while a data set is closed. When a data set is opened, this information is placed in a field in the data control block (DCB) in core storage. As records are added to the data set, the field in the DCB is appropriately updated. When the data set is closed, the DCB field is used to update the DSCB with the new allocatable address. If the system is lost after record adds, and the data set is not closed, the DSCB field is not updated. Positioning in the data set area is lost.

A special capability has been added to OSAM to alleviate this problem. Every time an OSAM record is added to a data set, a file mark (count field with key field and data field lengths zero) is written. When another new record is written, the file mark is overlaid and a new file mark is written after the added record. If the system is lost and the OSAM data set is not closed, the DSCB allocation pointer is not updated. If the data set is closed correctly, the DSCB is updated from the DCB. When the OSAM data set is subsequently opened, the next allocatable address from the DSCB is used for a record read. If a unit exception indication is received, the positioning is at a file mark, and the data set is assumed to have been previously closed correctly. If a unit exception indication is not received, the OSAM open routine sequentially reads records until a unit exception is received. This address then represents the proper positioning in the data set.

Data Language/I Data Base Space Allocation

When direct access storage is required for a data base, the amount of space needed and the device type must be specified. IMS/360 follows the same approach as Operating System/360. Refer to the manual OS/360 Supervisor and Data Management Services (GC28-6646), Part 3, "Data Set Disposition and Space Allocation".

The amount of space required can be specified in terms of blocks, tracks, or cylinders. If it is desired to maintain device-independence across direct access device types, space requirements should be specified in terms of blocks. Otherwise, if the request is in terms of tracks or cylinders, such items as their capacity must be considered. ISAM data sets must be allocated by cylinder. Table 12 of the OS/360 Supervisor and Data Management Services manual lists the physical characteristics of a number of direct access storage devices. The amount of space is supplied in the data definition (DD) statement for the data set.

Allocating the space for an IMS/360 Data Language/I data base that uses ISAM and OSAM is similar to allocating space for an Operating System/360 indexed sequential data set; similar because an Operating System/360 data set can be divided into three areas, prime, index, and overflow. The three areas of a Data Language/I data base are index, prime, and OSAM overflow.

Normally, DBD generation computes from the user's definition of segment frequency the logical record length (LRECL) of a data base. It considers the device and rounds to the next higher 1/4 track, 1/3 track, or 1/2 track. The computed LRECL will not exceed 1/2 track for any device.

For the Systems Operation function, IMS/360 has two additional parameters that can be inserted when it executes the DBD generation. These provide an additional means of specifying the LRECL and blocking factor (BLKFACT) for a data set within a data base.

In the DMAN control card, the additional parameters are LRECL and BLKFACT. Instead of DBD generation specifying the LRECL, it can be overridden by specifying the LRECL and the BLKFACT.

Allocation Problem Example

With the reader's knowledge of the data base structure, the application programs that will access that structure, and the tools of IMS/360 DBD generation, the following space allocation example will be meaningful. The example deals with an IBM 2314 Direct Access Storage Facility.

When an IMS/360 Hierarchical Indexed Sequential organization (HISAM) data base is loaded on an IBM 2314 Direct Access Storage Facility, it is necessary to allocate space for that data base with JCL data definition statements. The creation of a Data Language/I single data set group data base may require up to three DD statements, one each for the index, prime, and OSAM overflow areas. This example should provide assistance in initially determining the amount of space to allocate to these areas for any specific application. Each data set group within a Data Language/I multiple data set group can be treated as a Data Language/I single data set group data base when determining space requirements. The output of DBD generation also supplies minimum primary ISAM and OSAM allocation requirements.

ISAM Prime Area

This area contains the majority of the data records and all related track indexes. For this example, each root segment and all its dependent segments comprise a single logical data base record. For a specific data base, a logical record could vary from only the root segment to the root segment with a maximum number of segment occurrences for each dependent segment type. The distribution of the lengths of logical records of this example's data bases is plotted in Figure 12.

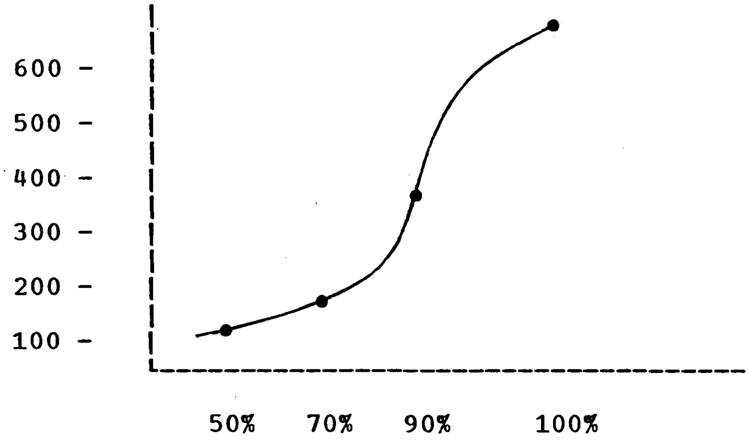


Figure 12. Logical record length distribution

The graph of Figure 12 indicates that 50% of the logical records are 150 bytes or less in length, 70% of the logical records are 200 bytes or less in length, 90% of the logical records are 500 bytes or less in length, and 100% of the logical records are 600 bytes or less in length.

With fixed-length ISAM, it is necessary to establish a fixed value for the logical record length (LRECL) in the prime area. If a value of 600 bytes is selected for the LRECL, all logical records can fit in the prime area. However, 90% of the records then have at least 100 bytes of slack or wasted space; 70% of the logical records have at least 400 bytes of unused space.

In this example, if a value less than 600 bytes is selected for the size of the LRECL, the ISAM prime area is not capable of holding all the logical data base records. Those dependent segment occurrences which do not fit in an ISAM prime LRECL are housed in OSAM overflow records. Therefore, the determination of an ISAM prime LRECL must consider the tradeoff between storage in the ISAM prime area and in the OSAM overflow area.

To determine a best balance between ISAM prime and OSAM overflow, the following points must be considered:

1. Access to data that is wholly contained in the prime area is more rapid than access to data contained in two areas. Access is even slower to those logical data base records that require more than one overflow record.

2. Disk space allocated to OSAM overflow can be used to hold segment occurrences that overflow from any logical data base record. Unused space in the prime area is tied to specific roots.

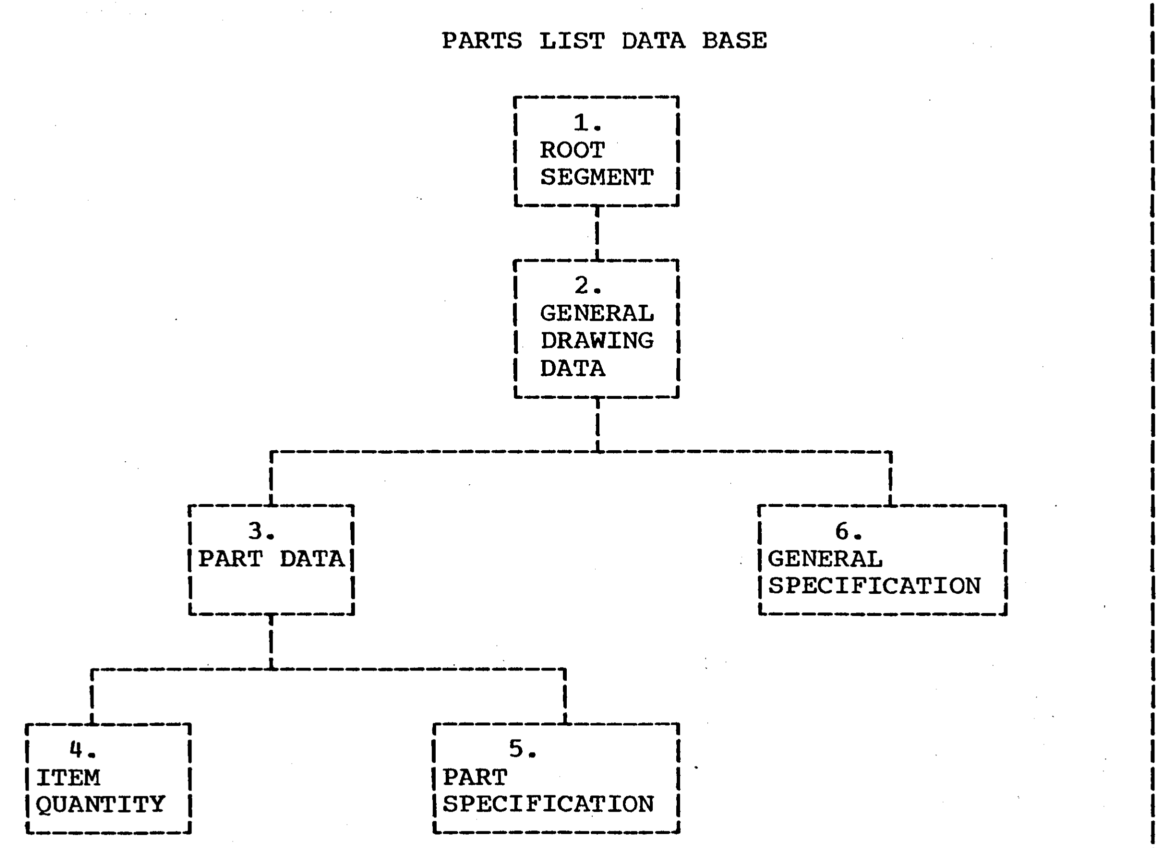
3. Records are blocked in the prime area but unblocked in the OSAM overflow area. This difference is negligible for large data base records but can be significant for small records. For example, with an LRECL of 27 bytes, 4864 records can be stored in one

cylinder in the prime area with half-track blocking. Only 840 similar records can be stored in one cylinder of overflow.

4. The nature of the accesses to the large logical data base records also has an important effect. If the large logical data base records are highly used, the size of the prime LRECL should be increased to completely house more logical records, and the total size of OSAM overflow should be reduced. If the large logical data base records are infrequently accessed, an opposite shift should be made to increase the use of OSAM overflow.

Considering these relevant factors for a specific data base, a percentage balance must be established between the ISAM prime and the OSAM overflow. For example, it may be best, in the context of optimizing space and time utilization, that 90% of the logical data base records completely fit in the prime area and 10% require some OSAM overflow storage. After this percentage is selected, the frequency of dependent segment occurrences is developed for the 90th percentile of the parent segments. The 90% is an estimated value for this specific data base. This is similar to the DBD generation requirements, except that the frequency will apply to 90% or more of the parents rather than to all of them. Those dependent segments that occur with certainty, that is, with fixed frequency for 100% of that segment's parents, will be specified at their full value.

Taking the segments contained in the example's data base (Parts List Data Base), Figure 13 shows the hierarchical structure relationship of the segments, a table of parent estimated frequency, and segment length. Note that the segment length in this table contains two overhead bytes needed for Data Language/I.



Data Base	Name of Segment	Parent	Estimated No. per Parent	Segment Length (plus 2 bytes control)
Parts List Data	1. Root Segment	--	--	20
	2. General Drawing Data	1.	1	8
	3. Part Data	2.	10(90% have 10 or less)	15
	4. Item Quantity	3.	2	5
	5. Part Specification	3.	1	10
	6. General	2.	4(90% have 4 or less)	25

Figure 13. Hierarchical segment structure and table of parts list data base

The calculations of the prime LRECL are:

Prime LRECL=20 + 1 {8 + 10 [15 + 2(5) + 1 (10)] + 4[25]} + 7*

Prime LRECL = 488 bytes

* The addition of 7 represents overhead bytes per LRECL for Data Language/I.

This establishes the fixed record size (calculated LRECL) for the ISAM prime area needed so that 90% of the logical data base records can be completely housed in the prime area.

The next step is to determine the IMS/360 LRECL that considers the calculated LRECL, the track length, and the blocking factor. Assuming half-track blocking, there are 3476 bytes in one block on the 2314 storage facility. The value of calculated LRECL is divided into 3476 to determine the number of records in a block, and the remainder from this division is equally distributed among the records.

The results of this process can be tabulated in the convenient form shown in Figure 14.

Calculated LRECL Range	IMS/360 LRECL	LRECL's per Block
1739 - 3476	3476	1
1159 - 1738	1738	2
870 - 1158	1158	3
696 - 869	869	4
580 - 695	695	5
497 - 579	579	6
435 - 496	496	7
387 - 434	434	8
348 - 386	386	9
317 - 347	347	10
291 - 316	316	11
268 - 290	290	12
249 - 267	267	13
233 - 248	248	14
218 - 232	232	15
205 - 217	217	16

Figure 14. LRECL for half-track blocking on the 2314 direct access storage facility

A calculated LRECL of 488 falls in the tabulated range of 435 - 496, which results in an IMS/360 LRECL of 496 bytes, with seven records in a half-track block.

The next step is to calculate the total amount of space that is required for the ISAM prime area. An estimate must be made for the number of roots that exist in the data base. In the example under consideration, there are 50,000 roots, that is, 50,000 logical data base records.

Fourteen records are blocked on a single track, and there are 19 tracks on a cylinder, excluding track indexes. Therefore, 266 records fit in a cylinder, and 188 cylinders hold the prime area. Since there are 199 usable cylinders in one 2314 pack, the ISAM prime area requires about 95% of one pack.

Index Area

This area contains master and cylinder indexes associated with the data set. It exists for any ISAM data set that has a prime area occupying more than one cylinder. The user can place this area on 2314 or 2301/2303 drum.

OSAM Overflow

The OSAM overflow area holds those dependent segment occurrences of a logical data base record that do not fit on one LRECL in the ISAM prime area. One or more OSAM records may be required in addition to one ISAM record to hold one logical data base record. A physical break in a logical data base record must not divide a segment occurrence. The determination of the amount of space needed for the OSAM overflow area depends on the percentage figure used to develop the space for the prime area. In the Parts List example, 90% of the logical data base records are expected to completely fit into the prime area.

This percentage figure must also consider the fact that the IMS/360 LRECL is equal to or larger than the calculated LRECL. This may have the effect of increasing the percentage of logical records that completely fit in the prime area. Assuming that 90% of the logical data base records do completely fit in the prime area, the remaining 10% are of interest in calculating the OSAM requirements.

It is necessary to determine which dependent segment types overflow and how many of these fit into one OSAM record. It is assumed in this example that, in the Parts List Data Base, segments 3 and 6 occur approximately in the proportion of 2-1/2 to 1. Note that if segment 3 overflows, the children of segment 3, segments 4 and 5, overflow with it. This means that one OSAM record holds approximately 20 segment 3s and 8 segment 6s. The next step is to determine what part of the 10% of the logical data base records will overflow into only one OSAM record; then two OSAM records are considered, and so forth, until the entire 10% has been specified. For the example under consideration, the statistics are in Figure 15.

Number of OSAM Records Required per Logical Record	% of Data Base	Number in Data Base	Number of OSAM Records Needed
0	90	45,000	0
1	6	3,000	3,000
2	3	1,500	3,000
5	1	500	2,500
	100	50,000	8,500

Figure 15. Statistics on 10% of the overflow of the logical records in a data base

Since the overflow records are 496 bytes long and unblocked, 220 records can be stored in one cylinder. Therefore, 39 cylinders are needed for OSAM overflow.

Example Conclusion

Following the above process, which is an estimation process, not a precise algorithm with exact values, estimates have been developed for the 2314 space requirements for a sample problem. The space requirements for the Parts List Data Base are 188 cylinders for the ISAM prime area and 39 cylinders for OSAM overflow. These values would be used in data definition statements to allocate space for the Parts List Data Base.

It should be noted that the objective of this estimation process is to develop values for use in initially loading and processing an IMS/360 data base. File growth has not been included in this example, but it must be considered. The developed space requirements may be later refined by historical data on actual space usage. Even without historical data, however, the use of the estimation process described here should result in reasonably accurate initial space requirements for a hierarchical data base of any degree of complexity.

Program Specification Block Generation

The Program Specification Block (PSB) generation utility is an important part of the IMS/360 system. It is normally the responsibility of the Systems Operation function to perform this generation from the information supplied by the Application Programming function.

There is a close relationship between PSB generation and DBD generation. The data base name must be specified in the PCB control card in the PSB generation (see Item 9 in the Systems Operation checklist). The application program's hierarchical (sensitive) segments must be named in PSB generation and described in detail in DBD generation (see Items 11 and 11a of the Systems Operation checklist).

Note that historical data should be kept for reference along with cross-reference information to sensitive data between application programs using the same data base.

The only difference between a PSB generation for a program of only data base usage and one which also includes teleprocessing is that message PCB's may be included at the beginning of the PSB generation deck. A message PCB in the PSB is not required for message processing programs. The input message PCB is part of the resident IMS/360 control program in the IMS/360 Type 0 region. That is, an input message PCB is generated for each message region by system definition. It is modified at scheduling time to reflect the source terminal of the current input message. The inclusion of PCB's within a PSB for message processing enables the message program to output messages to destinations other than the source of the input message. Output destined for the source of the input message is via the same PCB used for acquisition of the input message.

This IMS/360 Systems Operation manual describes, in Chapter 4, a library for residence of PSB's and an Operating System/360 procedure for creating PSB's. The default name of the program specification block (PSB) library is IMS.PSBLIB. This library name is used in the procedure PSBGEN. PSBGEN is a two-step assemble and link-edit procedure to produce each PSB.

JCL for PSB Generation

Use the procedure PSBGEN when running the different PSB generations. The JCL cards are:

```
//PSBGEN    JOB MSGLEVEL=1
//          EXEC    PSBGEN,MBR=
//C.SYSIN   DD  *

        ( PCB    )
        ( SENSEG )      Control cards for PSB generation
        ( PSBGEN )
        ( END    )

/*
```

where keyword operand MBR= is the name of the PSB to be generated.

Data Base Description Generation

The Data Base Description (DBD) generation is normally the responsibility of the Systems Operation function. It is an important factor in building the Data Language/I control blocks used to describe in detail the structure and storage organization of every data base.

The details of DBD generation are contained in Chapter 7 of the IMS/360 Program Description Manual. However, two additional parameters are a part of DBD generation: the logical record length (LRECL=) and the blocking factor (BLKFACT=). These parameters are a part of the options of the DMAN control card. The use of the LRECL and BLKFACT is discussed in the Data Language/I data base space allocation section of this chapter. Both parameters must be specified if either is used. If neither is specified, DBD generation attempts to calculate an optimum in logical record length and block size for the data base.

	DMAN	DD1=,DEV1=,[DD2=],[DLIOF=], [LRECL=,BLKFACT=]

LRECL is a specified number that is less than the maximum length allowed for a particular direct access device track. If the optional parameters are not used, the calculated optimum LRECL will be 1/4, 1/3, or 1/2 track.

The BLKFACT is a number that specifies the number of LRECL's per physical block.

Each data base to be used under the IMS/360 definition must be defined by a DBD generation.

The maximum allowable data base buffer requirements for all data set groups of any one data base can be calculated with the following formula:

$$\sum_{i=1}^{10} (\mathrm{BLKSIZE}_i +35) + ((\mathrm{LRECL}_i +19) * 4) <65{,}536$$

when i = number of data set groups.

Although DBD generation may allow construction of a DBD with greater buffer storage, execution of IMS/360 with that DBD and data base is not possible.

The DBD for a data base contains within it the Operating System/360 Data Control Blocks (DCB's) required by Operating System/360 data management. For HISAM, there are four DCB's - QISAM load mode, QISAM scan modes, BISAM read/write update, and OSAM. For HSAM, there are two DCB's - BSAM read and BSAM write. The DCB operands completed by DBD generation for each DCB type are:

```
QISAM LOAD: DSORG=IS,MACRF=PM,RECFM=FB,OPTCD=W,LRECL=,
    BLKSIZE=,RKP= ,KEYLEN= ,DDNAME=

QISAM SCAN: DSORG=IS,DDNAME= , MACRF=(SK,GL,PU)

OSAM: DSORG=PS,MACRF=E,RECFM=F,BLKSIZE=,
    LRECL= DDNAME=
BISAM: DSORG=IS,MACRF=(RU,WU), DDNAME=

BSAM: DSORG=PS,MACRF=(RP,WP)RECFM=U,BLKSIZE=,
    BUFNO=2,DDNAME=
```

The QISAM load and OSAM DCB's are used to create a HISAM data base in a Type 3 region. The QISAM scan and OSAM DCB's are used to read, update, and add to a HISAM data base in a Type 3 region. The BISAM read/write and OSAM DCB's are used by Data Language/I in the Type 0 region for servicing requests from Type 1 and Type 2 processing regions.

Items 9, 10, 10a, 11, 11a, and 20 in the Systems Operation checklist must be accomplished before considering DBD generation complete.

The output Assembler Language listing from step 1 of a DBD generation includes an estimate of the cylinder index space and prime data set space for all ISAM data sets with a Data Language/I data base.

It is assumed that during IMS/360 system definition, Chapter 4 of this manual, the user did not specify a name for the DBD library; the default name, IMS.DBDLIB, is therefore used in the generated DBD procedure. The generated procedure is DBDGEN, which is a two-step assemble and link-edit procedure to produce data base definition blocks.

JCL for DBD Generation

Use this procedure, DBDGEN, when running each DBD generation. The JCL cards are:

```
//DBDGEN JOB MSGLEVEL=1
//       EXEC     DBDGEN,MBR=
//C.SYSIN  DD  *
         ⎧ DBD     ⎫
         ⎪ DMAN    ⎪
         ⎪ SEGM    ⎪
         ⎨ FLDK    ⎬     DBD generation control cards
         ⎪ FLD     ⎪
         ⎪ DBDGEN  ⎪
         ⎪ FINISH  ⎪
         ⎩ END     ⎭
/*
```

where keyword operand MBR= is the name for the DBD to be generated.

Management of Data Bases

Once the system is considered to be online with its data bases, constant surveillance of these data bases must be maintained. Many helpful facts can be gained from the statistics reports of IMS/360.

From the Transaction Response Report, response time can be obtained for the calls to the data base. In the Application Accounting Report, all requests to Data Language/I are counted. From the OS/360 IEHLIST utility, the remaining amount of space can be obtained to receive an indication of when a data base may need to be reorganized.

Some of the data base statistics which the Systems Operation function should consider are:

1. Total data base record volume (total number of root segments)

2. Number of records in prime area and number in overflow area

3. Number of unused tracks in overflow area
4. Number of records in the data base marked for deletion
5. Average number of occurrences of a segment type per parent segment

Management of data bases could evolve into the concept of a data dictionary. (A data dictionary is a descriptive foundation of all data used in the data base environment.) The descriptive information relates to fields, segments, data sets, data bases, and data base interaction.

IMS/360 TELECOMMUNICATIONS CONSIDERATIONS

The telecommunications facilities of IMS/360 are characterized by the use of remotely located input/output terminals connected to a System/360 computer through a communications network.

Communication Terminals and Lines

The physical communication terminals supported by IMS/360 are IBM 2260 Model 1, 2740 Model 1, and 1050 Model 1 or Model 2. These terminals may be connected to a System/360 computer through leased, nonswitched communication lines or by a common-carrier switched communications network. The 2260 may be attached only via nonswitched communication lines. To interface with IMS/360, the user of a physical communication terminal attached through a common-carrier switched network must dial the System/360 using the data set attached to the remote terminal. IMS/360 supports either a single terminal or multiple terminals on a communication line.

The features of the above terminals, the communications equipment, and the System/360 control units required for proper IMS/360 support are described in the manual titled Information Management System/360 for the IBM System/360, Application Description Manual (GH20-0524).

The remainder of this discussion concerning IMS/360 telecommunication facilities describes physical communication terminals as physical terminals. All physical terminals of the same type (that is, 2260, 2740, or 1050) which are attached through the same communication line facilities (that is, switched or nonswitched) and which utilize the same polling technique (that is, autopoll or poll) are considered by IMS/360 to be part of the same BTAM data set line group. Therefore, it is required that a user of IMS/360 describe a separate line group, via the LINEGRP macro, to IMS/360 system definition for each of the following configurations that is employed:

1. 2740 nonswitched with station control and autopoll
2. 2740 nonswitched with station control and poll
3. 1050 nonswitched with autopoll
4. 1050 nonswitched with poll
5. 2740 switched with transmit control
6. 1050 switched
7. 2260 remote mode, nonswitched with station control

For further definition of a BTAM data set line group, reference should be made to Operating System/360, Basic Telecommunications Access Method (GC30-2004). At least one communication line must exist within each line group. At least one physical terminal must exist for each communication line.

The master terminal of IMS/360 must be a 2740 or a 1050 physical terminal attached through a nonswitched communication line. It must be either the only terminal on the line or the first terminal on a multidrop line.

Master Terminal

The master terminal is the heart of IMS/360. Particular attention should be given to the caliber of operator selected for the position. The operator should have knowledge of all the operating aspects of the system. The Systems Operation function should decide whether the master terminal operator is adequately trained.

The physical location of the master terminal with reference to the computer console is important. If for security reasons they are not in close proximity, telephone communication must be provided.

The details of starting the system, checkpoint, restart, and all the other commands available to the master terminal are contained in the IMS/360 Operations Manual, Volume II - Machine Operations.

IMS/360 Systems with No Master Terminal

It is strongly recommended that IMS/360 systems be configured with a master terminal. Under certain conditions, however, it may be impractical to provide a master terminal facility; for example, at an installation intended to support only a small number of switched network terminals. A specific example might be a Data Center where IMS/360 communications activity might be scheduled infrequently for demonstration purposes only.

The System/360 console may serve as a master terminal input facility for the majority of IMS/360 commands and as an output facility for most IMS/360 messages intended for the master terminal. The following IMS/360 master terminal functions are not supported for the system console:

- Transaction message input from the system console
- Message switching message input from the system console
- Transaction or message switching output
- /BROADCAST commands (neither input nor output)
- /DISPLAY, /RDISPLAY commands (neither input nor output)
- /TEST input commands
- /EXCLUSIVE input commands
- /END input commands

System Definition Considerations with No Master Terminal

Stage 1 input to the system definition procedure must include a definition of a master terminal. See "System Definition Examples" in Chapter 4 for appropriate definition.

Punched output from Stage 1 includes a DCB open list, which includes a pointer to the master terminal DCB. This list is located at label DFSICDB within module DFSICLL0. Users intending to execute without a master terminal may remove the two cards representing the address word for the DCB that is not to be opened. If the DCB pointer was the last entry in the list, adjust the end-of-list indicator byte, X'80', in the preceding address word. A temporary alternative, and more easily implemented, would be to simply remove the DD cards generated for that communication line group in the IMS0 and IMS1 cataloged procedures.

Format of IMS/360 Commands Entered from System Console

```
r nn,'/NRESTART CHKPT 0'
r nn,'/START LINE 4'
```

Format is that of Operating System/360 write-to-operator reply messages. Alphabetic information within quotation marks may be uppercase or lowercase.

Remote Terminals

The section entitled "Line and Terminal Network" deals with the initiation of IMS/360 and remote communication lines and terminals. The details of the remote terminal command language are contained in both the IMS/360 Program Description Manual and the IMS/360 Operations Manual, Volume II - Machine Operations.

The training of the remote terminal operator should be monitored by the Systems Operation function.

There should be communication between the master terminal operator and the remote terminal operators, probably by way of the /BROADCAST command.

In the following discussions, references to "physical terminal" always refer to the relative physical terminal on the line.

IMS/360 Logical Terminals

In addition to the definition and presence of physical terminals, communication lines, and BTAM communication line groups, IMS/360 requires a user to define one or more logical terminals for each physical terminal. A logical terminal is a resource within IMS/360 which is identified by a one- to eight-character alphameric name. Each logical terminal name within IMS/360 must be unique and must begin with an alphabetic character. The logical terminal is the means by which IMS/360 classifies input and output message data for one or more users of a particular physical terminal. The following list presents some of the reasons for the use by IMS/360 of the logical terminal concept.

1. Associated with each logical terminal is security definition. Each logical terminal may have unique or overlapping security definition with any other logical terminal defined within IMS/360.

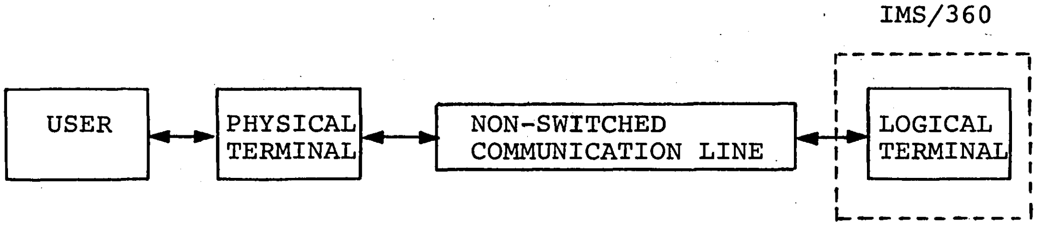
2. Multiple logical terminals may be associated with a single physical terminal. This can facilitate the use of a single physical terminal by multiple users, each associated with his unique logical terminal, especially if each logical terminal has a unique security definition.

3. The logical terminal is the interface between IMS/360 and the terminal operator and in addition, is the interface between an application program and a physical terminal. Using this approach, an application program can be insensitive to the idiosyncracies of a particular physical terminal and the communications network type by which it is attached. A significant degree of equipment independence is achieved, because the logical terminal provides a symbolic interface to the application program.

4. Because a logical terminal is a resource within IMS/360, it can be dynamically associated with different physical terminals by means of the /IAM and /ASSIGN commands. These commands thus allow the IMS/360 user an additional degree of flexibility and reliability in the use of his physical terminals.

Logical Terminal/Physical Terminal Relationship on Nonswitched Communications Network

The best manner in which to describe the relationship between a terminal user, a physical terminal, a communication line, and a logical terminal is a diagram:

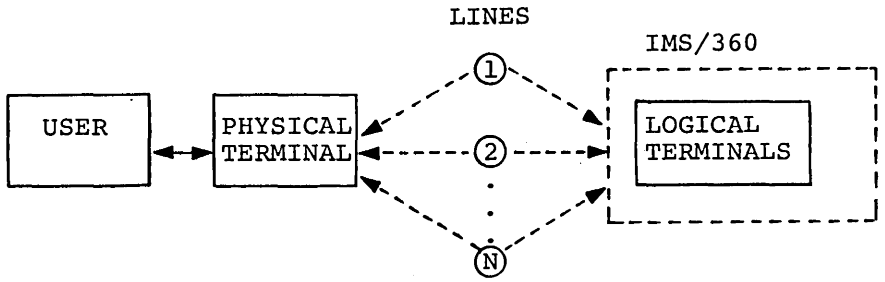


When an IMS/360 system user defines the IMS/360 system to his data processing environment, this definition includes the characteristics and relationships of physical terminals, communication lines, and logical terminals. On a nonswitched communication line, the relationship between a physical terminal at one end and a logical terminal within IMS/360 at the other is a stable relationship defined at system definition time. If there is only one user of a particular physical terminal, typically there is a one-to-one relationship between physical terminal and logical terminal. However, if a particular physical terminal is employed by multiple users, it may be more typical to have many logical terminals associated with the physical terminal. Perhaps the system definition includes a logical terminal for each user of a particular physical terminal.

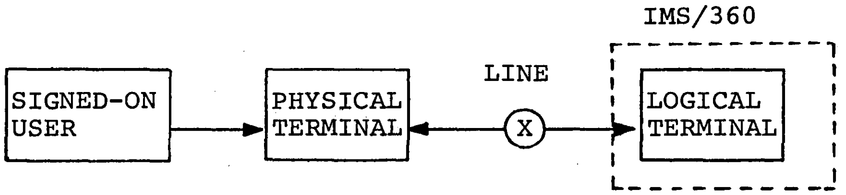
Once the relationship is established between a physical terminal and one or more logical terminals at system definition, the association can be changed only through the /ASSIGN command or a new system definition. The /ASSIGN command is normally executable from the master terminal only.

Logical Terminal/Physical Terminal Relationship on Switched Communications Network

The logical/physical terminal relationship on a switched communications network is considerably more complex than in the nonswitched communication line environment. The IMS/360 system definition is again the process which defines the characteristics of the physical terminals, communication lines, and logical terminals. However, the relationship between a particular physical terminal and a logical terminal is not established until the remote terminal user dials the System/360 computer to communicate with IMS/360. The relationship between a terminal user, a physical terminal, a communication network, and IMS/360 logical terminals at system definition time is depicted in the following diagram:



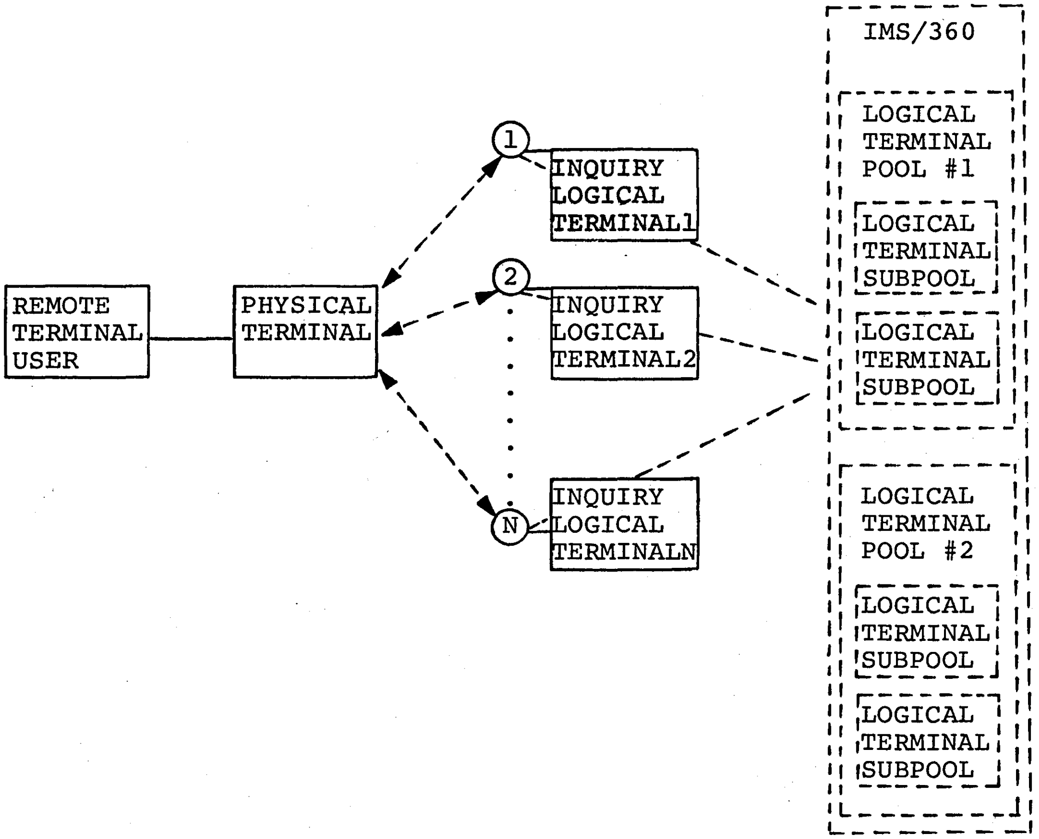
Once the remote terminal user dials the System/360 computer and issues the /IAM command to sign himself on to IMS/360, a stable relationship between the physical terminal and one or more logical terminals is established.



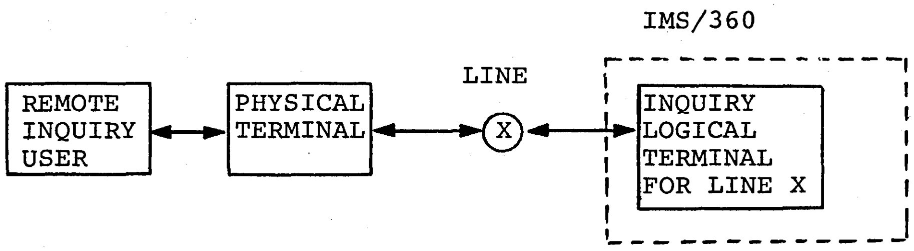
Logical Terminal Types in Switched Communications Network Environment

In a switched communications network environment, the IMS/360 user employs system definition to define one or more communication lines. One logical terminal must be associated with each of these lines. This logical terminal is designated as the inquiry logical terminal for the dialable communication line. In addition to an inquiry logical terminal for each dialable communication line, pools of logical terminals may be defined at system definition time. One or more logical terminals from the pools of logical terminals are associated with a particular line when a remote terminal user dials the IMS/360 system. The number of logical terminal pools which are defined for a switched communications network depends upon the number of WATS areas employed by an IMS/360 user. There is a one-to-one relationship between a WATS area and a logical terminal pool.

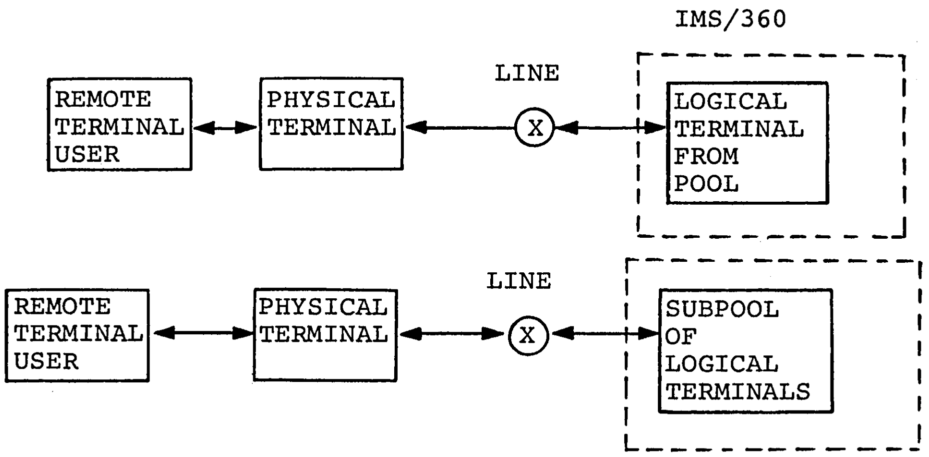
Within any logical terminal pool for a switched communications network, the IMS/360 user can define logical terminal subpools. A logical terminal subpool is composed of one or more logical terminals within a given logical terminal pool. A particular logical terminal can exist in only one pool and subpool. A remote terminal user can dial the IMS/360 system and sign on for a single logical terminal or for all logical terminals within a logical terminal subpool. At system definition, the environment appears as indicated in the following diagram:



After a remote terminal user has dialed a System/360 computer operating under IMS/360, several situations can exist. If the /IAM command is used to sign on and the LTERM parameter specifies the inquiry logical terminal, the following diagram applies:



If the /IAM command is used to sign on and the LTERM parameter specifies a logical terminal from the pools of logical terminals, the following diagram applies:



If the /IAM command is used to sign on and the LTERM and PTERM parameters are specified, all logical terminals within a subpool are associated with the physical terminal.

The use of the logical terminal subpool concept allows for efficient use of communication facilities. All output queued on each of the logical terminals in the subpool for which the /IAM command was issued is sent to the physical terminal.

A subpool may be defined to contain the logical terminals for all of the users of a single physical terminal. While a user is signed onto a logical terminal within the subpool, the subpool is unavailable to users signing on from other physical terminals. This is true whether or not the PTERM parameter is used in the /IAM command.

All inquiry logical terminal names must begin with the same first four characters. When signing on for an inquiry logical terminal, only the first four characters are considered significant by IMS/360. This permits a user to telephone in on any Autoanswer line and to sign on for and use the inquiry logical terminal for inquiry transactions only, if he simply knows the first four characters. The inquiry logical terminal can be used only for immediate-response, nonupdate transactions, and queued output is preserved on it for the duration of signon. So that IMS/360 can distinguish inquiry logical terminal names from subpool logical terminal names at signon, no subpool logical terminal name may begin with the first four characters used for inquiry logical terminal names.

Line and Terminal Network

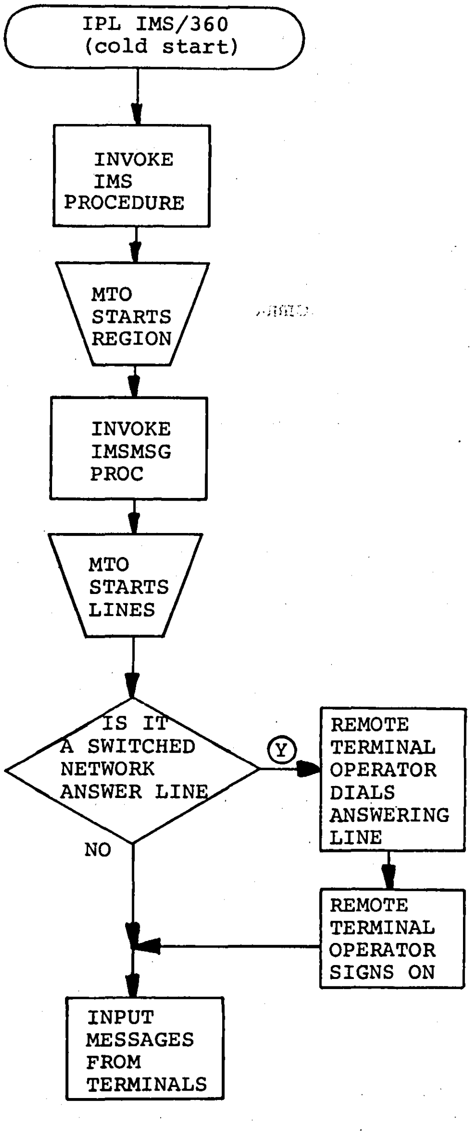
The planning and logistics of the teleprocessing line and terminal network must be considered. The IBM Field Engineers and the resident telephone company should have made their final checkout of the network, thus giving the assurance if IMS/360 is operational that the network will function.

An additional review of the Systems Operation checklist at this time should perhaps be made. A look at the section titled "Application | Development and Structuring of IMS/360" in the IMS/360 Program Description Manual may also be beneficial at this time.

The IMS/360 security maintenance program need not be executed before the IPL of the IMS/360 teleprocessing system. If password and terminal security are later installed, they become effective at the next "cold start" of IMS/360 or at the master terminal operator's option at the next "warm start".

The Systems Operation function should keep records of the physical-line-to-physical-terminal relationship and the physical-to-logical-terminal relationship, and of their relationship to the particular application program. Add the security maintenance characteristics, when they exist, and any information from the Machine Operations function as to line and terminal trouble reports.

When lines and terminals are installed and operating, when PSB generation and DBD generation are complete, and when system definition is complete, then, and only then, can the line and terminal network pertaining to an IMS/360 teleprocessing application be employed. The steps are as follows:


IPL IMS/360 (cold start)
INVOKE IMS PROCEDURE
MTO STARTS REGION
INVOKE IMSMSG PROC
MTO STARTS LINES
IS IT A SWITCHED NETWORK ANSWER LINE
Y
NO
REMOTE TERMINAL OPERATOR DIALS ANSWERING LINE
REMOTE TERMINAL OPERATOR SIGNS ON
INPUT MESSAGES FROM TERMINALS

This procedure executes IMS/360 Type 0 region, the online processing region.

Master terminal operator (MTO) starts region.

This procedure executes IMS/360 Type 1 processing region.

Master terminal operator starts all communication lines.

IMS/360 Terminal Commands and Messages

This section serves only to introduce the commands that must be considered in the development of an IMS/360 system. A quick index follows. For an operational discussion and details of terminal commands, refer to the IMS/360 Operations Manual, Volume II - Machine Operations.

For the purpose of dynamically interrogating or altering the processing functions of IMS/360, special messages may be entered from terminals. These messages are known as command messages and are designated by a slash in the first position of the message.

Most command messages are limited to a single line in length. Any command message results in the issuance of completion or error messages to the originating and affected terminals. Furthermore, these response messages override any limiting status of a particular line or terminal.

Certain command messages should be restricted to entry from the master terminal, the source of systems control and information messages. These are the messages that interrogate, alter, and control the overall system. The System/360 console itself may be used as a master terminal in relation to the entry of master terminal commands.

Other command messages may be entered from any terminal, within the limitations of user-defined terminal security provisions. The function of the remote terminal command language is to change the status or mode of operation of the user's own terminal to provide extended security facilities and to provide extended user message entry or data output facilities.

The master terminal commands for restart, checkpoint, data base dump, and data base recovery are treated separately in a later section of this chapter.

Refer to the Machine Operations Manual for a description of the commands. The following is provided as a guide in determining which command to use.

Command	Explanation of Command
1. /ASSIGN	This command: • correlates a specified logical terminal with a specific physical terminal • temporarily assigns a current priority to one or more specific transaction codes • assigns a particular limit priority to one or more specific transaction codes • assigns a particular normal priority to one or more specific transaction codes • assigns a particular limit count to one or more specific transaction codes • assigns a particular processing limit count to one or more specific transaction codes
2. /BROADCAST	This command transmits messages to one or more terminals.
3. /CANCEL	This command causes the cancellation of the complete message currently

being entered into the same terminal.

4. /CHANGE

   This command is used to change one password to another.

5. /CHECKPOINT

   (See later section in chapter.)

6. /DBDUMP

   (See later section in chapter.)

7. /DBLOG

   This command starts data base segment logging, which allows backout of data base modifications during emergency restart.

8. /DBNOLOG

   This command stops data base segment logging.

9. /DBRECOVERY

   (See later section in chapter.)

10. /DELETE

    This command:

    - eliminates password security for one or more transaction codes, physical terminals, logical terminals, programs, or data bases

    - eliminates terminal security for one or more transaction codes

11. /DISPLAY

    This command displays critical fields of certain IMS/360 control blocks and system queues.

12. /END

    This command terminates the mode initiated through the /TEST or /EXCLUSIVE command.

13. /ERESTART

    (See later section in chapter.)

14. /EXCLUSIVE

    This command places the user's terminal into exclusive use or inquiry mode.

15. /IAM

    This command allows a terminal user at a switched line terminal to identify himself. Required if a switched (dialup) terminal.

16. /LOCK

    With keyword TRAN, do not schedule this transaction code.

    (If a particular transaction code cannot be processed correctly, use this command at the remote terminal to ensure that this transaction code is not scheduled.)

    With keyword PROGRAM, do not schedule this program.

    (If a particular program cannot be executed correctly, use this command at the remote terminal to ensure that this program is not scheduled or used.)

With keyword DATABASE, do not schedule any program that uses this data base.

(If a particular data base is not correct, use this command at the remote terminal to ensure that no program is scheduled that uses this data base.)

With keyword PTERM, queue output but do not send output to this physical terminal.

(PTERM applies to the physical terminal into which the command is entered. A password may be included with the keyword PTERM; no parameters are acceptable. /LOCK and /UNLOCK are used with nonimmediate-response-type messages only. The user can enter a series of nonimmediate-response-type messages and lock his terminal. No response will be printed on the terminal until such time as the terminal is unlocked. Exception: system messages will be printed.)

With keyword LTERM, queue output but do not send to these logical terminals.

(These commands are used with nonimmediate-response-type messages only. The user can enter a series of nonimmediate-response-type messages and /LOCK his terminal. This normally implies that the messages must be secured by logical terminal, since the user must know what logical terminal or terminals to lock. No responses will be printed on the terminal until such time as the terminal is unlocked. Exception: system messages will be printed.)

17. /LOG

This command:

- causes the contents of the message entered at this terminal to be logged but not processed by a program. The slash is the first character logged.
- applies only to the currently entered message line and does not establish a continuing operational mode

18. /NRESTART

(See later section in chapter.)

19. /PSTOP

With keyword LINE:

- do not receive input
- do not send output
- queue output

With keywords LINE and PTERM:

- do not receive input
- do not send output
- queue output

With keyword LTERM:

- queue output messages
- do not send messages to this logical terminal

With keyword TRAN:

- queue input
- do not schedule this transaction code

20. /PURGE

(/PURGE and /STOP stop queuing of output only if the message to be queued originates at the terminal (message switching). Output from an application program is always queued.)

With keyword LINE:

- do not poll
- send output
- do not queue output

With keywords LINE and PTERM:

- do not receive input
- send output
- do not queue output

With keyword LTERM:

- do not queue output messages
- send messages to this logical terminal

With keyword TRAN:

- do not queue input
- schedule this transaction code

21. /RDISPLAY

This command displays the identification of the master terminal.

22. /RESET

This command negates the action of the /SET command.

23. /SET

This command allows the setting of a destination mode for messages entered thereafter into the entering physical terminal. The destination must be a TRAN code or an LTERM (message switching), thereby eliminating the use of a destination code.

24. /START

With keyword LINE:

- poll
- send output
- queue output

With keywords LINE and PTERM:

- receive input
- send output
- queue output

With keyword LTERM:

- queue output messages
- send messages to this logical terminal

With keyword TRAN:

- queue input
- schedule this transaction code

With keyword DATABASE, schedule a program using this data base.

With keyword PROGRAM, schedule this program.

With keyword REGION, use the facilities of OS/360 to start a message region (one).

(This command is cumulative in effect. To start two message regions, the command is entered twice. The processing is also done on a net basis. If /START were entered once, the net result would be to start one message region.)

25. /STOP

(/PURGE and /STOP stop queuing of output only if the message to be queued originates at a terminal (message switching). Output from an application program is always queued.)

With keyword LINE:

- do not poll
- do not send output
- do not queue output

With keywords LINE and PTERM:

- do not receive input
- do not send output
- do not queue output

With keyword LTERM:

- do not queue output messages
- do not send messages to this logical terminal

With keyword TRAN:

- do not queue input
- do not schedule this transaction code

With keyword DATABASE, do not schedule a program using this data base.

With keyword PROGRAM, do not schedule this program.

With keyword REGION, use the facilities of OS/360 to terminate a message region (one).

(This command is cumulative in effect. To stop two message regions, the command is entered twice. The processing is also done on a net basis. If /STOP were entered once, the net result would be to stop one message region.)

26. /TEST

This command implies that no independent messages will be transmitted to the user's terminal. Messages entered into the user's terminal are transmitted back to the user's terminal.

27. /UNLOCK

With keyword TRAN, schedule this transaction code.

With keyword PROGRAM, schedule this program.

With keyword DATABASE, a program may be scheduled that uses this data base.

With keyword PTERM, queue output and send output to this logical terminal.

(PTERM applies to the physical terminal into which the command is entered. A password may be included with the keyword PTERM; no parameters are acceptable. /LOCK and /UNLOCK are used with nonimmediate-response-type messages only. The user can enter a series of nonimmediate-response-type messages and /LOCK his terminal. No response will be printed on the terminal until such time as the terminal is unlocked. Exception: system messages will always be printed.)

With keyword LTERM, queue output and send output to these logical terminals.

A detailed discussion of the commands is in the IMS/360 Operations Manual, Volume II - Machine Operations.

IMS/360 MESSAGE QUEUES

Because an understanding of IMS/360 message queues affects the decision for Items 7c, 7j, and 16 of the Systems Operation checklist and restart, the following is presented.

The IMS/360 control program provides the capability to queue messages received on direct access storage and in core storage. Messages may be received from communication terminals or application programs and may be destined for communication terminals or application programs. A message destined for an application program is called a transaction and begins with a transaction code. All transactions of the same type (same code) are queued in a serial chain based upon time of receipt by IMS/360. A serial queue exists for each defined transaction code. All messages destined for a particular communications logical terminal are queued serially like transactions. A serial queue exists for each defined logical terminal (Figure 16).

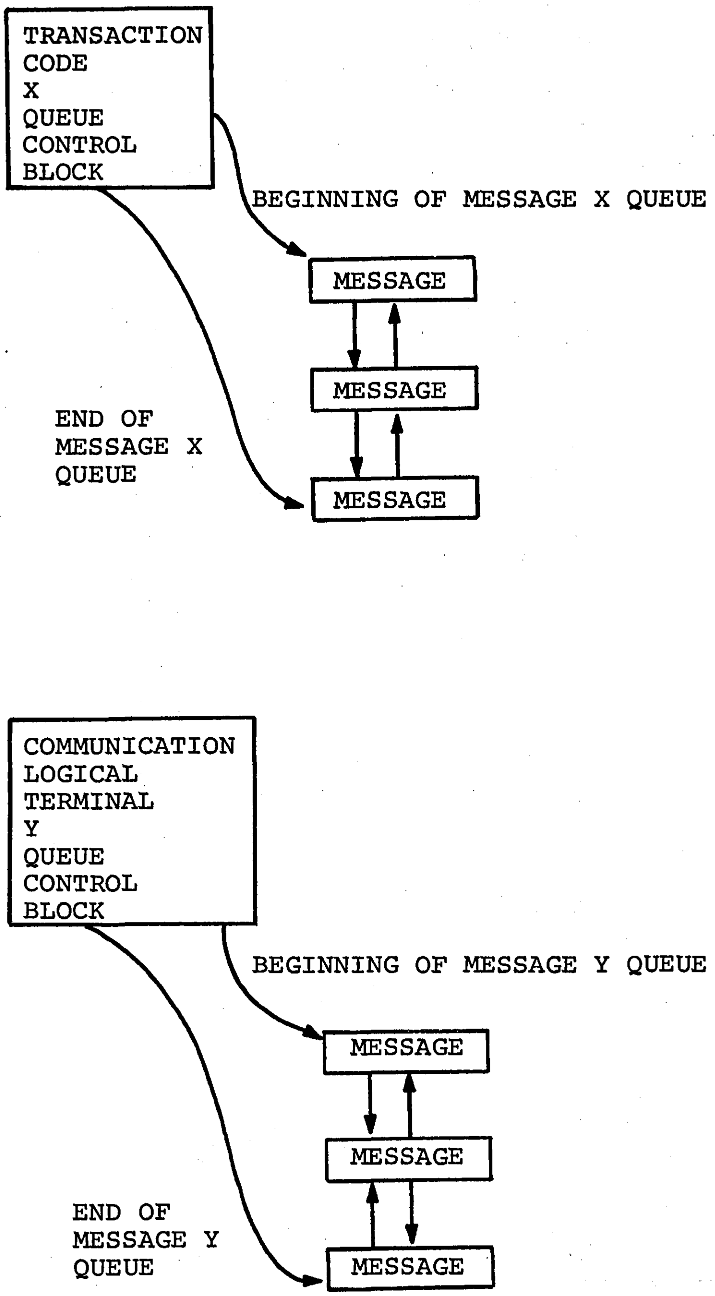

TRANSACTION CODE X QUEUE CONTROL BLOCK
BEGINNING OF MESSAGE X QUEUE
MESSAGE
MESSAGE
END OF MESSAGE X QUEUE
MESSAGE
COMMUNICATION LOGICAL TERMINAL Y QUEUE CONTROL BLOCK
BEGINNING OF MESSAGE Y QUEUE
MESSAGE
MESSAGE
END OF MESSAGE Y QUEUE
MESSAGE

Figure 16. IMS/360 message queues

All messages received are written to direct access data sets. However, the core storage buffers used by the IMS/360 control program are used on a rotating basis, thus retaining an image of the message in core as long as possible. If the message still exists in core storage when it is dispatched to its destination (input to a program or output to a terminal or another program), reference to the direct access data sets is not necessary, since the IMS/360 control program uses the incore storage image. All messages received are written to direct access data sets to ensure that a copy is available if the IMS/360 system or Operating System/360 fails (the contents of core storage are lost). In addition, all messages received are written to the IMS/360 system log in consideration of possible failure of the direct access data set queues. The reuse of core storage buffers that already contain messages which have not been sent to their destinations is based upon the time that the message has remained in the system; the oldest buffer is used first.

Messages received may be represented by single or multiple lines of text. If a message is represented by multiple lines of text, the queues are stored on direct access and in core storage in a blocked format. The blocking factor is determined by the device upon which the various queue data sets are resident.

The IMS/360 control program utilizes OSAM data sets for direct access queue storage. Either two or four data sets are required: one or two data sets are used for input and output of single-line messages, and one or two data sets for input and output of multiple-line messages. The choice of two or four data sets is made by the IMS/360 user at system definition time (Item 7j on the Systems Operation checklist). Figure 17 shows examples of storage using two or four data base sets.

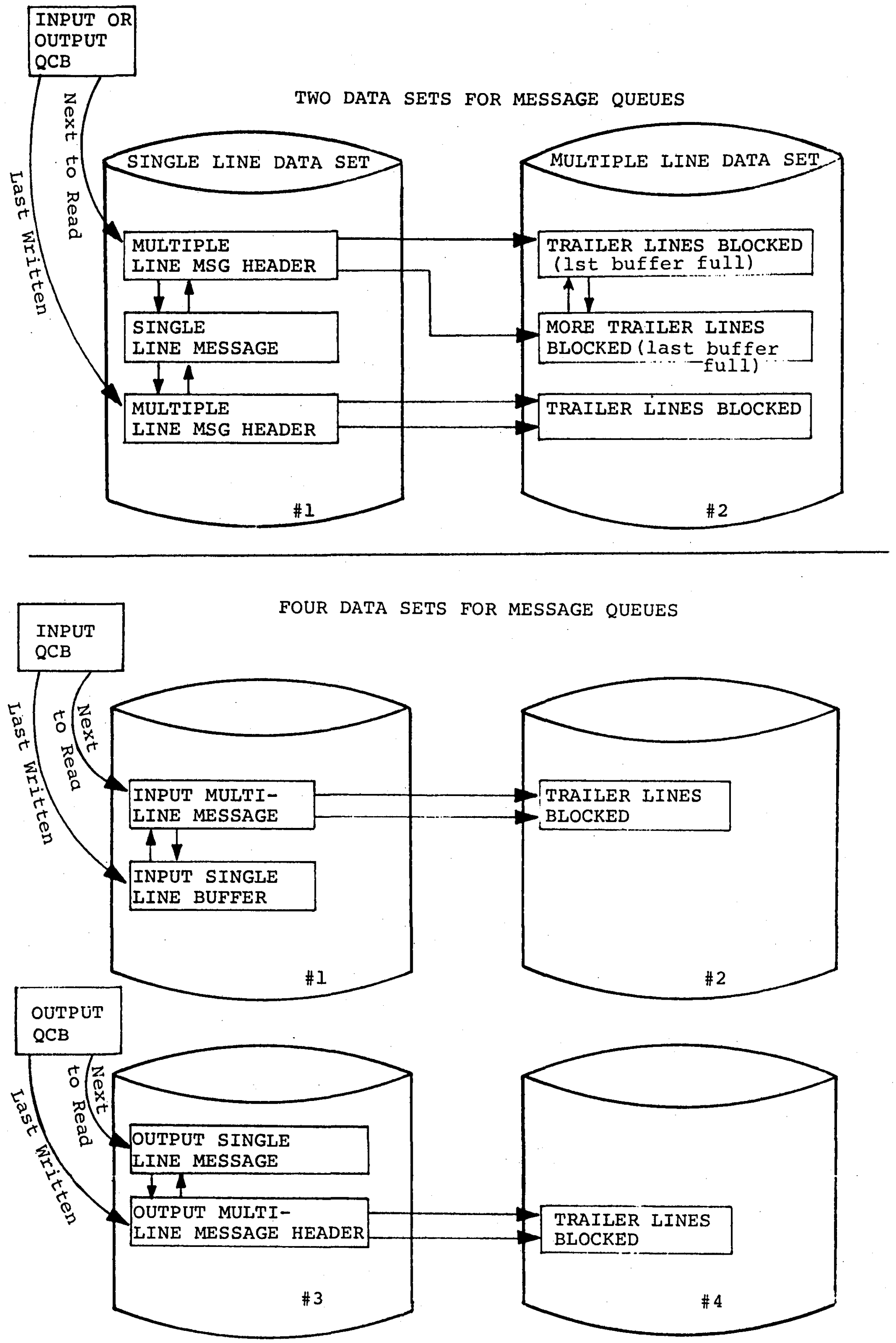


Figure 17. Examples of two or four data base sets for direct access queue storage

The IMS/360 message queue data sets must be preformatted before initial usage. The preformatting is performed by restarting IMS/360 with a request to format. The need to reformat the message queues arises only if an input/output error occurs within a queue data set.

The use of preformatted queues provides increased performance and reliability. Performance is increased through the preassignment of direct access storage records for any chain of messages, resulting in a reduction in the number of input/output operations for management of the queues. Reliability is increased with the preformatted data sets because the count field of the direct access device record X is not relied upon to write record X+1. Since direct access space is allocated sequentially on a chronological basis even though the queue chains are random, a write error results in the assignment of the next available direct access record. A write error does not result in the inability to write subsequent records in the data set as might be the case with unformatted queue data sets. The effect of a write error is the automatic assignment by IMS/360 of an alternate direct access record (the next sequential record in the data set). The preformatted record in which a write error is encountered is skipped over. No queue chain points to this record; in effect, it is lost space on the direct access volume. Approximately 10 seconds is required to format each 2314 cylinder without write-checking and 20 seconds with write-checking in an IMS/360 message queue data set.

Until formatted space within a direct access device data set used by IMS/360 for message queues is exhausted, records are allocated on a sequential basis from the beginning to the end of the data set and no reuse of space is attempted. When an entire data set is exhausted, IMS/360 will begin to reuse space (records) which no longer contains active messages (are already sent to their destinations). Once reuse of data set records occurs, a reduction in queue performance is experienced because of the need to maintain a free queue of direct access space. The IMS/360 system may subsequently be terminated with a checkpoint purge or dump queue. A restart with build queue operand after a purge or dump queue causes the allocation of queue space to be reinitialized to the beginning of the queue data sets.

In order to provide for message queue recoverability if the queue data sets are destroyed, the IMS/360 control program logs:

1. All input and output message text

2. The queue pointers to each message queue chain whenever a message is enqueued onto or dequeued from the chain

If a system failure occurs and the message queue data sets are retained intact, the restart facilities of IMS/360 can properly reposition the queues by use of the enqueue/dequeue pointers which were logged. If the queue data sets are destroyed, the restart facilities of IMS/360 can be employed to rebuild the queues from the log entries of message text.

Message Queue Space Allocation

The amount of direct access storage space allocated to the message queue data sets depends upon how many data sets are used (two or four), how many messages are received and sent to terminals, and the length of the messages received and sent. The best way to provide guidance for space allocation is to consider a specific example. Assume:

1. The system has four message queue data sets.

2. The multiple-line data sets contain physical blocks equal to five message lines (one message line and prefix is approximately 200 characters -- multiple-line buffer for five lines is approximately 700 characters; the other four lines do not contain a prefix).

3. 50,000 input messages a day (12 hours of operation) and 50,000 output messages are handled.

4. 10,000 of the 50,000 input messages are multiple-line messages with an average of five lines. 25,000 of the 50,000 output messages are multiple-line messages with an average length of ten lines.

5. Nine multiple-line OSAM records per 2314 track and 20 single-line OSAM records per 2314 track.

- First, calculate the single-line input/output message space required:

$$\frac{50{,}000}{20} = \frac{\text{single-line messages}}{\text{single-line messages/track}}$$

input single-line input tracks	=	2500	
input single-line cylinder	=	125	cylinders
output single-line cylinders	=	125	cylinders

- Second, calculate the multiple-line input message space required:

$$\frac{10{,}000}{9} = \frac{\text{multiple-line input messages}}{\text{multiple-line messages/track}}$$

1112 = tracks for multiple-line input messages

56 = cylinders for multiple-line input messages

- Third, calculate the multiple-line output message space required:

$$\frac{2\text{x}25{,}000}{9} = \frac{\text{records per output message x multiple-line output messages}}{\text{multiple-line output message/track}}$$

5,556 = tracks for multiple-line output messages
278 = cylinders for multiple-line output messages

The total space requirements are 584 cylinders of 2314. This space requirement is of course an outside limit because no consideration has been given to the reuse of message queue space. Although reuse of direct access space is quite practical, it does reduce the efficiency of message queuing (Item 7j on the Systems Operation checklist).

Using the above example, assume also:

200 transaction types (SMB's)
200 logical terminals (CNT's)

Since reuse of queue space is dependent upon the turnover of messages, calculate the average message turnover per QCB per hour:

$$\frac{50{,}000}{12\text{x}200} = \frac{\text{total messages to be processed}}{\text{number of hours x number of queue blocks (SMB's or CNT's)}}$$

21 = number of messages to be turned around per QCB per hour

To run three hours before reuse begins, allocate

$$\frac{3\text{x}21\text{x}200}{20} = \frac{\text{hours x turnovers x number of queue blocks}}{\text{single-line messages (QCR's) per 2314 track}}$$

Input single-line tracks $= \frac{12{,}600}{20} = 630$

Input single-line cylinders $= \frac{630}{20} = 32$

Output single-line cylinders$= \quad = 32$

It was determined that one out of five input messages is a multiple line. Therefore,

$$\frac{12{,}600}{5\text{x}9} = \frac{\text{multiple-line records x number of messages in time period}}{\text{total messages x multiple-line records per 2314 track}}$$

Input multiple-line tracks $= \frac{3150}{9} =$ 350 tracks

Input multiple-line cylinders$= \frac{350}{20} =$ 18 cylinders

$$\frac{(2\text{x}25{,}000) \text{ x } 12600}{50{,}000 \text{ x } 9} = \frac{\text{multiple-line records x number of messages in time period}}{\text{total messages x multiple-line records per 2314 track}}$$

Output multiple-line tracks $= \frac{12{,}600}{9} =$ 1400 tracks

Output multiple-line cylinders $= \frac{1400}{20} =$ 70 cylinders

The total space requirements are 152 cylinders of 2314, if reuse is allowed after three hours of running. The overhead incurred when reuse has begun is as follows:

1. For each record written, one read is added to obtain a "next" record pointer.

2. For each QCR record written which had message buffer(s) attached, a message buffer must be read and rewritten to provide a contiguous chain of message buffer records.

3. When the last QCR of a string has been reused, at least two additional QCR records must be read and one QCR rewritten to obtain a new string for reuse.

The I/O operations per hour prior to reuse are:

Input QCR writes	4150
Input message buffer writes	1050
Output QCR writes	4150
Output message buffer writes	4150
Total	13,500

The I/O operations per hour after beginning reuse are:

		Writes	Reads	Total I/O
1.	Input QCR	4150	4150	8300
	Output QCR	4150	4150	8300
	Input MSG buffer	1050	1050	2100
	Output MSG buffer	4150	4150	8300
		13,500	13,500	27,000

2. String MSG buffers for reuse:

Number of strings $= \frac{\text{number of multiple-line message records}}{\text{records per message}}$

$$\text{Input} = \frac{1050}{1} = 1050 \text{ strings}$$

$$\text{Output} = \frac{4150}{2} = 2075 \text{ strings}$$

	Writes	Reads	Total I/O
String input message buffers	1050	1050	2100
String output message buffers	2075	2075	4150
	3125	3125	6250

3. To obtain QCR strings for reuse:

Since reuse begins after three hours of operation and all records have been reused after six hours, the average QCR string may be calculated:

$$\text{hours x MSG} \quad \frac{3x21+6x21}{2} = \frac{189}{2} = 94$$

The average QCR string obtained is two less than the actual QCR string:

$$94-2 = 92$$

$$\text{Searches for Input QCR string} = \frac{\text{records required}}{\text{string length}}$$

$$= \frac{4150}{92}$$

$$= 46$$

Since the output message queuing requires two physical queues per CNT, the average QCR string is half as long as an input string:

$$\frac{94}{2} - 2 = 45$$

$$\text{Searches for output QCR strings} = \frac{\text{records required}}{\text{string length}}$$

$$= \frac{4150}{45}$$

$$= 93$$

	Writes	Reads	Total I/O
I/O for input QCR search	46	92	138
I/O for output QCR search	93	186	279
	139	278	417

The overhead imposed by reuse in this instance is:

27,000+6250+417-13,500 = 20,167 I/O operations per hour

The IMS/360 system definition execution can generate a procedure for execution of the IMS/360 control program in the Type 0 processing region. This procedure includes DD cards for the message queue data sets but assumes the user has allocated the data sets. Message queue data set allocation is the responsibility of the IMS/360 user. The message queue data set DD cards should include the following parameters: SPACE= , DISP= , UNIT= , VOLUME=SER= , and DCB=(DSORG=PS).

Message Queue Space Allocation - Secondary

For most efficient operation, message queue data set space should be allocated in terms of contiguous cylinders on separate devices. Secondary allocation should not normally be requested. If secondary allocation is requested, all 16 extents will be obtained at the time the queue data set is formatted.

IMS/360 CHECKPOINT, RESTART, DATA BASE DUMP, AND DATA BASE RECOVERY

| This section attempts to provide the reader with a description of the checkpoint and restart facilities of IMS/360. The operational details of checkpoint, restart, data base dump, and data base recovery then follow in this manual and in the IMS/360 Operations Manual, Volume II - Machine Operations.

Checkpoint

The checkpoint facilities of the IMS/360 control program provide the means for periodically recording control information and status to enable IMS/360 restart after failure. This failure may be the termination of the IMS/360 control program or the loss of Operating System/360. In addition, the checkpoint facilities are the means for terminating the IMS/360 system in an orderly way, creating a tape image (backup) of a data base used for message processing, or assisting in the reconstruction of a data base which has been destroyed. There are four checkpoint commands and two data base dump commands. All these commands are executed from the master terminal of IMS/360.

Simple Checkpoint

The first checkpoint command is /CHECKPOINT with no operands (simple checkpoint). It may be invoked automatically by the IMS/360 control program or from the master terminal. The automatic invocation of simple checkpoint is based upon the number of entries to the system log. The user of IMS/360 may specify the number of entries between system-invoked checkpoints during system definition. The simple checkpoint, like all other checkpoint commands, uses the IMS/360 system log for recording control data. The simple checkpoint logs the status of all dynamically changing IMS/360 control program blocks. These include the logical-to-physical-terminal relationships, the input and output message queue control blocks, the security blocks, and others. The simple checkpoint command causes the scheduling of programs into message processing regions to halt momentarily while the control block information is logged. The simple checkpoint command has no effect upon internal operations in the IMS/360 control program or operations upon the communication lines. As soon as the simple checkpoint command is terminated, scheduling into message regions is automatically initiated by the IMS/360 control program.

Checkpoint Freeze

The three remaining checkpoint commands are each used for orderly termination of the IMS/360 system. Each is invoked only from the master terminal. The checkpoint freeze command is the fastest means of orderly termination. Input communication lines are terminated as soon as any messages being entered are completely received. Output communication lines are terminated as soon as any messages being sent are completely transmitted. Message regions are terminated as soon as the current messages being processed have been completed. All input messages remaining to be processed and all output messages remaining to be transmitted are retained in the message queue data sets. The same mechanics as in simple checkpoint are now invoked to log the status of all control blocks. Finally, the checkpoint facility causes the termination of the IMS/360 control program job. The IMS/360 user should employ the /NRESTART command without message queue reconstruction to restart IMS/360 after a /CHECKPOINT FREEZE.

Checkpoint DUMPQ

The checkpoint dump queue command operates in exactly the same manner as the checkpoint freeze command, but performs the additional function of dumping all the input and output messages out of the message queue data sets. The /NRESTART command with message queue reconstruction should be employed to restart IMS/360 after a checkpoint dump queue termination. The restart of IMS/360 in this manner causes allocation of space in the queue data sets to start from the beginning of the data sets. The messages dumped from the queue data sets during the checkpoint dump queue command are reloaded into the message queue data set during the /NRESTART with-message-queue-reconstruction command.

Checkpoint PURGE

The checkpoint purge command is the most orderly yet most time-consuming manner of terminating IMS/360. The input communication lines are terminated first as soon as all messages being entered are completely received. All messages in the input queue are processed, and all resultant output messages are transmitted to their specified destinations (terminals, etc.). The message regions are then terminated, and output communication lines are stopped. Finally, any input messages which could not be processed or any output messages which could not be transmitted are dumped to the IMS/360 system log, and the IMS/360 control program job is terminated. The /NRESTART command with message queue reconstruction should be employed to restart IMS/360 after termination with a /CHECKPOINT PURGE command.

Data Base Dump

The data base dump capabilities of checkpoint include the functions of creating a dumped tape image of a complete data base and performing preparatory functions for the reconstruction of a data base.

The /DBDUMP command, which is also entered from the master terminal, creates a dump tape image by stopping all transaction input from terminals that would update the data base and processing all transactions already in the input queue against the data base. A special utility (a message processing program) is then scheduled for execution. This message processing program retrieves all segments from the data base with GET calls and creates a copy on an HSAM tape data base with ISRT calls. When the data base dump is complete, the tape volume containing the copy is unloaded. Finally, the update transactions that were stopped are again allowed entry from terminals. This command causes a Force End of Volume on the IMS/360 log so that a new log is started immediately after the data base dump. The user must create an application program, PSB for the program, and DBD for the HSAM data base if the dump copy is to be subsequently used to restore the data base. The application program is executed in a Type 3 region. The data base recovery command may then be used to reprocess transactions. The HSAM data base is composed of a BSAM data set with a name (DFSIDUMP.dbdname), where dbdname is the DBD name associated with the data base which was dumped.

A second form of the data base dump command is /DBDUMP with STOP. This command is used in preparatory procedures prior to data base reconstruction. The /DBDUMP with STOP command causes all update transactions against a data base that must be reconstructed to be retained in the input message queue. However, these transactions are not scheduled for processing. The continuance of input of these transaction types is allowed, but no processing occurs. The data base must be reconstructed with a batch program executed from an IMS/360 Type 3 region and a previously dumped copy of the data base. Once the data base is reconstructed with the last dump tape, all transactions from the data base dump until the current point in time must be reprocessed. This is accomplished with the old system log tapes and the data base recovery command.

The DD card for the tape used when dumping a data base to tape with the data base dump command is contained in the IMS0 procedure supplied by IMS/360 system definition. The DD name of the DD card is DBDUMP.

Checkpoint Guide

The following table may be used as a guide in determining which checkpoint command to use.

Command	When to Use
1. /CHECKPOINT	If a simple checkpoint is desired without terminating the IMS/360 system, use this command.
2. /CHECKPOINT FREEZE	Use this command if (1) IMS/360 must be terminated quickly, (2) the disk message queues will not be disturbed before restarting, or (3) the output messages can wait until later.
3. /CHECKPOINT DUMPQ	Use this command if (1) IMS/360 must be terminated, (2) the disk message queue space may be used for other purposes before restarting, or (3) the output messages can wait until later.
4. /CHECKPOINT PURGE	Use this command if (1) IMS/360 must be terminated, or (2) it is desired to process and send all messages currently in the system.

All the checkpoints output a message identifying the checkpoint. This message is of the following format:

```
*CHECKPOINT**yyddd/hhmmtt**type**serial
```

where:

yyddd is the Julian date.

hhmmtt is the time in hours, minutes, seconds.

(yyddd/hhmmtt together identify the checkpoint.)

type is the checkpoint type: SIMPLE, FREEZE, DUMPQ, or PURGE.

serial is the serial number of the volume on which the checkpoint was written.

The last three checkpoint commands, FREEZE, DUMPQ, and PURGE, terminate all message regions and the IMS/360 control region, region 0. For a better understanding of the sequence of events that take place with each checkpoint command, see Figure 18. The numbers in each column of Figure 18 indicate the sequence of events occurring for that checkpoint type. The Abnormal End column defines the termination of IMS/360 if an abnormal condition occurs that requires immediate termination of the IMS/360 control program. Under normal operation, this column should never be used.

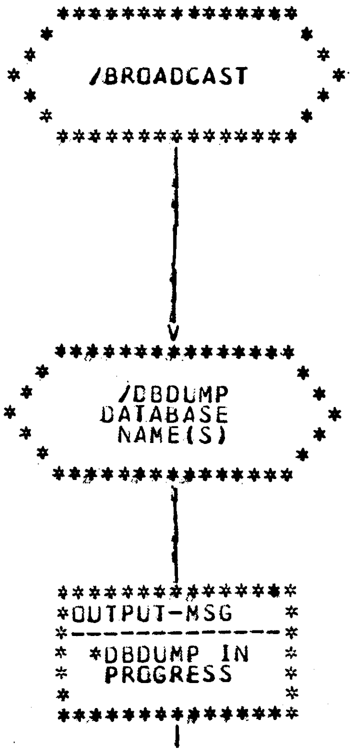
	SYSTEM CHECKPOINTS					
	SIMPLE			SHUTDOWN		
PRELIMINARY CONDITIONING	ACTIVITY COUNT	TERMINAL COMMAND	ABNORMAL END	PURGE	DUMPQ	FREEZE
STOP TERMINAL INPUT				1	1	1
STOP TERMINAL OUTPUT				6	2	2
SEND ALL OUTPUT				5		
PROCESS ALL QUEUED TRANSACTIONS				2		
FREE MESSAGE REGIONS	1	1		3	3	3
TERMINATE MESSAGE REGIONS				4	4	4
CHECKPOINT ACTION						
FORCE END OF VOL. LOG TAPE				1	1	1
DUMP QUEUES TO LOG TAPE				2	2	2
CLOSE QUEUES			5	3	3	2
CLOSE ALL DATA BASES			4	4	4	3
LOG BLOCKS, TABLES, STATUS	1	1	1	5	5	4
WRITE CHECKPOINT ID TO MASTER TERM.	2	2	6	6	6	5
WRITE CHECKPOINT ID TO SYSTEM CONSOLE			2	7	7	6
CLOSE LOG			3	8	8	7
RESUME NORMAL PROCESSING	3	3				
TERMINATE			7	9	9	8

Figure 18. Checkpoint sequence

Data Base Dump Execution

The data base dump capabilities of checkpoint include the functions of creating a dumped tape image of a complete data base and performing preparatory functions for the reconstruction of a data base.

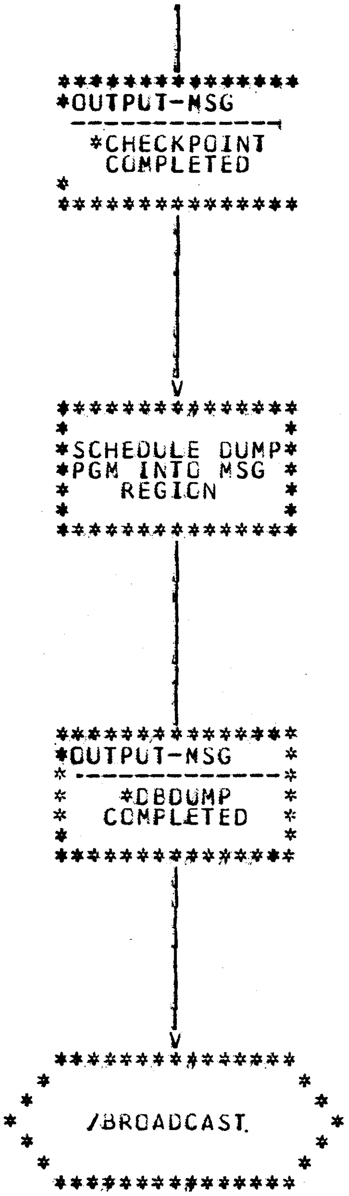
The following is a list of events that should be implemented to perform a data base dump:



When it is decided to do a data base dump from the master terminal, for either backup or reorganization, the appropriate users of remote terminals should be notified.

This identifies the data base(s) to be dumped and causes the preparation in the IMS/360 control program for dumping the data bases.

All input against the specified data base(s) is stopped (prohibited entry) during data base dump. All enqueued messages are processed. A simple checkpoint is taken. This message is output to the master terminal only.



The current IMS/360 system log is closed and a new one opened to provide a clean starting point in case of a later data base recovery. This message is output to the master terminal only.

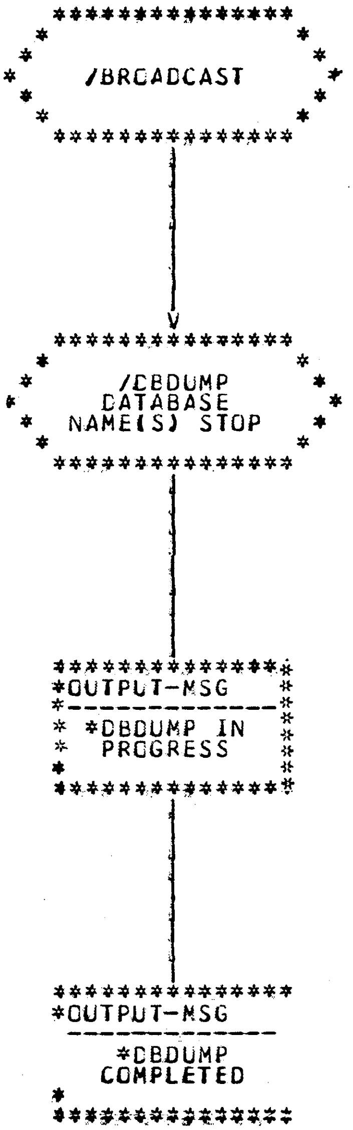
This dumps the data base to tape as an HSAM tape data base. A special utility message processing program which is part of IMS/360 is used for all data base dumps.

The transactions are started (allowed entry) and the master terminal is notified that the dump is complete and that normal operations may resume.

The appropriate users should be notified that normal operations have resumed.

Data Base Dump with Stop Execution

Data base dump with STOP is the second form of the /DBDUMP command and may be used whenever it is desired to stop the message processing activity against a data base. This command is used in preparation for data base recovery. The sequence of events for implementing data base dump with STOP follows:



The terminal users of the data base(s) should be notified of the impending action.

This identifies the data base(s) to be stopped.

All transactions which use the data base(s) are PSTOPed (that is, input messages may be entered and will be enqueued, but will not be processed). The data base(s) is also closed. A force end of volume is employed on the current log tape to allow its use in the /DBRECOVERY command.

This notifies the master terminal that all data base activity is stopped.

The data base to be rebuilt should now be reconstructed in a Type 3 region with a previously dumped copy. Then the /DBRECOVERY command and all log tapes from the last dumped copy should be used to reconstruct the data base to the current point in time.

Restart

The restart facilities of the IMS/360 system provide for the recovery after failure of IMS/360, its message queues, and the data bases used for message processing. This section concerns itself with the execution of all the restart facilities and their commands.

There are two restart commands with various operands, normal and emergency restart. The normal restart command is used after the IMS/360 system has been terminated in a normal manner (that is, with a /CHECKPOINT command). The emergency restart command is employed when the IMS/360 system was not terminated normally. There is an event listing of each command in this section.

The final capability of the restart facilities of IMS/360 is data base recovery. Data base recovery is used to rebuild or recreate a data base used for message processing. There is also an event listing of the data base recovery command in this section.

Normal Restart Format

The normal restart command has two basic versions. One version is a cold start and involves no previous system log tape. The other version is a warm start where the system is restarted with the checkpoint data on a previous (normally the last used) system log tape. The format of the cold start version is:

```
┌──────────────────────────────────────────────────────┐
│            │                                         │
│ /NRESTART  │   CHKPT 0                               │
│            │                                         │
└──────────────────────────────────────────────────────┘
```

Checkpoint number zero signifies a cold start. The format of the warm start version is:

```
┌──────────────────────────────────────────────────────┐
│            │                                         │
│ /NRESTART  │   CHKPT 68173/141020                    │
│            │                                         │
└──────────────────────────────────────────────────────┘
```

The checkpoint number specifies Julian date and time of day.

In addition to the checkpoint number operand, both versions of the normal restart command allow the master terminal operator to format the message queue data sets. The formatting of the message queue data sets need be done only at initial system start (first time use of system), when an input/output error occurs, or when the size of the message queue data sets is to be changed. The format for the normal restart command with formatting is:

```
┌──────────────────────────────────────────────────────┐
│            │                                         │
│ /NRESTART  │   CHKPT 0 [FORMAT ALL]                  │
│            │                                         │
│            │                                         │
│ - - - - -  │ - - - - - - - - - - - - - - - - - - - - │
│            │                                         │
│ /NRESTART  │   CHKPT 68165/141050 [BLDQ]             │
│            │                                         │
│            │   [SER number , number  ]               │
│            │              1         2                │
│            │                                         │
└──────────────────────────────────────────── ─────────┘
```

The FORMAT ALL operand causes all message queue data sets to be formatted.

An additional operand, BLDQ (build queue), may be specified with the warm start version of the normal restart command. The BLDQ operand should be specified if the system was terminated with a /CHECKPOINT PURGE or /CHECKPOINT DUMPQ command. The BLDQ operand assumes that any messages remaining in the message queue data sets when the /CHECKPOINT PURGE or DUMPQ terminated were logged to the system log tape. The BLDQ operand causes the normal restart command to use the old log tape specified in the CHKPT operand and to reload, from the log to the message queue data sets, any retained messages. The format of the normal restart command is:

```
/NRESTART    CHKPT 68143/11300 [BLDQ FORMAT ALL]

             [SER number , number  ]
                         1        2
```

The warm start version of the /NRESTART command without BLDQ assumes that the IMS/360 system was terminated with a /CHECKPOINT FREEZE command. All messages are retained in the message queue data sets. When the /NRESTART command (with warm start) is executed, the data on the old system log tape provides the IMS/360 system with correct positioning within the data set.

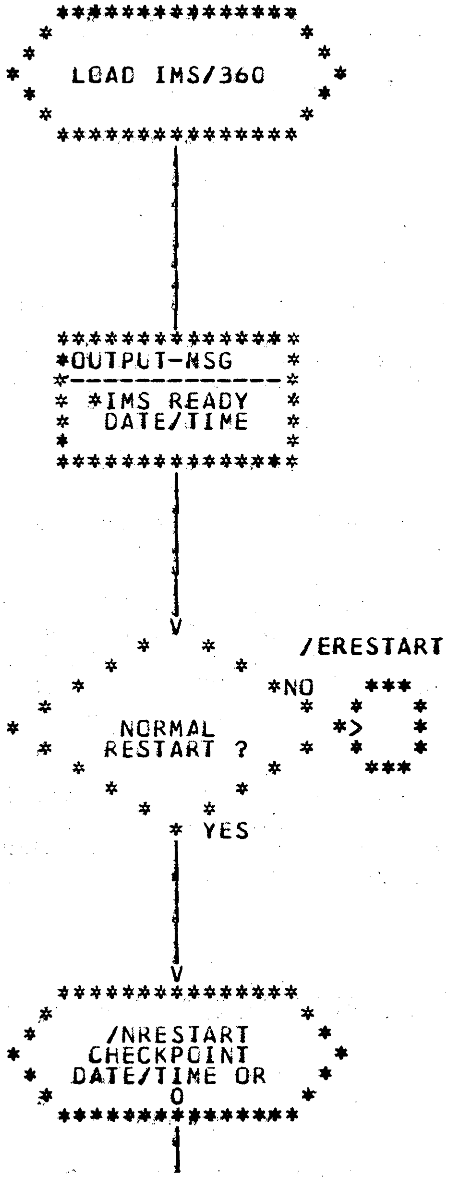
The security information (password and terminal) employed by the IMS/360 user can be built from the output of the last security maintenance program execution located in IMS.RESLIB or from the checkpoint data used to restart. If the information in IMS.RESLIB is desired, the normal restart command should include the TERMINAL and PASSWORD operands. If the security information from the last checkpoint is desired, the normal restart command should omit the TERMINAL and PASSWORD operands.

When a warm start is performed, the IMS/360 user may specify the log tape serial number for the old log on a DD card or through an operand of the normal restart command (that is, SER volume). The use of the SER operand on a restart command facilitates the use of a cataloged procedure for the IMS/360 control program.

The IMS/360 Operations Manual, Volume II - Machine Operations illustrates the use of TERMINAL, PASSWORD, and SER operands for restart.

Normal Restart Execution

The following is a list of events that should be followed in order to cause a normal restart of IMS/360:

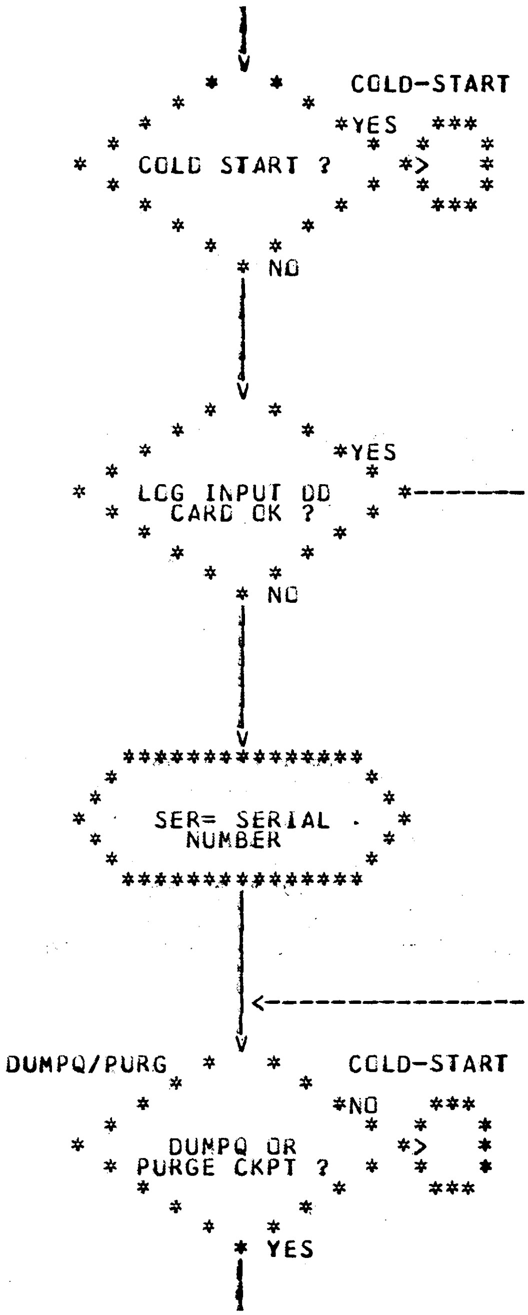


Load IMS/360 as with any Operating System/360 job.

This message goes to the system console and to the master terminal. At this point, the operator must enter the restart command. No other command is acceptable.

Was the last IMS/360 job terminated normally with a checkpoint or will a cold start be done? If the answer is no, go to the label /ERESTART.

Begin to enter the restart command. yyddd/hhmmss is the checkpoint identification. If cold start, enter 0 instead of yyddd/hhmmss. No EOB yet.



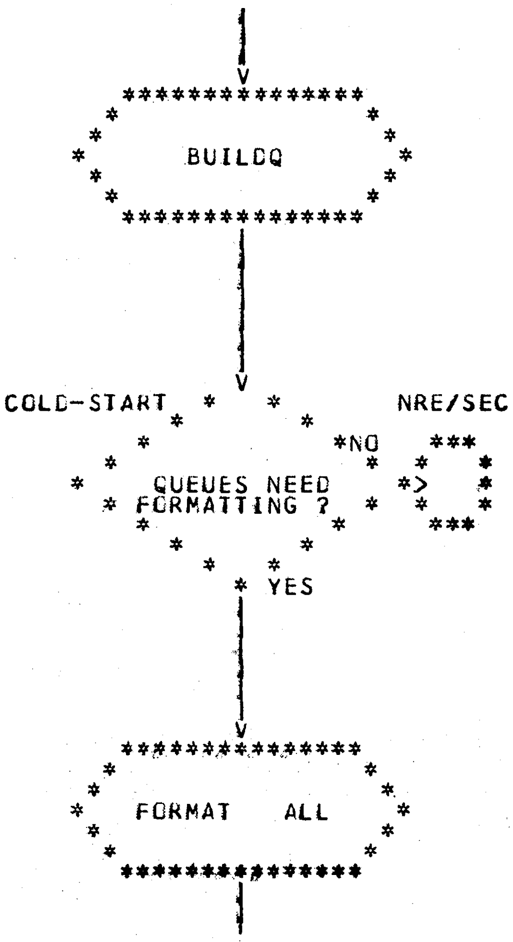
If cold start, go to the label COLD-START.

Does the input log DD card specify the proper tape volume?

Add SER parameter to restart command and specify tape serial number. It should be the serial number from the appropriate checkpoint message.

How was the system last terminated? Was the checkpoint specified a DUMPQ or a PURGE?

Go to next page

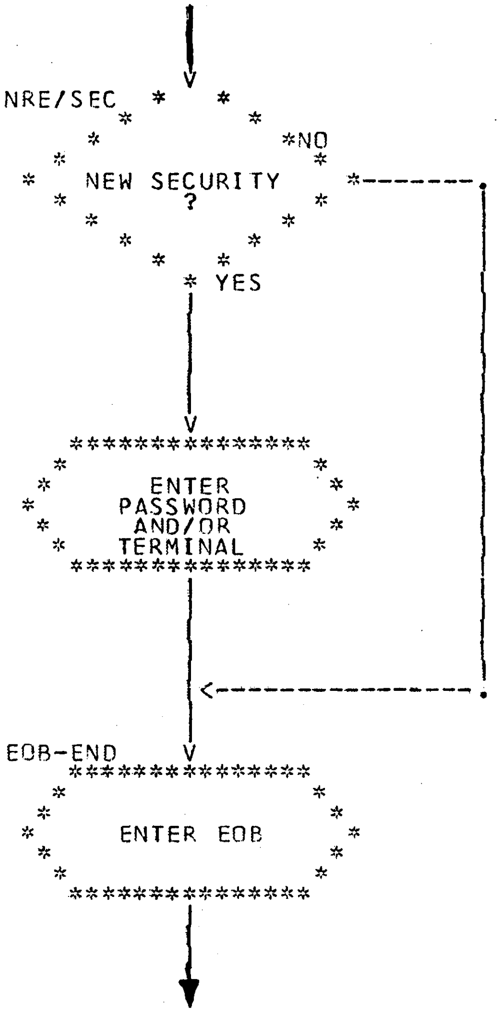


Add BUILDQ parameter to restart command. This will reconstruct the disk message queues from the specified checkpoint.

Is disk message queue allocation new? Has their space been used by someone else? If warm start, do not format unless BUILDQ was specified.

Add FORMAT ALL to restart command. This will preformat the disk message queues. (This is similar to Operating System/360 cold start formatting the JOBQ.) About ten seconds per 2314 cylinder is required for formatting each message queue data set.

Go to next page

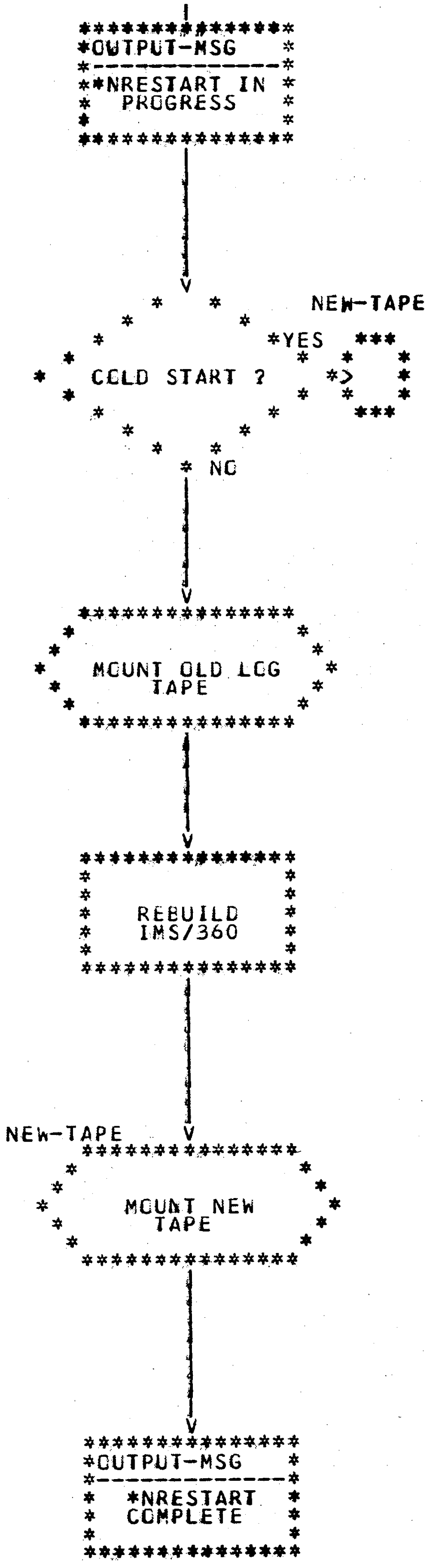


Has terminal or password security been changed? If the output from the last execution of the security maintenance program is different from the terminal or password security on the log from the checkpoint that terminated the system, which is desired? (They can be different because of commands entered from the master terminal.)

Add security parameter(s) to restart command. Password parameter causes new password security to be loaded. Terminal parameter causes new terminal security to be loaded. The absence of the SECURITY parameter indicates that the conditions on the log tape from the last checkpoint shutdown will be used. If present, the tables output from the last execution of the system security maintenance program are used.

End of normal restart command.

Go to next page



Notification to master terminal that restart is in progress.

If cold start, skip rebuild of IMS/360.

Mount log tape containing specified checkpoint.

Rebuild IMS/360 blocks, pointers, and queues (if specified) to status of checkpoint.

Mount log tape to be written by this run of IMS/360.

Complete restart and notify master terminal. All commands except restart are now acceptable to IMS/360.

Emergency Restart

The emergency restart command is used to restart IMS/360 after a failure which caused the IMS/360 control program region job or Operating System/360 to terminate abnormally. The emergency restart command always employs the last IMS/360 log tape to reinitiate system operation if only the contents of core storage are lost. The simplest version of the emergency restart command is used when a failure occurs that involves only the loss of core storage contents. The format of this version of the emergency restart command is:

```
┌─────────────────────────────────────────────────────┐
│            │                                        │
│ /ERESTART  │    CHKPT 68176/105010 [SER  TAPE50]    │
│            │                                        │
└─────────────────────────────────────────────────────┘
```

The checkpoint number to be used is the last checkpoint executed prior to loss of the system. This would have been recorded on the master terminal as:

```
*CHECKPOINT COMMAND COMPLETE *68176/105010* SIMPLE *TAPE50
```

where *SIMPLE indicates simple checkpoint, and *TAPE50 indicates that the volume serial of the system log tape was TAPE50.

The failure of the IMS/360 or Operating System/360 system may have included a failure of the message queue data sets. In this situation, the emergency restart command with FORMAT ALL and BLDQ operands should be employed. The format of this version of the emergency restart command is:

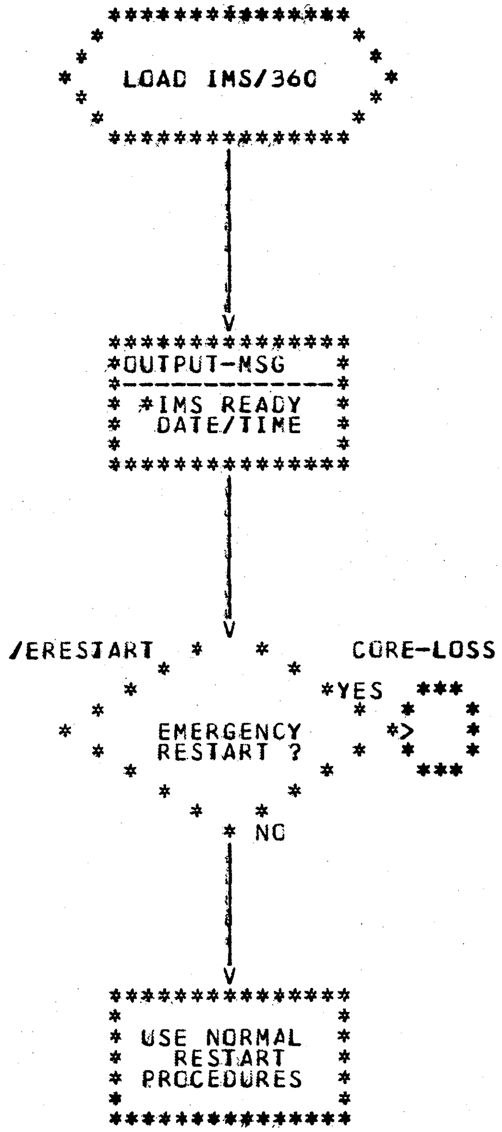
```
┌─────────────────────────────────────────────────────┐
│            │                                        │
│ /ERESTART  │    CHKPT 68141/091050 [BLDQ FORMAT ALL]│
│            │                                        │
│            │    [SER number ,...number  ]           │
│            │                1          2            │
│            │                                        │
└─────────────────────────────────────────────────────┘
```

This command causes all the message queue data sets to be formatted and all messages yet to be processed or transmitted to be reloaded from old system log tapes to the proper message queue data set. Emergency restart with BLDQ and FORMAT ALL operands requires that the IMS/360 system be restarted from the last cold start or last system termination in which the message queue data sets were dumped (that is, /CHECKPOINT PURGE or /CHECKPOINT DUMPQ). If the emergency restart is performed from a previous cold start, the checkpoint number must be 0.

Either version of the emergency restart command attempts to reestablish the IMS/360 system as of the time of failure. The message queue data sets are repositioned for each input message type and each output logical terminal. If the message(s) being processed at the time of system failure caused modification of message processing data bases, an additional function of emergency restart causes "backout" of any partial data base modifications. This is an optional feature of emergency restart that is controlled at the data base level. It involves the logging of all data base modifications during normal system operation for those data bases for which backout is specified. The IMS/360 user specifies which data bases are to employ backout in the DATABASE macro system definition.

Emergency Restart Execution

The following is a list of events that should be followed in order to cause an emergency restart:

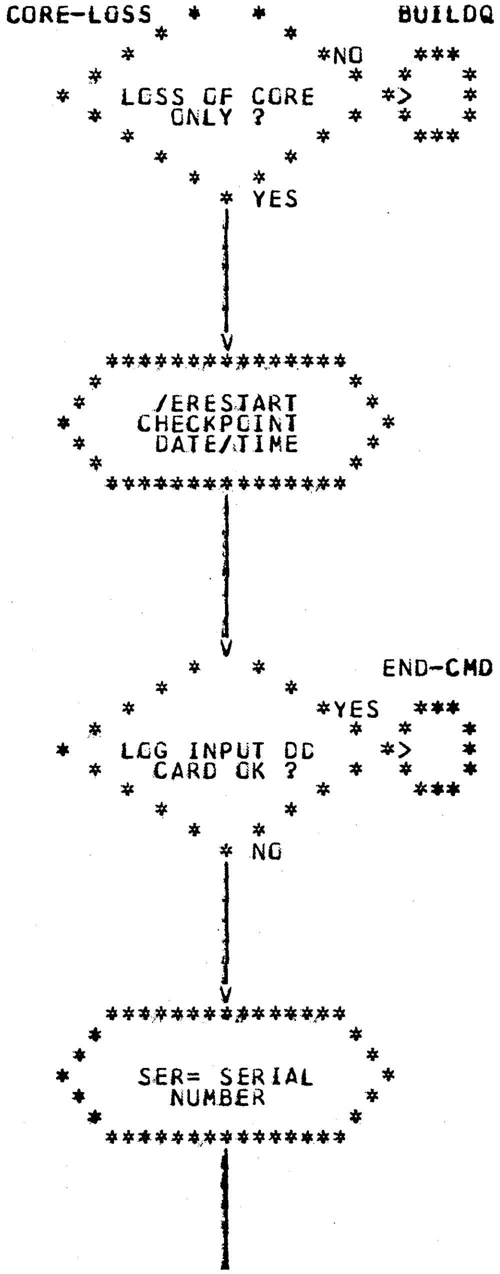


Load IMS/360 as with any Operating System/360 job.

This message goes to the system console and to the master terminal. At this point, the operator must enter the restart command. No other command is acceptable.

Is this an emergency restart? If no, enter normal restart.

Return to normal restart procedure.



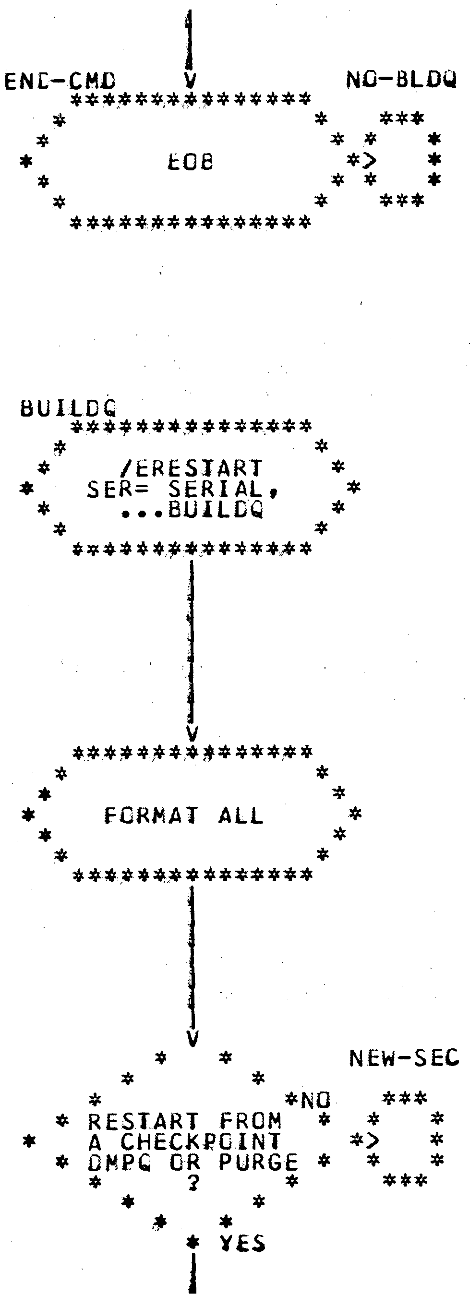
Did previous ABEND result in core loss only? Are disk message queues intact?

The checkpoint identification will be the last checkpoint number printed on the master terminal.

Does the input DD card specify the proper volume?

The serial number should be that printed in the checkpoint message.

Go to next page



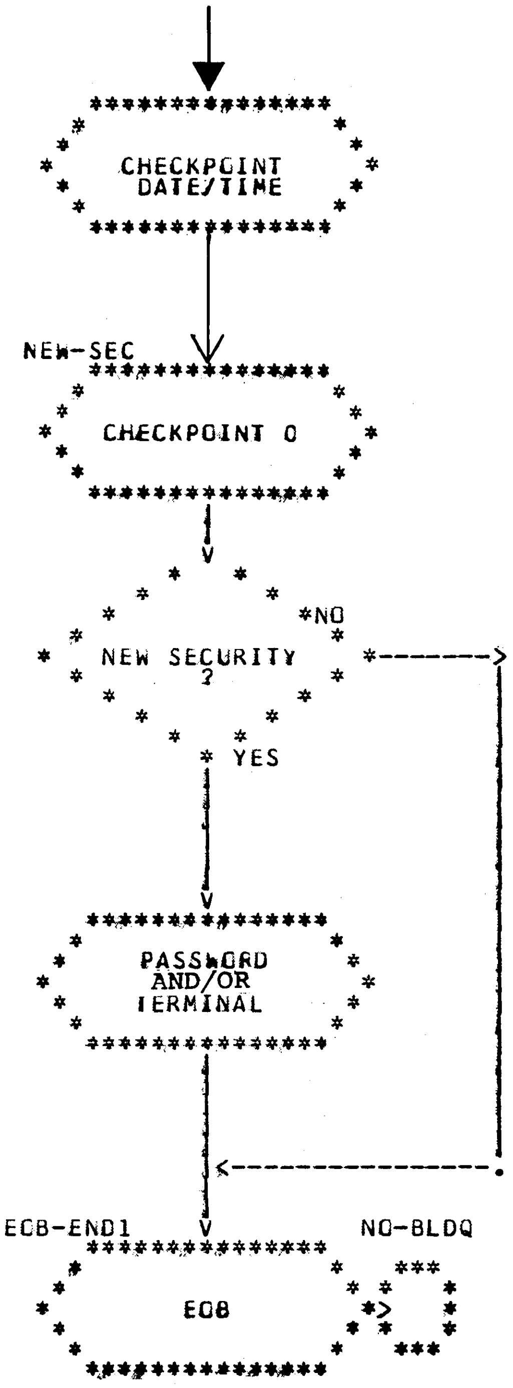
End of restart command.

Emergency restart command with BUILDQ. The tape serial numbers must be in chronological order beginning with the tape used at the last IMS/360 cold start or the one used at the last checkpoint PURGE or DUMPQ. If the message queues are destroyed, restart cannot be initiated from a simple checkpoint or a checkpoint FREEZE. The last tape to be used is the one mounted at the time of the ABEND.

This parameter is required for emergency restart with BUILDQ.

This restart must begin at the point when the message queues are known to be good. This point will be the last checkpoint DUMPQ or PURGE or a cold start.

Go to next page



Enter checkpoint identification.

Enter CHECKPOINT 0.

Did the cold start used in this restart specify either of the security parameters?

Use the same parameter as used on the cold start.

End of restart command.

```
NO-BLDQ
     ***************
     *OUTPUT-MSG    *
     *--------------*
     **ERESTART IN  *
     *   PROGRESS   *
     *              *
     ***************
            |
            |
            |
            |
            |
            V
     ***************
   *                 *
  *                   *
 *    MOUNT NEW LOG    *
  *       TAPE        *
   *                 *
     ***************
            |
            |
            |
            |<---------------.
            |                |
OLD-LOG     V                |
     ***************         |
   *                 *       |
  *                   *      |
 *    MOUNT OLD LOG    *     |
  *       TAPES       *      |
   *                 *       |
     ***************         |
            |                |
            |                |
            |                |
            |                |
            V                |
     ***************         |
     *              *        |
     *              *        |
     *    REBUILD   *        |
     *    IMS/360   *        |
     *              *        |
     ***************         |
            |                |
            |                |
            |                |
            |                |
            V                |
         *     *             |
      *           *          |
   *                 *YES    |
 *                     *     |
*      MORE LOG         *------.
 *     TAPES ?         *
   *                 *
      *           *
         *     *
            * NO
            |
```

Go to next page

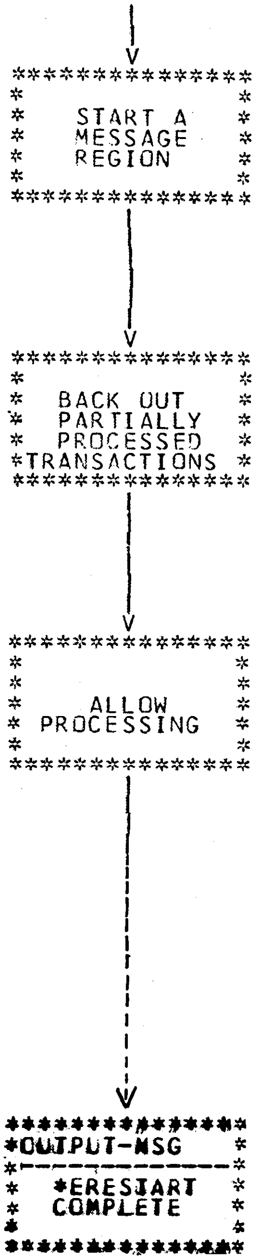
Notification to master terminal that emergency restart is in progress.

Mount log tape to be written by this run of IMS/360.

If more than one log tape, they must be mounted in the order specified.

Using the information on the log tapes, IMS/360 is restored to the point of ABEND.

Process all log tapes.



The message region is started directly by IMS/360 and will be used in reprocessing partially completed transactions.

Any unpredictable results from transactions in process at the time of the ABEND are "backed out" to restore the integrity of the data bases.

The partially processed transactions are now rescheduled and allowed to process normally.

Notification is made to the master terminal that restart is completed and processing may resume.

Normal or Emergency Restart in Minimum System

Where insufficient core storage is available to start the message or online batch partition following initiation of the Type 0 control region, the sequence of initiation may be reversed. If message or batch regions are initiated prior to the Type 0 region, the operator will be given the option to wait for the Type 0 region. (See "Messages and Completion Codes", message IMS050D, in the IMS/360 Operations Manual, Volume II - Machine Operations for a further explanation.) Obviously, the user must initiate the IMS/360 Type 1 (message) or Type 2 (batch) region with a Job Control Language deck from an Operating System/360 SYSIN stream rather than a /START REGION command.

Data Base Backout

When an emergency restart is necessary, IMS/360 has failed for some reason. When the failure occurred, there may have been one or more Type

1 and Type 2 processing regions operative. In addition, these regions may have been executing an application program which was deleting, updating, or adding to a data base used for message processing. An additional function of emergency restart is to "insert" the update, delete, or add operations being performed by the application programs at system failure. This insertion attempts to place the data base back into the state that existed prior to scheduling the application program into the Type 1 or Type 2 region. This function of data base backout is provided on an optional basis by data base because it involves writing all modifications of the data bases that use the feature to the IMS/360 system log. During emergency restart, any changes made to logged data bases by all programs actually in progress at the time of failure are removed. This restores the data bases and the input message queues to the status they had when the program was scheduled. Processing then resumes at that point rather than at the point of failure.

Since IMS/360 cannot schedule programs in the IMS/360 Type 2 batch environment, IMS/360 cannot control during emergency restart the integrity of any data bases that may be updated in this environment. For this reason the use of Type 2 batch programs for updating purposes should be discouraged. A possible method for large volume updating is:

1. The Type 2 batch program is used only as an editing procedure that reads the input data, formats it, and routes it as an output message to an SMB rather than to a terminal.

2. As these "messages" are enqueued on the SMB, a message processing program is scheduled to perform the actual updates.

3. The output can be directly to terminals, to an HSAM file to be printed by another Type 2 batch job, or to a zero priority SMB that can be referenced by another Type 2 batch job.

4. To effectively use the data base backout facility of emergency restart, an audit trail is also needed to allow processing of input data to resume at the point of failure.

Data Base Recovery

The final capability of the restart facilities of IMS/360 is data base recovery. Data base recovery is used to rebuild or recreate a data base used for message processing. The concept involves the periodic dumping of each data base, using the /DBDUMP command. This command is part of the checkpoint facilities of the IMS/360 system. The /DBDUMP command causes a copy of a data base to be created as an HSAM tape data base. When a data base must be recreated, the /DBDUMP command with the STOP operand, the /DBRECOVERY command, and all system log tapes since the /DBDUMP are employed.

First the data base to be recreated must be restored as the last dumped copy. This is accomplished by:

1. Issuing a /DBDUMP command with STOP operand to halt all processing against the data base.

2. Restoring the data base to its state of the last dumped copy. A batch program in a Type 3 processing region is employed.

Then, from the master terminal, a /DBRECOVERY command is issued specifying the data base names and the volume serial numbers of the log tapes to be used in reconstruction. The format of the /DBRECOVERY command is:

```
/DBRECOVERY    DATABASE name SERIAL number,,numbers

                        [ { TAPE   } ]
                        [ { RESEND } ]
```

The DATABASE operand may have multiple names to allow multiple data base reconstruction. The SERIAL operand specifies the volume serial numbers of the log tapes to be employed.

The output from the reprocessing of messages during the recovery procedure can be handled in one of three ways:

1. If the TAPE parameter is added to the /DBRECOVERY command, all reprocessed output will be logged on the system log; it will not be resent to the terminals.

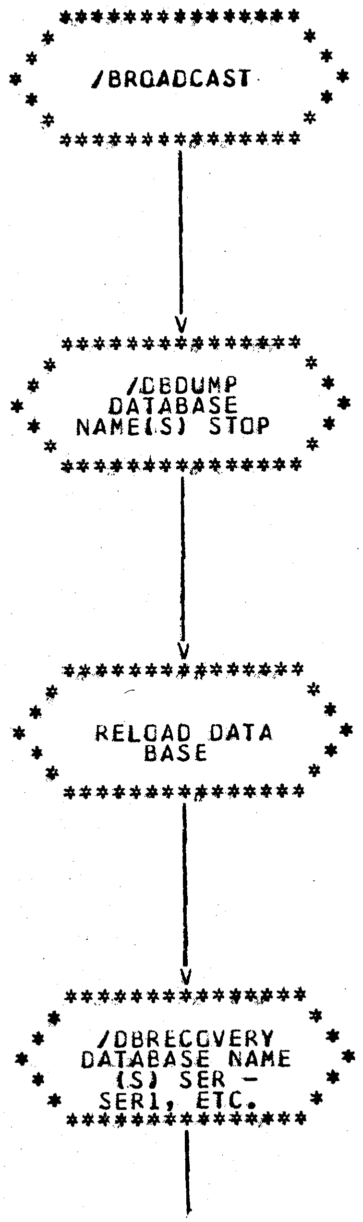
2. If the RESEND parameter is added to the /DBRECOVERY command, the output is resent to the terminals.

3. If neither TAPE nor RESEND is specified, all reprocessed output is ignored.

The old log tapes are used to reprocess all transactions against the data base since the last dump. The DD card used for the tape volumes during data base recovery is supplied in the IMS0 procedure. The DD name is IMSLOGR.

Data Base Recovery Execution

The following is a list of events that should be implemented to attempt a data base recovery:


/BROADCAST
/DBDUMP DATABASE NAME(S) STOP
RELOAD DATA BASE
/DBRECOVERY DATABASE NAME (S) SER - SER1, ETC.

If a data base(s) becomes unusable and it is decided to attempt a recovery, all terminal users of the data base(s) should be notified that the data base is unavailable while it is being reconstructed.

This allows input messages to be accepted and placed in the IMS/360 message queues, but closes the data base itself to allow it to be reloaded. The input messages are not processed. A "Force End of Volume" is also issued to the system log tape.

The latest backup copy of the data base should be used to reload it. This is done in an IMS/360 Type 3 processing region (batch) environment.

This specifies the data base(s) to be recovered and the log tapes needed. The serial numbers must be in chronological sequence. The sequence starts with the first one after the backup copy was created with the /DBDUMP command. Those log tapes that are used to restore the data base(s) include all those from the dump to the log tape mounted when this command is given.

Go to next page

```
              |
              |
              V
       ***************
       *OUTPUT-MSG    *
       *--------------*
       * *DBRECOVERY  *
       * IN PROGRESS  *
       *              *
       ***************
              |
```

All transactions affecting the data base(s) are stopped to further input, and a simple checkpoint is taken. This message is to the master terminal only.

```
              |
              V
       ***************
       *OUTPUT-MSG    *
       *--------------*
       * *CHECKPOINT  *
       *COMMAND COMP  *
       *              *
       ***************
              |
```

At this point, the current log tape is closed so that it can be used as input to the recovery program. A new log tape will be opened. This message is to the master terminal only.

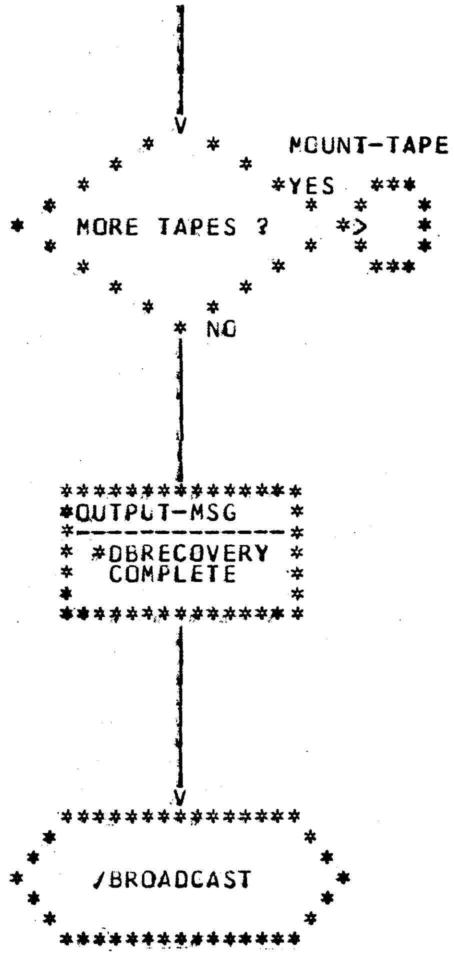
```
MOUNT-TAPE    V
       ***************
     *                 *
    *                   *
   *     MOUNT INPUT     *
    *     LOG TAPES     *
     *                 *
       ***************
              |
```

The input log tapes will be requested and processed in the order specified in the /DBRECOVERY command.

```
              |
              V
       ***************
       *              *
       *              *
       *  REPROCESS   *
       *TRANSACTIONS  *
       *              *
       ***************
              |
```

The input messages will be processed from the log tape as if they were from the terminals.

Go to next page



Process all tapes listed in the command. (This is a system decision that will notify the computer console.)

Start all transactions and notify the master terminal that normal operations may resume.

Notify all users that normal operations may resume.

Note that the /DBRECOVERY command is a single-line command. If there are too many tapes for one line, the command must be reentered for the extra tapes after the first one is completed. If the serial number of the current log tape is not known, issue a /CHECKPOINT command. The checkpoint-completed message will contain the desired serial number.

SYSTEMS OPERATION INTERFACE WITH OTHER FUNCTIONS

The main interfaces to the functional portions of IMS/360 are delineated in Chapter 1 of this manual. However, a few other planning items must be considered.

Interface with Machine Operations

1. Monitor to see that necessary manual logs are maintained. The following manual logs may be required and/or desirable.

   a. Log of all checkpoints taken, by checkpoint type

   b. Log for each data base when it was dumped, to what tapes, and what IMS/360 log tapes are required if reconstruction is necessary

   c. Log of all remote terminals (names, location, telephone number connections)

   d. Log of resources - stopped, PSTOPed, purged, started, etc.

   e. Log of all restarts taken, by restart type

2. Coordinate to the satisfication of the Systems Operation function that normal and emergency master terminal operator procedures are complete.

3. Monitor to see that instructions for remote terminal trouble diagnosis and reporting are complete.

4. Supply machine operators with adequate instructions for monitoring data base overflow records.

5. Supply machine operators with the necessary information for controlling and protecting the libraries of PSB's and DBD's. (Can use expiration date protection.)

6. Coordinate and verify procedures for normal, scheduled batch processing of the system log for accounting data (and statistics) and interface with users' billing systems.

7. Coordinate and verify the training of master terminal operators and remote terminal operators.

Interface with Management

Because the Systems Operation function is the "hub" of the system (see Figure 1), another important interface is with Management. IMS/360 provides statistical reports and accounting information that can be condensed for management analysis. Systems Operation planning should provide Management with information on the need for additional equipment, and applications to be added to the system, along with backup of current and historical data. A weekly and/or monthly report should be devised that condenses the information with which Management is concerned.

System definition is the means by which a user of IMS/360 structures IMS/360 to the data processing environment. This structuring includes a definition of communication line groups, lines, physical terminals, logical terminals, pools, and subpools. It also includes the transactions, application programs, data bases, and various Operating System/360 interfaces. Security maintenance is the means by which a user defines the terminal and password security characteristics of a defined IMS/360 system.

IMS/360 SYSTEM STRUCTURING CONSIDERATIONS

Before structuring the IMS/360 system, the user of IMS/360 must consider the requirements and capabilities of IMS/360 in relation to his own requirements for the most expeditious operating environment. Consideration must be given to such things as the amount of core and direct access storage to be dedicated to IMS/360, the number of application programs to be run, and how many of these programs are to be run concurrently. A determination must be made of the transaction codes which are going to initiate the various application message processing programs and how many of these types of transaction codes are necessary. The user must decide which transaction codes are to be of the response type and which of the nonresponse type. Decisions must be made concerning how many transaction codes cause data base updates, and how many are restricted to entry from a particular terminal. Consideration must also be given to how many lines and terminals there will be in the system (Items 1a, 1e, 4, 5, 7a, 7b, 7c(1), and 7f in the Systems Operation checklist).

In supplying answers to these questions, the user should consider the possible impact of his decisions on the operating capability of the system and the efficiency of its operation.

IMS/360 allows the user to batch online transactions. The user would be wise to consider whether the types of codes he chooses can be queued up and can wait for processing on an as-required basis. Time accounting is an example of the type which may fall into this category. Attendance reporting is another. Transaction codes of these types can be readily batched, because there is no necessity for an immediate-type response.

("Response-type" and "nonresponse-type" messages should not be confused with true "message types". See Chapter 5 of the Program Description Manual under "Message Formats and Structures".)

Whenever he enters a response-type message, the user should always be aware that his terminal locks and he must wait for a response before he can enter another message from that terminal. The nonresponse-type message is entered and competes with other messages, on a priority basis, for system resources, but the terminal and communication line are always available for further message input until response. Note, too, that a design consideration was that response-type messages be single-line nonupdate messages. If incore buffer space is miscalculated, system efficiency can be reduced by allowing response-type messages to be multisegment messages.

The limit count feature of IMS/360 allows consideration of the number of messages which a reusable application program can process in one load of the program. Whether the messages are of the response type is of vital concern. The limit count feature, in conjunction with the limit priority, does not say that a program will never be processed if there are always higher priority messages. It does say that, if the particular message is not called for execution by the time a certain number of messages have been received and queued, the selection priority is changed to a higher one. If there are messages with higher selection priorities in the system, of course, this message may still have to wait.

The total IMS/360 system must be considered by the user when structuring his system. The user must consider what the various types of transactions mean to the system, what the responses are, how many there are, etc.

Again, the user must consider the number of programs he wishes to be operating concurrently, how large these programs are and how many transaction codes they are operating against, and how many terminals he will be using. These considerations affect the amount of core which is dedicated to IMS/360. Each application program, and the system, at any given time, may require additional amounts of space. Even the number of terminals concurrently being transmitted to has an effect upon the amount of core buffer space which should be allocated.

For example, assume a message is entered from a terminal. The application program for processing this message may send messages to each of six different terminals; therefore, IMS/360 may require core buffer space for one line of the message output to each terminal. If the system is executing three application programs concurrently and trying to transmit to those six terminals, it will require allocation for 3 (number of application programs) times 6 (number of terminals) buffers in addition to the core required to hold the three application programs (in message processing regions).

The I/O units on which a system user chooses to place his message queues have special significance on system operations. For example, the choice of disk or drum affects the number of messages run and consequently how many are processed. Since drum access is faster than disk, its use allows a greater number of messages to come in and go out of the system faster than when disk storage is used. Of course, there is more storage available on disk, but this is part of the tradeoff analysis to be made while structuring the system.

Defining the IMS/360 System

So far this discussion has centered on what the user wants the IMS/360 system to do. Now to be considered are what the system user is trying to do and how the IMS/360 system is tailored to his needs. This tailoring is done with the IMS/360 system definition macro-instructions.

The IMS/360 requirements are described above. The modifications necessary to make IMS/360 compatible with Operating System/360 are accomplished through the use of three supervisor calls (SVC's), which must be made a part of Operating System/360. This is a simple matter for the system programmer to accomplish.

The System/360 used for the IMS/360 two-stage definition process must be at least a Model 40, with the F-level assembler and at least 128K storage. The IMS/360 system definition must be run using the same version of Operating System/360 under which the generated system will execute. The three SVC's must be placed by the system user into (link-edited with) the Operating System nucleus of the system under which IMS/360 execution is to occur. The choice of cataloging IMS/360 system data sets is the user's, but this normally simplifies system execution and control.

If Stage 1 was not properly defined, Stage 2 input can be corrected without the necessity for complete regeneration. A system programmer knowledgeable in IMS/360 control block structure can accomplish this function.

IMS/360 System Definition Macro-Instructions

The input to Stage 1 of the IMS/360 system definition is a set of control cards which invoke macro-instructions. These macro-instructions tailor IMS/360 to a particular user's environment by creating the control blocks upon which the IMS/360 modules execute. Two types of system definition are possible:

1. Complete online and Type 3 batch region system

2. Type 3 processing (batch stand-alone) only

Some IMS/360 system definition macro-instructions appear only once in the Stage 1 input stream, while others may be used multiple times in a hierarchical set arrangement to describe related user requirements. Figure 19 lists which macro-instructions are required and which may be used more than once. The end of this chapter provides examples of IMS/360 system definition.

	MACRO- INSTRUCTION	TYPE OF DEFINITION	
		COMPLETE SYSTEM	BATCH TYPE 3
1	IMSCTRL	REQUIRED 1	REQUIRED 1
2	APPLCTN	REQUIRED n	N/A*
3	DATABASE	REQUIRED n	N/A
4	TRANSACT	REQUIRED n	N/A
5	LINEGRP	REQUIRED n	N/A
6	LINE	REQUIRED n	N/A
7	TERMINAL	REQUIRED n	N/A
8	NAME	REQUIRED n	N/A
9	POOL	OPTIONAL 1	N/A
10	SUBPOOL	OPTIONAL 1	N/A
11	MASTTERM	REQUIRED 1	N/A
12	MSGQUEUE	REQUIRED 1	N/A
13	MACLIB	OPTIONAL 1	OPTIONAL 1
14	RESLIB	OPTIONAL 1	OPTIONAL 1
15	PGMLIB	OPTIONAL 1	OPTIONAL 1
16	PSBLIB	OPTIONAL 1	OPTIONAL 1
17	DBDLIB	OPTIONAL 1	OPTIONAL 1
18	PROCLIB	OPTIONAL 1	OPTIONAL 1
19	IMSGEN	REQUIRED 1	REQUIRED 1

* N/A - Not allowable

Figure 19. Complete IMS/360 system definition macro-instruction

Two groups of macro-instructions form hierarchical sets that are required for the description of user resources. One group (Figure 20) describes application programs and their related resources (transactions and data bases). The other (Figure 21) describes communications line groups, communication lines, and associated physical and logical terminals.

Note: All macro-instruction positional or keyword operand values that are names must start with an alphabetic character.

MACRO-INSTRUCTION	NUMBER PER SET	PURPOSE
APPLCTN	1	Names application program. Delimits this set of macro-instructions.
DATABASE	n	Names data bases used by application program.
TRANSACT	n	Names transaction codes which will be processed by the above application program.

Figure 20. Application description macro-instruction set

MACRO-INSTRUCTION	NUMBER PER SET	PURPOSE
LINEGRP	1	Names collection of terminals with like attributes. Delimits this set of macro-instructions.
LINE	n	Provides address of line and delimits terminals on same line.
TERMINAL	n	Provides physical terminal data and delimits logical terminal name.
NAME	n	Provides logical terminal names.

Figure 21. Terminal description macro-instruction set

IMSCTRL Macro

The IMSCTRL macro-instruction is used to describe the basic IMS/360 control program options and the Operating System/360 environment under which IMS/360 will operate. The IMSCTRL macro-instruction is always required. For Type 3 batch definition, the MAXIO, MSGBUFF, MAXREGN, and COMMSVC operands need not be specified.

For Type 1 and 2 processing regions:

```
|              | IMSCTRL  |  SYSTEM = [({MVT   }     ,ALL)],
|              |          |            [({MFT-II}         )]
|              |          |
|              |          |  MAXIO = number,
|              |          |  MAXREGN = number,
|              |          |  COMMSVC =(number¹, number²),
|              |          |
|              |          |  OCENDA = appendage suffix,
|              |          |  OSAMSVC = number,
|              |          |  MSGBUFF = number,
|              |          |  CKPT     =(1000   ,NO )
|              |          |            (number  YES)
```

For Type 3 processing region:

```
|              | IMSCTRL  |  SYSTEM = [({MVT   }       )]
|              |          |            [({MFT-II}, BATCH)] ,
|              |          |            [({PCP   }       )]
|              |          |
|              |          |  OCENDA = appendage suffix,
|              |          |
|              |          |  OSAMSVC = number
```

Note: The only other macro-instructions needed for stand-alone batch are the xxxLIB macro-instructions.

Operand field:

SYSTEM=

specifies whether IMS/360 operates in an Operating System/360 environment with a variable number of tasks (MVT) or a fixed number of tasks (MFT-II). The default value for this keyword is MVT. If PCP is specified, all other operands of the IMSCTRL macro-instruction must be omitted except OCENDA and OSAMSVC. ALL means that IMS/360 teleprocessing and stand-alone Data Language/I batch systems are generated. BATCH means that only a stand-alone Data Language/I batch system is generated.

MAXIO=

specifies the maximum number of terminal I/O requests, message queue requests, and Data Language/I data base requests which may be in process in the IMS/360 control program region at any one time. A recommended minimum number is two times the value specified in the MAXREGN parameter plus the number of queue data sets specified. The value should never be less than the value specified in the MAXREGN parameter. If no value is specified, IMS/360 definition will provide an optimized value based upon peak system activity at 50% of possible requests in process at any one time.

MAXREGN=

specifies the maximum number of regions or partitions which IMS/360 is expected to support at any one time. This value includes Type 2 batch regions as well as Type 1 message processing regions. Default value is 2.

COMMSVC=

specifies the numbers for the Operating System/360 Type 1 SVC's which IMS/360 uses for interregion communication. The first number is for calls to the IMS/360 control program from other regions; the second is for replies from the IMS/360 control program. Default values are 253 and 254.

OCENDA=

specifies the load module member name to be given the OSAM channel end appendage. This module is placed into the IMS/360 load module library during IMS/360 system definition. The name of this module must start with IGG019. Two additional characters must be appended. These characters may range from WA to Z9. Only the last two characters of the name should be specified in the macro operand. The default name is OCENDA=Z9 (that is, IGG019Z9).

OSAMSVC=

specifies the user SVC number to be given the OSAM Type 2 SVC. This SVC is used to construct and extend OSAM data extent blocks (DEB) in system queue space when using MVT. The default SVC number is 255.

MSGBUFF=

specifies the number of incore message buffers for multiple line messages. See the section "IMS/360 Message Queues" in Chapter 3, for a further definition.

CKPT=

specifies a threshold value for the number of log records written. Upon reaching the threshold, IMS/360 generates an internal request for a simple checkpoint. Default is 1000. Range is 500 to 32,767. The second parameter (NO or YES) defines whether or not logging of replaced data base records is to be performed.

APPLCTN Macro

The APPLCTN macro-instruction describes the program resource requirements for application programs which run under the control of the IMS/360 Type 0 region. When combined with one or more DATABASE and TRANSACT macro-instructions, the set defines the total scheduling and resource requirements for an application program. The APPLCTN macro-instruction describes only programs which operate in Type 1 message processing regions or Type 2 batch processing regions. Application programs which operate in a Type 3 batch processing region are not to be described through the APPLCTN macro-instruction.

```
┌──────────────────────────────────────────────────────────────┐
│               │ APPLCTN │        PSB = psbname,              │
│               │         │                                    │
│               │         │               / TP  \              │
│               │         │  PGMTYPE = (( -----  } [,OVLY]))   │
│               │         │               \BATCH/              │
│               │         │                                    │
└──────────────────────────────────────────────────────────────┘
```

If a TP application program outputs messages which are input to another TP application program and the second program intends to modify a given data base, the first program must declare at least shared usage of the same data base for proper execution of the /DBRECOVERY command. This is performed with a DATABASE macro-instruction.

7

Operand field: 6

PSB=

specifies the logical name of the program specification block (PSB) as generated using the IMS/360 PSB generation utility. At execution time, the PSB must exist as a load module member of the partitioned data set named in the PSBLIB macro-instruction. The application program must also exist as a load module under the same member name in the partitioned data set named in the PROGLIB macro-instruction.

PGMTYPE=

TP identifies a message processing program which executes in a Type 1 region as a teleprocessing program. A BATCH program may utilize Data Language/I in the IMS/360 control region and may reference the message queues. If BATCH is coded, all TRANSACT macro-statements which follow will be assigned a normal and limit priority value of zero. The OVLY value indicates that the application uses overlay design. If OVLY is specified for application programs which do not use overlay design, it will result in unnecessary processing overhead. If OVLY is not specified when required, it will cause unnecessary core storage to be used in the message processing region and may eventually cause the message region control program to be abnormally terminated.

DATABASE Macro

The DATABASE macro-instruction defines all data bases to be used by the preceding APPLCTN macro-instruction. It is part of the set APPLCTN, DATABASE, and TRANSACT, which describe the total resource and scheduling requirements of each application known to IMS/360. The DATABASE macro-instruction may be omitted or used one or more times with each APPLCTN macro-instruction.

	DATABASE	DBD = dbdname,
		INTENT = {UPDATE
		SHARE
		EXCLUSIVE},
		LOG= {YES
		NO}

Operand field:

DBD=

specifies the logical name of the data base description (DBD) block as generated using the IMS/360 DBD generation utility. At execution time, the DBD must exist as a load module member in the partitioned data set named in the DBDLIB macro-instruction. The

name of the DBD load module and the DBD=dbdname must be identical. This operand is required.

INTENT=

specifies whether the application program named in the preceding APPLCTN macro-instruction intends to use the data base for read-only, update, or solely to the exclusion of all other applications which may use the same data base. SHARE specifies that the user intends read-only usage.

WARNING: If SHARE is specified, and the program specified in the APPLCTN macro-instruction attempts to perform a get hold, insert, replace, or delete operation against the data base, no checking is performed. The operation is performed, and the call is treated as valid. Regardless of the processing option specified at PSBGEN time, the application program will be scheduled for execution in a processing region. If an application program performs update operations against a data base toward which SHARE intent is declared, the integrity of that data base may be destroyed. In addition, physical coding and pointers contained in the data base records may be so damaged that the entire data base can no longer be accessed using Data Language/I.

UPDATE specifies that the program intends to perform insert, delete, or replace functions against the data base and ensures that no other program which intends to UPDATE is scheduled for execution at the same time. EXCLUSIVE specifies that the program must be scheduled to the exclusion of all other programs which use the same data base, regardless of intent. The default value is UPDATE.

LOG=

If LOG=YES is specified on any DATABASE card for a particular data base, all modifications by any application program are logged. The logging of all segments added, deleted, or replaced in the data base allows data base "backout" during emergency restart. The user is cautioned against specifying logging for a SHARE data base. SHARE logging results in unnecessary overhead in ordinary operations.

TRANSACT Macro

The TRANSACT macro-instruction may be used one or more times with each APPLCTN macro-instruction. It specifies the transaction codes which cause the application program named in the APPLCTN macro-instruction to be scheduled for execution in an IMS/360 Type 1 message processing region. It also provides the IMS/360 control program with information which influences the application program scheduling algorithm.

```
|          | TRANSACT |  CODE = transaction code,
|          |          |
|          |          |  PRTY = (normal, limit, limit
|          |          |          count),
|          |          |
|          |          | MSGTYPE = {(MULTSEG,}{NONRESPONSE)},
|          |          |           {(SNGLSEG,}{RESPONSE)    }
|          |          |
|          |          |  PROCLIM = (count, seconds),
|          |          |
|          |          |  INQUIRY =  {NO }
|          |          |             {YES}
```

Operand field:

CODE=

specifies the transaction code. The transaction code may be one through eight characters in length. The first character of transaction codes and logical terminal names must be any of the 29 characters (A through Z, $, #, and @) as defined by <u>IBM System/360 Operating System: Assembler Language</u> (GC28-6514). Transaction codes and NAME macro-instructions must comprise a set of values, each of which is unique in the system. That is, transaction codes and logical terminal names collectively may not contain duplicates. The CODE operand is required.

PRTY=

specifies the priority levels at which this transaction code contends for scheduling selection with other transaction codes being processed by the system. The normal and limit values may range from 0 through 14 and are coded as one or two numeric digits. The limit count value may range from 1 through 65535. The normal field is the priority assigned to this transaction when the number of input transactions enqueued and waiting to be processed is less than the value specified in the limit count field. The limit priority field is the priority to which this transaction code is raised when the enqueued count of waiting input messages is equal to or exceeds the value specified in the limit count field. Once the priority of this transaction has been raised to the limit priority, it is not reduced to the normal priority until all enqueued messages for this transaction code have been processed by the program specified in the preceding APPLCTN macro-instruction, that is, the input queue is empty. If the limit priority feature is not desired for this transaction, code the normal and limit values equal and the limit count value zero. Default values for normal, limit, and limit count are 1,1, and 65535.

MSGTYPE=

specifies the time at which an incoming message is considered complete and available to be routed to an application program for subsequent processing. MULTSEG means that the incoming message is more than one line in length and is not to become eligible for scheduling to an application program until the terminal operator depresses the EOT key. SNGLSEG specifies that the incoming message is always only one line in length and becomes eligible for scheduling when the terminal operator depresses the EOB key (carriage return if the Auto EOB feature is present).

NONRESPONSE specifies that, upon completion of the input message, single or multiple segment, the terminal is to accept further input without waiting for the completed input message to be

processed. RESPONSE specifies that, upon completion of the input message, single or multiple segment, the terminal and communication line to which it is attached are to accept no further input until the program specified in the APPLCTN macro-instruction has been scheduled, has processed the input message, and has sent an output message to the input terminal. Default value is (MULTSEG, NONRESPONSE).

PROCLIM=

specifies the maximum processing time per message and the maximum number of messages to be processed per application program load in a Type 1 IMS/360 message processing region. The seconds field specifies a numeric value in seconds, which may range from 1 through 65535 and represents the maximum CPU time allowed for each message to be processed in the message processing region. The count field specifies the maximum number of messages which are provided to the application program by the IMS/360 control program for processing without reloading the application program. The count field value may range from 1 through 65534. Code the count field value at 65535 if no limit is to be placed upon the number of messages processed at a single program load. Default value for the PROCLIM operands is (65535, 65535).

The seconds value assigned is used for the purpose of application program erroneous looping control. No attempt need be made to optimize the seconds value for program-transaction execution time. However, the seconds time value assigned should not be less than the expected per-transaction execution time. If the scheduled application program exceeds the product of seconds and count, the application program will be terminated abnormally.

The count value assigned is used to determine how many messages an application program is allowed to process in a single scheduling cycle, that is, program load. When the application program has requested and received the number of messages indicated in the count value, it will receive a "no more messages" indicator upon any subsequent requests from the IMS/360 control program. IMS/360 may, in fact, have other messages enqueued for the application program. Upon receiving the indication that no more messages are available, the message processing application program must terminate, thus making available the region it occupied for rescheduling. This feature enables IMS/360 to allow scheduling of higher priority transactions which may have entered the system while the previous transactions were in process. In addition, if any equal priority transactions are enqueued, they will become eligible for scheduling on a first-in, first-out (FIFO) basis.

INQUIRY=

Entered value is used by data base recovery and DBDUMP only.
and SWITCHED TERMINALS w INQUIRY SIGN ON
If the INQUIRY operand is NO:

1. Data base recovery reprocesses all messages entered against this transaction code.

2. No input is allowed against this transaction code during DBDUMP.

3. Transactions will not be ~~processed~~ accepted when this is No &/[illegible] specified INQUIRY

If the INQUIRY operand is YES, IMS/360 assumes that this transaction code will not cause alterations to data bases.

Therefore:

1. ~~Data base recovery will not reprocess messages entered against this transaction code.~~

2. ~~During DBDUMP, input is allowed against this transaction code.~~

LINEGRP Macro

The LINEGRP macro-instruction defines the beginning of a set of communication lines and physical terminal, logical terminal pool, logical terminal subpool, and logical terminal description macro-instructions which include LINE, TERMINAL, POOL, SUBPOOL, and NAME. These sets are used to describe the user's telecommunications system. The LINEGRP macro-instruction is used to begin a description of one or more lines of the same type, over which the same type of terminal will communicate.

```
┌──────────────────────────────────────────────────────────┐
│           │            │                                 │
│  [name]   │  LINEGRP   │  DDNAME = name,                 │
│           │            │                                 │
│           │            │          {STACTL,    NONSWITCH} │
│           │            │  FEAT =  {TRANSCTL,  SWITCHED }, │
│           │            │                                 │
│           │            │            {2740}               │
│           │            │  UNITYPE = {1050}               │
│           │            │            {2260}               │
│           │            │                                 │
└──────────────────────────────────────────────────────────┘
```

Operand field:

DDNAME=

specifies the DD name that is used to allocate the communication line devices described in the following LINE and TERMINAL macro-instructions. This name is placed in the DD statements generated as a part of the execution procedure called IMS0, which is placed in the procedure library specified in the PROCLIB macro-instruction during Stage 2 of IMS/360 system definition. The operand is required.

FEAT=

specifies certain features that establish part of the characteristics used to determine which lines comprise a line group. The allowable combinations of values for this operand related to terminal unit type are shown in the following table. See also the LINE macro-instruction FEAT operand.

Exhibit 41

OPERAND VALUES	TERMINAL TYPE			
	1050		///////////	2260
	////////////	2740		
	STACTL SWITCHED	STACTL NONSWITCHED	TRANSCTL SWITCHED	STACTL NONSWITCHED
AUTOANS	OPTIONAL	N/A	OPTIONAL	N/A
AUTOPOLL	N/A	OPTIONAL	N/A	N/A
POLL	N/A	OPTIONAL	N/A	OPTIONAL

UNITYPE=

specifies the unit number of the device as either an IBM 1050, 2260, or 2740 communication terminal. All terminals in this LINEGRP must be the same.

LINE Macro

The LINE macro-instruction defines the beginning of a set of TERMINALs and NAME macro-instructions which describe the physical and logical terminals on a single communications line. This macro-instruction is used to describe both switched and nonswitched communication lines. If the line described has only one terminal attached, only one TERMINAL macro-instruction appears after the LINE macro-instruction. Multiple TERMINAL macro-instructions would appear if the description were for a multidrop line. Multiple NAME macro-instructions, each of which describes a logical terminal, may appear after each TERMINAL macro-instruction that follows a LINE macro-instruction. Each LINE macro-instruction must be followed by at least one TERMINAL macro-instruction.

```
[name]    LINE      FEAT =   (AUTOANS )
                             {AUTOPOLL} ,
                             (POLL    )

                    ADDR = hexnumber,
```

Operand field:

ADDR=

specifies the address of the communication line. The address value is three hexadecimal digits ranging from 000 through 6FF.

FEAT=

describes the features of the terminals which are attached to this line. See the table following the FEAT operand of the LINEGRP macro-instruction for restrictions. This operand is required. If the LINEGRP macro specifies SWITCHED, the only allowable FEAT operand is AUTOANS. If the LINEGRP macro specifies NONSWITCH, the only allowable FEAT operands are AUTOPOLL and POLL. There are no default options.

TERMINAL Macro

The TERMINAL macro-instruction describes a physical terminal which must be an input device and may, in addition, be an output device. This macro-instruction describes a physical terminal on a nonswitched line or the representation to BTAM of a physical terminal on a switched line. NAME macro-instructions which follow the TERMINAL macro-instruction supply the logical terminal name(s) associated with the physical terminal at system definition time. Within the definitions and restrictions of terminal security, the first NAME macro-instruction encountered following a TERMINAL macro-instruction becomes the response or input/output logical terminal. Each TERMINAL macro-instruction must be followed by at least one NAME macro-instruction.

[name]	TERMINAL	ADDR = terminal address character,
		[UNIT = 2848 unit address]

Operand field:

ADDR=

specifies the physical terminal addressing character in terminal code, hexadecimal representation. For example, physical terminal address "a" for a 2740 would be coded ADDR = E2.

UNIT=

is the 2848-unit address onto which the specified TERMINAL is attached. This operand is required when the UNITYPE operand on the preceding LINEGRP is 2260. The value range is 40 - A8.

POOL Macro

The POOL macro-instruction describes a pool of logical terminals which are to be associated with a set of switched communication lines. The IMS/360 user need have only one logical terminal pool for all autoanswer or communication lines. All POOL macro-instructions must follow after all LINE macro-instructions within a LINEGRP. See the section of this chapter titled "Teleprocessing Example".

	POOL	FEAT = AUTOANS

Operand field:

FEAT=

specifies the pool of logical terminals to be associated with those physical lines defined by the LINE macro-instructions with the equivalent FEAT operands.

SUBPOOL Macro

The SUBPOOL macro-instruction defines a set of logical terminals within a pool which may be associated with a given physical terminal on a switched communication line when the /IAM command is executed. One or more subpools may be defined within a POOL macro-instruction. At least one must be defined for each POOL macro-instruction.

	SUBPOOL	

Operand field:

There are no operands for this macro. This macro-statement defines the beginning of a subpool set.

NAME Macro

The NAME macro-instruction defines the logical terminal name to be associated with the physical terminal described by a preceding TERMINAL or SUBPOOL macro-instruction. At least one NAME macro-instruction must follow each TERMINAL or SUBPOOL macro-instruction to establish a logical terminal name for the physical terminal or within the subpool.

Only one NAME macro-instruction defining the inquiry logical terminal should follow a TERMINAL macro-instruction which describes the BTAM representation of a physical terminal associated with a switched communication line. Multiple NAME macro-instructions may follow a TERMINAL macro-instruction on a nonswitched communication line or a SUBPOOL macro-instruction where the subpool contains multiple logical terminals.

All inquiry logical terminal names in a system generation must begin with the same first four characters. Only transactions described with the TRANSACT macro-instruction with the operand INQUIRY = YES may be entered through the inquiry logical terminal on a switched line. No subpool logical terminal name may start with the first four characters used for inquiry logical terminal names. See section in Chapter 3 titled "Logical Terminal Types in Switched Communications Network Environment" about transactions that can be input when a user signs on for an inquiry logical terminal.

	NAME	lterm name,
		COMPT= 0 or PTR1

Operand field:

lterm name

defines a name for a logical terminal associated with the preceding TERMINAL or SUBPOOL macro-instruction. The value for this operand may be one to eight alphameric characters. The value assigned to this operand must be unique in the collective group that includes values assigned to the CODE operand of the TRANSACT macro-instruction. The operand is required.

COMPT=

specifies the particular device in a 1050 terminal complex to which the specified terminal is associated. The specified lterm is used to direct output messages to the terminal. Operand values are:

0 or PTR1 = 1050 printer 1

MSGQUEUE Macro

The MSGQUEUE macro-instruction defines the input and output single-line message and multiple-line message data sets desired by the user. This macro-instruction is required.

	MSGQUEUE	QCRIN = (ddname,dsname,unit,serial),
		[QCROUT = (ddname,dsname,unit,serial),]
		MSGIN = (ddname,dsname,unit,serial),
		[MSGOUT = (ddname,dsname,unit,serial),]

		REUSE = {YES, 100
		 NO n}

Operand field:

QCRIN=

specifies the DD name, DS name, unit type, and volume serial of the direct access device upon which the input single-line message data set resides. The value for the unit must be 2311, 2314, 2301, or 2303. The data set itself is not required until execution time. The information from this macro-statement is combined with other information to produce a system execution procedure called IMS0, which is placed in the library named by the user in the PROCLIB macro-instruction. This operand is required.

QCROUT=

specifies the DD name, DS name, unit type, and volume serial of the direct access device upon which the output single-line message data set resides. The field values are subject to the same restrictions as the QCRIN operand. If separate data set control for input and output queue control records is not desired, omit this operand. All single-line messages are maintained on the data set as defined in the QCRIN operand. This operand is optional.

MSGIN=

specifies the DD name, DS name, unit type, and volume serial of the direct access device upon which the input multiple-line message data set resides. The field values are subject to the same restrictions as the QCRIN operand. This operand is required.

MSGOUT=

specifies the DD name, DS name, unit type, and volume serial of the direct access device upon which the output multiple-line message data set resides. The field values are subject to the same restrictions as the QCRIN operand. If separate data set control for input and output message buffers is not desired, omit this operand. If this operand is present, all output message buffers are maintained on this data set. This operand is optional.

REUSE=

specifies, by the first operand, whether reusable queues are desired. The second operand specifies the number of queue records reserved for orderly shutdown of the system queues. When the number of remaining records on any queue data set is equal to or less than the entered value, a /CHECKPOINT DUMPQ command is automatically initiated by IMS/360. If reuse of queue is desired, previously used records will be reused when the number of remaining records reaches this point. However, if there are no records available for reuse, a /CHECKPOINT DUMPQ command will be initiated. A system ABEND will occur if the /CHECKPOINT DUMPQ command cannot be honored. (See /CHECKPOINT DUMPQ command.) The n value of the second operand has a default value of 100 records and a minimum value of 10 records. The maximum value is 36,863 records; however, the entered value cannot be larger than the smallest queue data set (if the value is too large, an immediate /CHECKPOINT DUMPQ will be initiated when IMS/360 is started).

MASTTERM Macro

The MASTTERM macro-instruction identifies the logical terminal that is the master terminal in the generated system.

	MASTTERM	logical name

Operand field:

logical name

is the logical name of an input terminal defined in a previous TERMINAL and NAME macro-instruction set. The master terminal cannot be attached through a switched communication line.

The associated NAME macro-instruction must be the first NAME macro-instruction following the associated TERMINAL macro-instruction. There must be at least two NAME macro-instructions in the TERMINAL and NAME macro-instruction set defining the master terminal.

If the terminal referred to by the associated NAME macro-instruction is on a multidrop line, that is, there are other TERMINAL macro-instructions within the same line set, the referred-to TERMINAL macro-instruction must be the first TERMINAL macro-instruction following the associated LINE macro-instruction.

MACLIB Macro

The MACLIB macro-instruction designates the library upon which macro-definitions (such as PSB and DBD macro-definitions) output from Stage 2 are placed. If the MACLIB macro-instruction is used, the partitioned data set named in the PDS operand must be allocated and cataloged by the user and must exist in the generating system. If the MACLIB statement is omitted, no IMS/360 macro-definitions will be transferred from IMS.GENLIB to any user library, including those required to perform PSB and DBD generation.

	MACLIB	UNIT = name,
		VOLNO = serial,
		PDS = $\left\{\begin{array}{c}\underline{\text{IMS.MACLIB}}\\ \text{name}\end{array}\right\}$
		COPY = $\left\{\begin{array}{l}\underline{\text{UTILITY}}\\ \text{ALL}\end{array}\right\}$

Operand field:

UNIT=

specifies the unit name of the direct access device upon which the macro library PDS is to reside in the generated system. Mandatory entry value must be 2311 or 2314.

VOLNO=

specifies the serial number of the volume that is to contain IMS/360 macro-definitions in the generated system. Mandatory entry.

PDS=

is the name of the macro-definition library upon which the IMS/360 macro-definition will reside in the generated system. If no PDS name is provided, Stage 2 assumes the PDS name to be IMS.MACLIB.

COPY=

specifies which macro-definitions are to be transferred to the PDS specified in the PDS operand. If this operand is omitted, only those macro-definitions required to perform PSB and DBD generation are copied. If ALL is specified, all macro-definitions in IMS.GENLIB are copied. Direct access allocation space requirements may be found in this chapter under the topic "IMS/360 System Data Set Allocation".

RESLIB Macro

The RESLIB statement defines the PDS in the generating and generated system upon which all IMS/360 load modules are placed during Stage 2. It must be a preallocated, cataloged data set in the generating system. It may be SYS1.LINKLIB.

	RESLIB	UNIT = name,
		VOLNO = serial,
		PDS = $\left\{\begin{matrix}\underline{\text{IMS.RESLIB}}\\ \text{name}\end{matrix}\right\}$

Operand field:

UNIT=

specifies the unit name of the direct access device upon which the macro library PDS is to reside in the generated system. Mandatory entry value must be 2311 or 2314.

VOLNO=

specifies the serial number of the volume that is to contain IMS/360 macro-definitions in the generated system. Mandatory entry.

PDS=

specifies the DS name of the library in the generated system upon which the IMS/360 load module library is placed. If this operand is omitted, it is assumed that the PDS named IMS.RESLIB is cataloged and preallocated in the generated system. Direct access allocation space requirements may be found in this chapter under the topic "IMS/360 System Data Set Allocation".

PGMLIB Macro

The PGMLIB macro-instruction designates the library upon which all executable application programs reside.

	PGMLIB	UNIT = name,
		VOLNO = serial,
		PDS = $\left\{\begin{matrix}\underline{\text{IMS.PGMLIB}}\\ \text{name}\end{matrix}\right\}$

Operand field:

UNIT=

specifies the unit name of the direct access device upon which the application program library is to reside in the generated system. If this operand is omitted, the VOLNO operand must also be omitted. System definition then assumes that the data set is cataloged on the generated system.

VOLNO=

specifies the serial number of the volume that is to contain IMS/360 application programs in the generated system. If this operand is omitted, the UNIT operand must also be omitted. See UNIT operand above.

PDS =

is the name of the program library. If the PGMLIB statement is made and the PDS operand is omitted, the default name of the partitioned data set is IMS.PGMLIB. Direct access allocation space requirements may be found in this chapter under the topic "IMS/360 System Data Set Allocation".

PSBLIB Macro

The PSBLIB macro-instruction designates the library upon which the output from the IMS/360 PSB generation utility resides.

	PSBLIB	UNIT = name,
		VOLNO = serial,
		PDS = $\left\{\begin{matrix}\underline{\text{IMS.PSBLIB}}\\ \text{name}\end{matrix}\right\}$

Operand field:

UNIT=

specifies the unit name of the direct access device upon which the PSB library is to reside in the generated system.

VOLNO=

specifies the serial number of the volume that is to contain the PSB library.

PDS=

is the name of the PSB library. If the PSBLIB statement is made and the PDS operand is omitted, the default name of the partitioned data set is IMS.PSBLIB. The library need not be allocated in the generated system. Direct access allocation space requirements may be found in this chapter under the topic "IMS/360 System Data Set Allocation".

DBDLIB Macro

The DBDLIB macro-instruction designates the library upon which output from the IMS/360 DBD generation utility resides.

	DBDLIB	UNIT = name,
		VOLNO = serial,
		PDS = $\left\{\begin{array}{c}\underline{\text{IMS.DBDLIB}}\\ \text{name}\end{array}\right\}$

Operand field:

UNIT=

specifies the unit name of the direct access device upon which the DBD library is to reside in the generated system.

VOLNO=

specifies the serial number of the volume that is to contain the DBD library.

PDS=

is the name of the DBD library. If the DBDLIB statement is made and the PDS operand is omitted, the default name of the partitioned data set is IMS.DBDLIB. The library need not be allocated in the generated system. Direct access allocation space requirements may be found in this chapter under the topic "IMS/360 System Data Set Allocation".

PROCLIB Macro

The PROCLIB macro-instruction designates the library upon which procedure output from Stage 2 is placed. If the PROCLIB macro-instruction is used, the PDS name specified must exist in the generated system. If the PROCLIB statement is omitted, no user procedures are generated. If the statement is included, and if all conditions stated in other generation macros which affect procedure generation are satisfied, the following procedures are generated.

Name	Procedure
PSBGEN	Generate (assemble) PSB and link to appropriate library
DBDGEN	Generate (assemble) DBD and link to DBDLIB library
IMS0 and IMS1	Execute IMS/360 online system
IMS	DASD reader procedure to invoke IMS0 procedure
IMSMSG	Execute Type 1 processing region
IMSBATCH	Execute Type 2 processing region
DLIBATCH	Execute stand-alone Type 3 processing region
IMSCOBGO	Execute COBOL compile and go Type 3 batch processing, link to PGMLIB
IMSPLIGO	Execute PL/I compile and go Type 3 processing, link to PGMLIB

IMSCOBOL	Compile and link COBOL to PGMLIB library
IMSPLI	Compile and link PL/I to PGMLIB library
SECURITY	Execute security maintenance program assemble, and link to RESLIB library
DLITCBL	Linkage editor input for COBOL compiler
DLITPLI	Linkage editor input for PL/I compiler

	PROCLIB	UNIT = name,
		VOLNO = serial,

		PDS= {<u>IMS.PROCLIB</u>
		name}

Operand field:

UNIT=

specifies the unit name of the direct access device upon which the macro library PDS is to reside in the generated system. If this operand is omitted, the VOLNO operand must also be omitted. System definition then assumes that the data set is cataloged on the generated system.

VOLNO=

specifies the serial number of the volume that is to contain IMS/360 procedures in the generated system. If this operand is omitted, the UNIT operand must also be omitted. See UNIT operand above.

PDS=

is the name of the procedure library upon which the IMS/360 procedures reside in the generated system. If no PDS name is provided, Stage 2 assumes the preallocated PDS name to be IMS.PROCLIB. Direct access allocation space requirements may be found in this chapter under the typic "IMS/360 System Data Set Allocation".

IMSGEN Macro

The IMSGEN macro-instruction is used to specify the data sets, volumes, and I/O devices required for the definition process, the system definition output options.

The IMSGEN macro-instruction must be the last macro-instruction in the Stage 1 input stream. It must be followed immediately by an assembler END statement.

	IMSGEN	UT1SDS = dsname,
		ASMPRT = $\left\{\begin{array}{l}\underline{\text{OFF}}\\ \text{ON}\end{array}\right\}$,
		[LEPRT = (value^1,.....value^2)]

Operand field:

UT1SDS=

specifies the name of the utility data set to be used during Stage 2 of system definition by the assembler and linkage editor.

ASMPRT=

specifies whether assembly listings are to be procured for the modules assembled during system definition. ON specifies that assembly listings are to be generated; OFF, that assembly listings are not to be generated.

LEPRT= value

specifies linkage editor print options as one or two of the following values.

Value	Print Option
LIST	List of control statements in card-image format
MAP	Module map
XREF	Cross-reference table (XREF includes the MAP option)

If this parameter is omitted, only linkage editor error messages, if any, are printed. For a more detailed description of these options, see the publication *IBM System/360 Operating System: Linkage Editor* (C28-6538).

IMSTEST Macro

This IMSTEST macro-instruction is designed to provide a means of generating an alternate IMS/360 nucleus. This macro-instruction alters the Stage 2 job stream of system definition to provide only those job steps necessary to create the system control blocks, a composite system control block module, and the composite system nucleus. A system definition must have already been performed, creating a standard nucleus (as previously described), and the MACLIB specification of that standard generation must have included the COPY=ALL parameter. It is assumed that the same SVC numbers and the channel end appendage suffix specified in the standard system definition are also specified when generating an alternate IMS/360 nucleus using the IMSTEST macro. (While no check is made to verify that they are the same, execution of the alternate nucleus would be impossible if they were not.) It is also assumed that DFSILNK0 is in SYS1.LINKLIB (if not, a JOBLIB must be added to the Stage 2 job stream).

Note: Since the basic purpose of the alternate Stage 2 job stream is to provide the user an alternate composite system control block module (DFSIBLK) and an executable IMS/360 nucleus (DFSINUC), the Stage 2 job stream will not affect those IMS/360 modules whose function is not altered by or part of the alternate nucleus. These modules are:

DFSIRC00 -	region controller module
DFSIPC00 -	program controller module
DFSIDLL0 -	Data Language/I block loader module
DFSISVV0 -	MVT interregion SVC routines
DFSISVF0 -	MFT interregion SVC routines
DFSIOCE0 -	OSAM channel end appendage module

The above modules remain unaffected, to allow the alternate and the standard IMS/360 nuclei to operate with the same Operating System/360 nucleus, if the same SVC numbers and OSAM channel end appendage suffix values were provided in both generations.

To invoke the alternate IMS/360 system definition, the user includes an IMSTEST control card with the standard Stage 1 IMS/360 control cards. The IMSTEST control card must precede all other IMS/360 system definition control cards and supply three cataloged PDS names to be used during execution of the Stage 2 job. See the teleprocessing example at the end of this chapter.

The user supplies a suffix code to be appended to the composite control block module (DFSIBLK) and the executable nucleus module (DFSINUC) member names when they are link edited into the specified RESLIB. A separate cataloged data set can be provided to retain the individual control block modules as they are assembled prior to constructing the composite control block module (DFSIBLK). Since the individual modules are needed only during the system definition phase, this data set can be scratched after DFSIBLK is created. Other cataloged PDS names are provided containing the IMS/360 macros and the IMS/360 load modules.

	IMSTEST	GLIB = IMS.MACLIB,
		LLIB = IMS.RESLIB,
		BLIB = IMS.RESLIB,
		CODE = X

Operand field:

GLIB =

specifies the cataloged PDS to be used as SYSLIB by the assembler steps of Stage 2 and must contain the IMS/360 macros necessary to compile the control blocks. The default value is IMS.MACLIB.

LLIB =

specifies the cataloged PDS containing the executable IMS/360 load modules. The modules in this PDS should correspond to the modules moved to the specified RESLIB by Step 1 of the standard IMS/360 system definition. The default value is IMS.RESLIB.

BLIB =

specifies the cataloged PDS into which the individual system control blocks assembled by Stage 2 will be placed. All assembled system control blocks will be placed in this data set except DFSIBLK and DFSINUC. This may be a temporary data set existing only for the duration of the Stage 2 job. The default value is IMS.RESLIB. Note, however, that if the default value is used for this operand, the control block modules will replace those created by the standard IMS/360 system definition.

CODE =

specifies a one-character suffix to be appended to the load module member names DFSIBLK and DFSINUC upon placing them into the specified RESLIB. The default value is X. Note that the suffix should not be 0 or the original IMS/360 nucleus will be destroyed.

Maximum System Definition Macro-Instruction Occurrences

The IMS/360 system produced by IMS/360 system definition has defined limitations on the number of each system resource type. The maximum number of any resource type is controlled by the maximum number of occurrences of macro-instructions in each system definition. The following table provides a definition of the resource limits:

Macro - Instruction	Maximum Occurrences of Macro-Instruction
IMSCTRL	1
APPLCTN	254
DATABASE	254
TRANSACT	509
LINEGRP	255
LINE	253-A
TERMINAL	254-B
NAME	509
POOL	254-C
SUBPOOL	254-D
MASTTERM	1
MSGQUEUE	1
MACLIB	1
RESLIB	1
PGMLIB	1
PSBLIB	1
DBDLIB	1
PROCLIB	1
IMSGEN	1

where:

A is the number of MAXREGN defined in the IMSCTRL macro-instruction plus the number of occurrences of the POOL macro-instruction.

B is the number of occurrences of the SUBPOOL macro-instruction.

C is the number of MAXREGN defined in the IMSCTRL macro-instruction plus the number of occurrences of the LINE macro-instruction.

D is the number of occurrences of the TERMINAL macro-instruction.

System Definition Job Control Language Statements

The Job Control Language (JCL) for Stage 1 of system definition is for an assembly execution. Use the standard Operating System/360 Assembler procedure (ASMFC) with the following SYSLIB DD card override. The user generates a card deck of the following format and places these cards in the job stream.

```
//          JOB
//          EXEC  ASMFC
//ASM.SYSLIB DD DSNAME=IMS.GENLIB,DISP=OLD
//ASM.SYSIN  DD *

        (    IMS/360 Stage 1 -    )
        {INPUT CONTROL CARDS -    }
        (SYSTEM DEFINITION PROGRAM)

/*
```

The resulting output deck completes Stage 1. The JCL for Stage 2 is only a JOB card supplied by the user generating the system and placed in front of the punched card deck received from Stage 1. Place this deck of cards in the job stream.

Examples of system definition are shown at the end of this chapter.

IMS/360 System Data Sets

The various partitioned data sets used by IMS/360 for libraries must be defined and allocated by the user. The DCB characteristics for these data sets should be specified at time of allocation. In all cases, these DCB characteristics should be equated to existing Operating System/360 partitioned data sets. This can be done with a DCB= operand of the DD card used for allocation. The following lists the IMS/360-Operating System/360 data sets which would have equivalent DCB characteristics:

IMS/360	Operating System/360
IMS.RESLIB	SYS1.LINKLIB
IMS.PGMLIB	SYS1.LINKLIB
IMS.PROCLIB	SYS1.PROCLIB
IMS.MACLIB	SYS1.MACLIB
IMS.PSBLIB	SYS1.LINKLIB
IMS.DBDLIB	SYS1.LINKLIB

It is suggested that the Operating System/360 utility program IEHPROGM be used to allocate and catalog these IMS/360 system data sets.

To summarize, the different libraries made available or modified by the user or by the system definition program are as follows:

Complete System Definition	Type 3 Batch System Definition
IMS.RESLIB	IMS.RESLIB
IMS.MACLIB	IMS.MACLIB
IMS.PSBLIB	IMS.PSBLIB
IMS.PGMLIB	IMS.PGMLIB
IMS.PROCLIB	IMS.PROCLIB
IMS.DBDLIB	IMS.DBDLIB
SYS1.SVCLIB (OSAM channel end appendage)	SYS1.SVCLIB
SYS1.LINKLIB (Link pack modules)	SYS1.LINKLIB
IMS. MESSAGE QUEUE DATA SETS	------------
SYS1.NUCLEUS (Type 1 and 2 SVC's)	SYS1.NUCLEUS (Type 2 SVC)

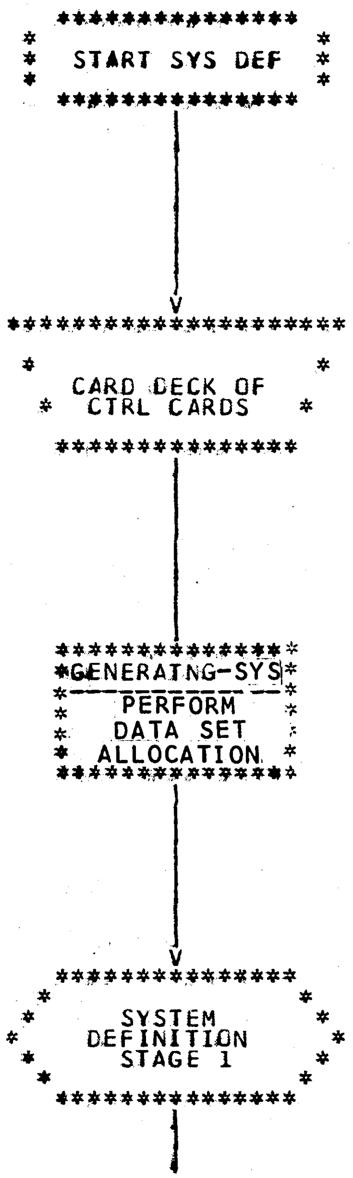
IMS/360 System Data Set Allocation

Space allocation for IMS/360 MACRO, PSB, DBD, PROGRAM, PROCEDURE, and RESLIB libraries is dependent upon user requirements. Space requirements for user libraries of programs, program specification blocks, and data base definition blocks will depend entirely upon the user's operating environment. Some examples may be useful:

- DBD Library - Each DBD (one per data base) requires approximately 1500 to 2500 bytes of direct access storage. Exact requirements depend upon the number of data set groups, segments, fields, and hierarchical levels.
- PSB Library - Each PSB (one per program) requires approximately 250 to 500 bytes of direct access storage. Exact requirements depend upon the number of data bases used in PSB and the number of sensitive segments.
- PROCLIB Library - About 10 tracks (2314) of space are required.
- RESLIB Library - About 20 cylinders of 2314 space are required.
- MACLIB Library - About 10 cylinders of 2314 space are required for ALL macro-instructions. About one cylinder is required for PSBGEN and DBDGEN macro-instructions only.
- PGMLIB Library - This contains application programs.

System Definition Guide

Execution of the system definition utility is shown in the general flowchart form following. It provides for both Stage 1 and Stage 2 of system definition and all the other requirements to make IMS/360 operative.



Start system definition.

Stage 1 system definition is a deck of control cards prepared from the macro-instructions of system definition shown earlier in this chapter.

The computer system used to execute Stage 1 and Stage 2 of system definition need not be the actual IMS/360 computer system. However, the version of Operating System/360 used for Stage 2 must be the same version under which the defined system will be executed. If it is not the same system, perform data set allocation for IMS.GENLIB and IMS.LOAD, and specify the data set in the OBJPDS of the operand of the IMSGEN macro.

System definition Stage 1 requires an assembly run to compile the control statements. The assembly requires its SYSLIB DD statement to point to the IMS/360 system definition macro data set IMS.GENLIB.

Go to next page

```
               |
               |
               V
    *********************
     *                 *
        PUNCHED CARD
      *     DECK      *
      ***************
```

System definition Stage 2 takes as input a punched deck of cards created as output from Stage 1.

```
               |
               |
               |
               V
      ***************
     *               *
    *     SYSTEM      *
   *    DEFINITION     *
    *     STAGE 2     *
     *               *
      ***************
```

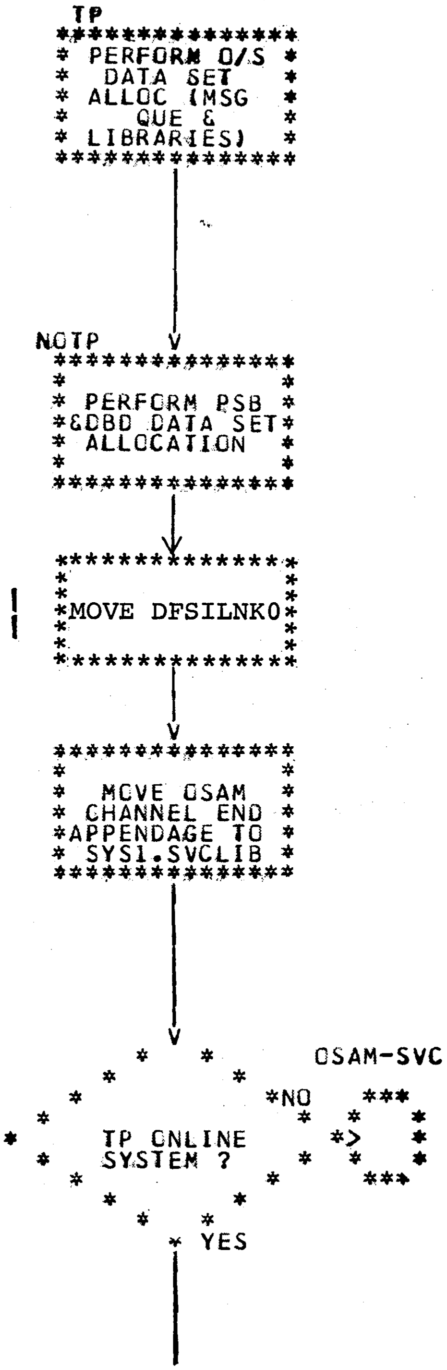
Perform Stage 2 of system definition.

```
               |
               |
               |
               V
      ***************
      * END OF SYS  *
      *DEF; CREATED *
      *  AS MAX -   *
      * IMSNUC & 3  *
      *    LIBS     *
      ***************
```

When Stage 2 is complete, system definition creates an IMS/360 nucleus and three libraries (as maximum output), IMS.RESLIB, IMS.MACLIB, and IMS.PROCLIB, on the preallocated data sets specified on the generating system.

```
               |
               |
               |
               V
            *     *               TP
         *           *
      *                *YES   ***
    *                     *  *   *
   *    A TP ONLINE        *>    *
    *    SYSTEM ?         *  *   *
      *                *      ***
         *           *
            *     *           ***
              * NO           *   *
              ----------->*      *
                             *   *
                              ***
                             NOTP
```

Is the IMS/360 system that was first generated a teleprocessing (online) system? If Yes, go to label TP. If No, go to label NOTP.


TP
PERFORM O/S DATA SET ALLOC (MSG QUE & LIBRARIES)
NOTP
PERFORM PSB &DBD DATA SET ALLOCATION
MOVE DFSILNK0
MOVE OSAM CHANNEL END APPENDAGE TO SYS1.SVCLIB
OSAM-SVC
NO
TP ONLINE SYSTEM ?
YES

Label TP: If the system being generated is to process Type 1 and/or Type 2 processing programs, the message queue data sets must be allocated on the computer system under which IMS/360 is to operate.

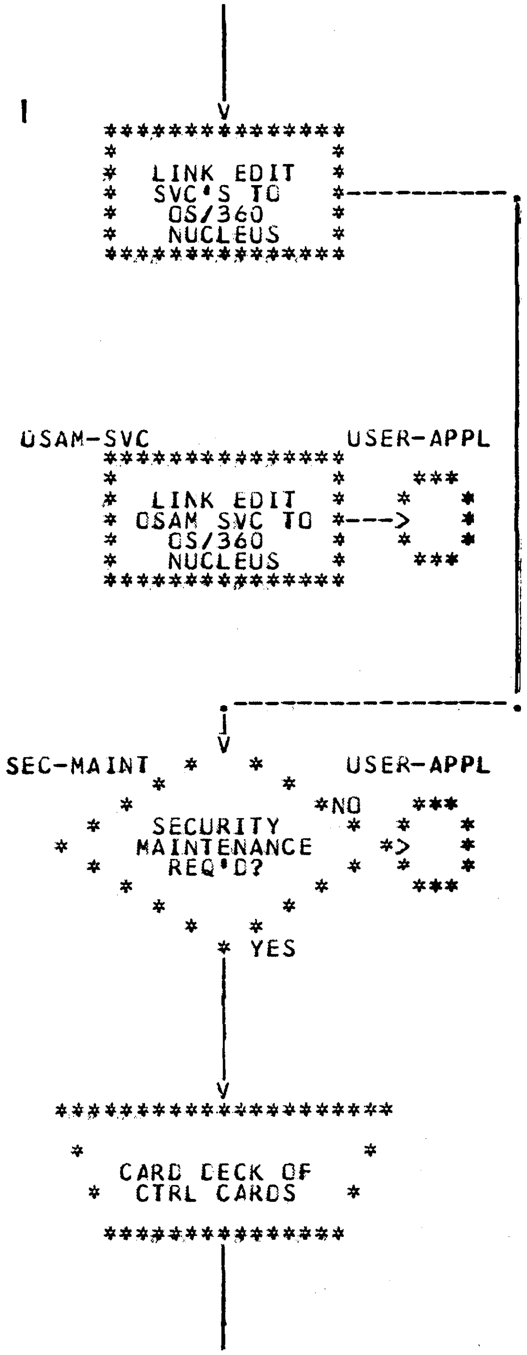
Label NOTP: Perform PSB and DBD data set allocation, whether generated system is a Type 1 and 2, or a Type 3 processing region system.

Move DFSILNK0 from IMS.RESLIB to SYS1.LINKLIB.

System definition creates OSAM channel end appendage and places this in IMS.RESLIB. OSAM channel end appendage must be moved to SYS1.SVCLIB (described later in this manual).

Again, is this an online (teleprocessing) system? If No, go to label OSAM-SVC.

Go to next page


LINK EDIT SVC'S TO OS/360 NUCLEUS
OSAM-SVC
USER-APPL
LINK EDIT OSAM SVC TO OS/360 NUCLEUS
SEC-MAINT
USER-APPL
SECURITY MAINTENANCE REQ'D?
NO
YES
CARD DECK OF CTRL CARDS

The three user SVC load modules created in Stage 2 of system definition must be link-edited into the OS/360 nucleus prior to attempting to execute IMS/360.

Label OSAM-SVC: The OSAM SVC is required for system definition. Link-edit OSAM SVC to the Operating System/360 nucleus.

Label SEC-MAINT: For user's system, is the IMS/360 security maintenance program required? If either terminal or password security is required, the IMS/360 security maintenance program must be run.

Set up control cards for input card deck to the security maintenance program. (See the description of control cards in Chapter 6 of this manual.)

```
               |
               |
               V
        ***************
        *             *
        *RUN SECURITY *
        * MAINTENANCE *
        *   PROGRAM   *
        *             *
        ***************
```

Execute the security maintenance program (SMP).

```
               |
               |
               V
        ***************
        *             *
        *OUTPUT OF SEC*
        * MAINT PGM - *
        *   CHANGES   *
        *   IMSNUC    *
        ***************
```

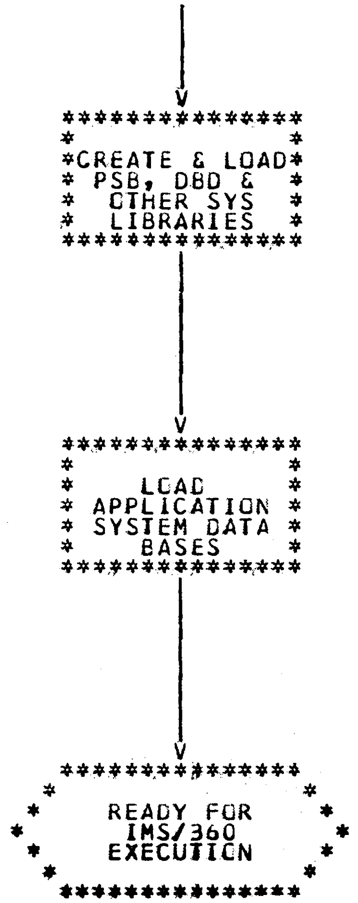
The output of SMP is added to the library specified in the RESLIB macro-instruction.

```
               |
               |
               V
        ***************
        *NRESTART     *
        *-------------*
        *INITIATE NEW *
        * SECURITY PGM*
        *             *
        ***************
```

The result of the SMP does not become effective until the next normal restart (cold start). (See Chapter 5 of this manual for more details.)

```
               |
               |
USER-APPL      V
        ***************
       *               *
      *      USER       *
     *    APPLICATION    *
      *  PGM FUNCTIONS  *
       *               *
        ***************
               |
               |

       Go to next page
```

Label USER-APPL: The user must perform all the application programming functions; that is, load his application programs in the libraries and the names in the directories, etc.


CREATE & LOAD PSB, DBD & OTHER SYS LIBRARIES
LOAD APPLICATION SYSTEM DATA BASES
READY FOR IMS/360 EXECUTION

The PSB and DBD data sets must be loaded prior to executing IMS/360. Any other IMS/360-oriented procedures or data sets desired can also be loaded at this time.

The application system data bases must be loaded before execution of IMS/360 can proceed.

Execute IMS/360 (IPL) per instructions in Chapter 5 of the IMS/360 Operations Manual, Volume II - Machine Operations.

IMS/360 Supervisor Call Routines

The IMS/360 system utilizes three supervisor call (SVC) routines. Two of these are used for interregion communication; the third routine is used by OSAM to create its multivolume data extent block (DEB). All three routines are required for the online IMS/360 system. Only the OSAM SVC routine is required for Type 3 region processing. IMS/360 system definition creates these routines with user-defined SVC numbers. The IMS/360 user must link-edit these routines with the Operating System/360 nucleus. The next section of this chapter explains how to perform the link edit (Items 17 and 19 on the Systems Operation checklist).

Inclusion of IMS/360 SVC Routines in Operating System/360 Nucleus

Three user SVC routines must be added to the Operating System/360 nucleus for execution of the IMS/360 system. Only one of these routines is required if Type 3 region execution is used exclusively. The SVC routines are created by IMS/360 system definition from macro-instructions. The SVC numbers utilized may be specified by the IMS/360 system user. The load modules which represent the SVC routines are placed in IMS.RESLIB by system definition. The two SVC routines used for interregion (partition) communication are Type 1 SVC's. The SVC routine used for OSAM is a Type 2 SVC.

When the IMS/360 user performs his Operating System/360 system generation, the appropriate accommodations must be made for the later incorporation of the SVC routines. The IMS/360 SVC routines need not and normally would not be incorporated at Operating System/360 system generation. They may, however, be incorporated at that time, if desired. The following SVCTABLE macro-instructions should be included in the Stage 1 input to Operating System/360 system generation no matter when the SVC routines are incorporated.

```
            SVCTABLE    nnn-T1-SO
            SVCTABLE    nnn-T1-SO
            SVCTABLE    nnn-T2-SO
```

If the actual SVC routines are not incorporated during Operating System/360 system generation, three "dummy" load modules should be placed in the RESMODS partitioned data set. This should be done prior to Stage 2 of Operating System/360 system generation. These modules are of the format:

```
            IGCXXX  CSECT
                    BR  14
                    END
```

where XXX is the unique SVC number. This effectively "no-ops" the SVC number.

The alternate approach, which would cause inclusion of the actual SVC routines during Operating System/360 system generation, requires placement of the actual SVC modules into the partitioned data set referred to by the RESMODS macro-instruction. This would require IMS/360 system definition execution prior to Stage 2 of Operating System/360 system generation. The RESMODS control card could then refer to the IMS.RESLIB data set for the incorporation of the SVC routines.

If the SVC routines are added after Operating System/360 system generation, the technique is to relink-edit the Operating System/360 nucleus. Basically, this involves replacing the "dummy" SVC routines through the link-edit with the actual SVC routines. The best explanation for performing this link-edit is to:

1. Start with JCL and control cards of link-edit step from Stage 2 of OS/360 system generation.

2. Provide an additional card for the IMS.RESLIB data set to access the SVC modules.

3. Provide an additional DD card to reference the SYS1.NUCLEUS data set other than //SYSLMOD.

4. Provide additional INCLUDE control cards for the three SVC routines from IMS.RESLIB immediately after the INSERT control cards of the original link-edit.

5. Replace the INCLUDE cards from the original Operating System/360 nucleus link-edit with one INCLUDE card for the old Operating System/360 nucleus (that is, the one without the SVC routines).

6. Provide a NAME card for the new Operating System/360 nucleus (for example, IEANUC0X).

It may be good practice to consider the output from the link-edit of the nucleus as another member in SYS1.NUCLEUS (for example, IEANUC02). The OS/360 Operator's Manual (GC28-6540) explains how to IPL an alternate Operating System/360 nucleus. If everything executes properly, then IEANUC02 can be renamed IEANUC01.

OSAM Channel End Appendage

OSAM requires a channel end appendage module created as a load module during execution of IMS/360 system definition. The module is distributed under the name DFSIOCE0 and is renamed during system definition to the user-specified IGG019XX. The created module is placed in IMS.RESLIB (Item 18 on the Systems Operation checklist).

OSAM Appendage to SYS1.SVCLIB

It is the user's responsibility to move the created OSAM appendage module from IMS.RESLIB to SYS1.SVCLIB. This should be done using the Operating System/360 IEHMOVE program.

DFSILNK0 to SYS1.LINKLIB

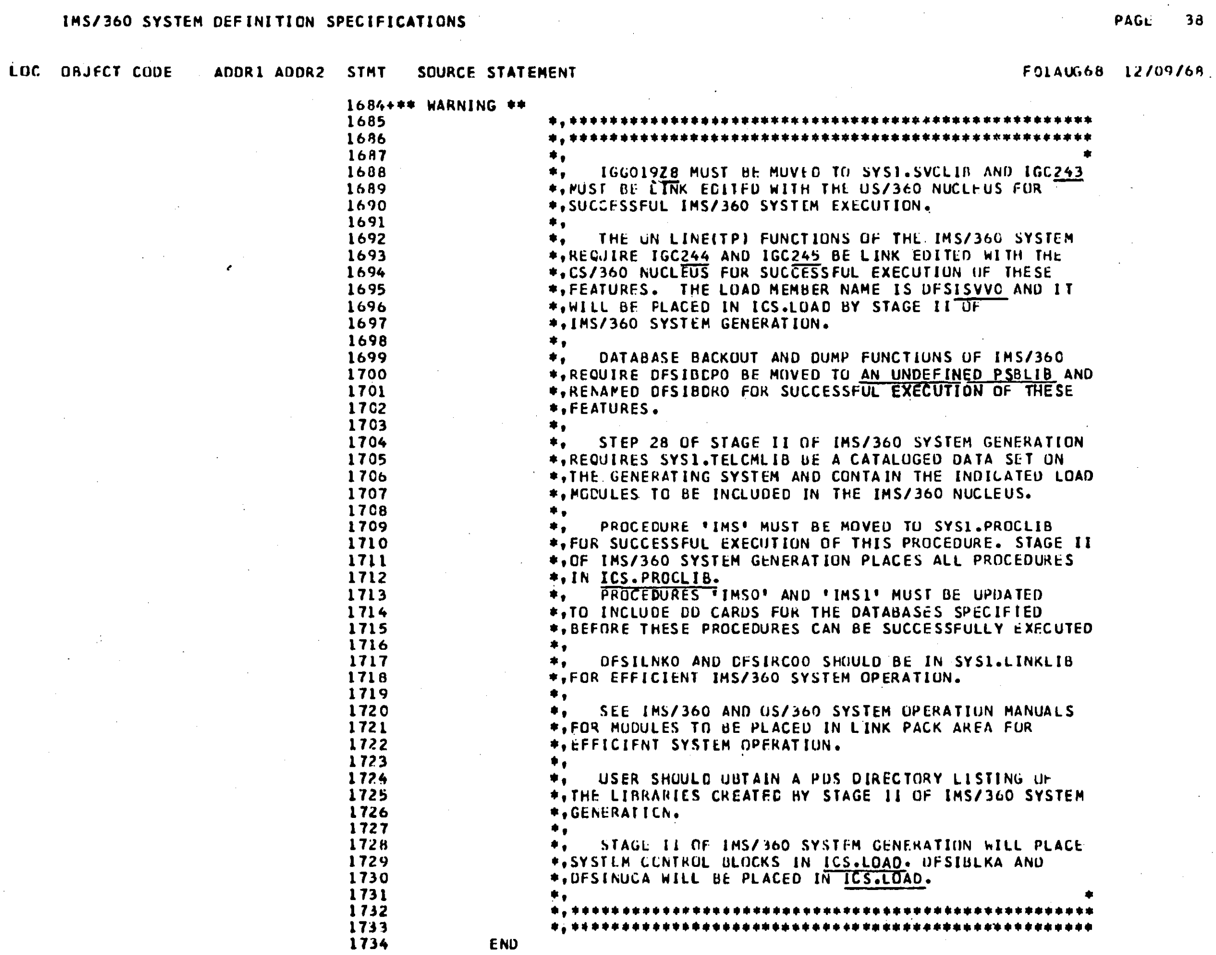
Prior to using any of the generated assembler or compiler procedures, the user must move the module DFSILNK0 from IMS.RESLIB to SYS1.LINKLIB. This module permits the use of SHR disposition on SYSLMOD data sets in the link steps of procedures. This module invokes the linkage editor under the alias name LINKEDIT.

System Definition Stage 1 Output Warnings

The following machine listing is an output example from Stage 1 of IMS/360 system definition. This listing informs the IMS/360 system user of actions which must be performed prior to IMS/360 system execution.

The following assumptions are made:

- Z8 are the last two letters of the OSAM channel end appendage chosen by the system user.
- 244 and 245 are the interregion SVC numbers chosen by the user.
- 243 is the OSAM SVC number chosen by the user.
- The PSBLIB card was omitted during Stage 1 of system definition.
- The PROCLIB card indicated PDS name of ICS.PROCLIB.
- The RESLIB card indicated PDS name of ICS.LOAD.



```
IMS/360 SYSTEM DEFINITION SPECIFICATIONS                                                         PAGE  38

LOC  OBJECT CODE     ADDR1 ADDR2  STMT   SOURCE STATEMENT                                   F01AUG68  12/09/68
                                  1684+** WARNING **
                                  1685              *,****************************************************
                                  1686              *,****************************************************
                                  1687              *,                                                   *
                                  1688              *,   IGG019Z8 MUST BE MOVED TO SYS1.SVCLIB AND IGC243
                                  1689              *,MUST BE LINK EDITED WITH THE OS/360 NUCLEUS FOR
                                  1690              *,SUCCESSFUL IMS/360 SYSTEM EXECUTION.
                                  1691              *,
                                  1692              *,   THE ON LINE(TP) FUNCTIONS OF THE IMS/360 SYSTEM
                                  1693              *,REQUIRE IGC244 AND IGC245 BE LINK EDITED WITH THE
                                  1694              *,OS/360 NUCLEUS FOR SUCCESSFUL EXECUTION OF THESE
                                  1695              *,FEATURES.  THE LOAD MEMBER NAME IS DFSISVVO AND IT
                                  1696              *,WILL BE PLACED IN ICS.LOAD BY STAGE II OF
                                  1697              *,IMS/360 SYSTEM GENERATION.
                                  1698              *,
                                  1699              *,   DATABASE BACKOUT AND DUMP FUNCTIONS OF IMS/360
                                  1700              *,REQUIRE DFSIBCPO BE MOVED TO AN UNDEFINED PSBLIB AND
                                  1701              *,RENAMED DFSIBDRO FOR SUCCESSFUL EXECUTION OF THESE
                                  1702              *,FEATURES.
                                  1703              *,
                                  1704              *,   STEP 28 OF STAGE II OF IMS/360 SYSTEM GENERATION
                                  1705              *,REQUIRES SYS1.TELCMLIB BE A CATALOGED DATA SET ON
                                  1706              *,THE GENERATING SYSTEM AND CONTAIN THE INDICATED LOAD
                                  1707              *,MODULES TO BE INCLUDED IN THE IMS/360 NUCLEUS.
                                  1708              *,
                                  1709              *,   PROCEDURE 'IMS' MUST BE MOVED TO SYS1.PROCLIB
                                  1710              *,FOR SUCCESSFUL EXECUTION OF THIS PROCEDURE. STAGE II
                                  1711              *,OF IMS/360 SYSTEM GENERATION PLACES ALL PROCEDURES
                                  1712              *,IN ICS.PROCLIB.
                                  1713              *,   PROCEDURES 'IMS0' AND 'IMS1' MUST BE UPDATED
                                  1714              *,TO INCLUDE DD CARDS FOR THE DATABASES SPECIFIED
                                  1715              *,BEFORE THESE PROCEDURES CAN BE SUCCESSFULLY EXECUTED
                                  1716              *,
                                  1717              *,   DFSILNKO AND DFSIRCOO SHOULD BE IN SYS1.LINKLIB
                                  1718              *,FOR EFFICIENT IMS/360 SYSTEM OPERATION.
                                  1719              *,
                                  1720              *,   SEE IMS/360 AND OS/360 SYSTEM OPERATION MANUALS
                                  1721              *,FOR MODULES TO BE PLACED IN LINK PACK AREA FOR
                                  1722              *,EFFICIENT SYSTEM OPERATION.
                                  1723              *,
                                  1724              *,   USER SHOULD OBTAIN A PDS DIRECTORY LISTING OF
                                  1725              *,THE LIBRARIES CREATED BY STAGE II OF IMS/360 SYSTEM
                                  1726              *,GENERATION.
                                  1727              *,
                                  1728              *,   STAGE II OF IMS/360 SYSTEM GENERATION WILL PLACE
                                  1729              *,SYSTEM CONTROL BLOCKS IN ICS.LOAD. DFSIBLKA AND
                                  1730              *,DFSINUCA WILL BE PLACED IN ICS.LOAD.
                                  1731              *,                                                   *
                                  1732              *,****************************************************
                                  1733              *,****************************************************
                                  1734         END
```

Underlines refer to text above.

System Procedures

If a PROCLIB macro-instruction is presented as is suggested in the Stage 1 input of IMS/360 system definition, certain procedures are created and placed in the library specified. These procedures are complete only to the extent of the information made available through the optional library macro-instructions. For example, if the user does not specify a name for the program specification block library, the default DSNAME value of IMS.PSBLIB is used in the generated procedure. This can mean that the created procedures are not executable in the IMS/360 operating environment. Other procedures have defaults as specified in the system definition macro-instruction. Created procedures should be examined carefully to determine whether the desired JCL has in fact been correctly specified. If an online IMS/360 system has been defined, particular attention should be devoted to the terminal device allocation created. At the end of Stage 1 definition, a table of unit addresses and of logical and physical terminals is printed. Before executing the defined system, the cross-reference table should be examined to ensure that the specifications provided in Stage 1 define the desired system. If all optional library macro-statements are included as input to Stage 1 of IMS/360 system definition, the following procedures are created:

Procedure Library Member Name	Description
PSBGEN	A two-step assemble and link-edit procedure to produce program specification blocks
DBDGEN	A two-step assemble and link-edit procedure to produce data base definition blocks
IMSCOBOL	A two-step compile and link-edit procedure for IMS/360 applications written in COBOL
IMSPLI	A two-step compile and link-edit procedure for IMS/360 applications written in PL/I
DLIBATCH	A one-step execution procedure for stand-alone Data Language/I Type 3 processing region
IMSCOBGO	A three-step compile, link-edit, and go procedure combining the procedures IMSCOBOL and DLIBATCH
IMSPLIGO	A three-step compile, link-edit, and go procedure combining the procedures IMSPLI and DLIBATCH
IMS	DASD reader procedure to read IMS0 procedure into Operating System/360 job stream from direct access devices
IMS0	Execution of IMS/360 Type 0 region, the IMS/360 online control program with complete JOB PROCEDURE LIBRARY
IMS1	Execution of IMS/360 Type 0 region, the IMS/360 online control program with JCL from system input stream
IMSMSG	Execution of IMS/360 Type 1 region, a message processing region
IMSBATCH	Execution of IMS/360 Type 2 region, an online batch region
SECURITY	A three-step execution, assembly, and link-edit procedure for terminal and password security which invokes the security maintenance program
DLITCBL	A SYSIN member used by the link steps of procedures IMSCOBOL and IMSCOBGO
DLITPLI	A SYSIN member used by the link steps of IMSPLI and IMSPLIGO. Note that entry point IHESAPD is specified. This corresponds to the PARM value OPT=1 in the corresponding compile procedures.

MFDBLOAD

A Data Language/I batch Type 3 execution procedure used to load the sample problem data base. Input data for the data base procedure is contained in the MFDFSYSN member of the user's MACLIB when the COPY=ALL option is used in the MACLIB statement. This procedure is not included in the user's library unless COPY=ALL is used in the MACLIB statement.

MFDBDUMP

This is a procedure to dump the sample problem data base onto a SYSOUT data set. This procedure is not included in the user's PROCLIB unless the COPY=ALL option is used in the MACLIB statement.

Note that the generated procedures accommodate the user Type 1 programming system, either MVT or MFT. Also, volume serial and unit appear if the specified library is not cataloged. Prior to using the generated assembler or compiler procedures, the user must move the module named DFSILNK0 to SYS1.LINKLIB. This module permits use of SHR disposition on the SYSLMOD data sets used by the various procedures.

Specific examples of the default procedures follow:

<u>MEMBER</u> <u>NAME</u> <u>PSBGEN</u>

```
//        PROC     MBR=TEMPNAME
//C       EXEC     PFM=IEUASM,PARM='LOAD,NODECK',REGION=92K
//SYSLIB     DD    DSNAME=IMS.MACLIB,DISP=SHR
//           DD    DSNAME=SYS1.MACLIB,DISP=SHR
//SYSGO      DD    UNIT=SYSDA,DISP=(,PASS),DCB=(BLKSIZE=400,          X
//                 RECFM=FB,LRECL=80),SPACE=(80,(100,100),RLSE)
//SYSPRINT   DD    SYSOUT=A,DCB=(LRECL=121,RECFM=FBM,BLKSIZE=605,     X
//                 SPACE=(121,(500,500),RLSE,,ROUND)
//SYSUT1     DD    UNIT=SYSDA,DISP=(,DELETE),SPACE=(1700,(100,50))
//SYSUT2     DD    UNIT=SYSDA,DISP=(,DELETE),SPACE=(1700,(100,50))
//SYSUT3     DD    UNIT=(SYSDA,SEP=(SYSLIB,SYSUT1,SYSUT2)),           X
//                 SPACE=(1700,(100,50))
//L       EXEC     PGM=DFSILNK0,PARM='XREF,LIST',COND=(0,LT,C),       X
//                 REGION=100K
//SYSLIN     DD    DSNAME=*.C.SYSGO,DISP=(OLD,DELETE)
//           DD    DDNAME=SYSIN
//SYSPRINT   DD    SYSOUT=A,DCB=(LRECL=121,RECFM=FBA,BLKSIZE=605),   X
//                 SPACE=(121,(100,100),RLSE)
//SYSLMOD    DD    DSNAME=IMS.PSBLIB(&MBR),DISP=SHR
//SYSUT1     DD    UNIT=(SYSDA,SEP=(SYSLMOD,SYSLIN)),                 X
//                 DISP=(,DELETE),SPACE=(1024,(100,10),RLSE)
```

MEMBER NAME DBDGEN

```
//       PROC     MBR=TEMPNAME
//C      EXEC     PGM=IEUASM,PARM='LOAD,NODECK',REGION=92K
//SYSLIB     DD   DSNAME=IMS.MACLIB,DISP=SHR
//           DD   DSNAME=SYS1.MACLIB,DISP=SHR
//SYSGO      DD   UNIT=SYSDA,DISP=(,PASS),DCB=(BLKSIZE=400,        X
//                RECFM=FB,LRECL=80),SPACE=(80,(100,100),RLSE)
//SYSPRINT   DD   SYSOUT=A,DCB=(LRECL=121,RECFM=FBM,BLKSIZE=605,   X
//                SPACE=(121,(500,500),RLSE,,ROUND)
//SYSUT1     DD   UNIT=SYSDA,DISP=(,DELETE),SPACE=(1700,(100,50))
//SYSUT2     DD   UNIT=SYSDA,DISP=(,DELETE),SPACE=(1700,(100,50))
//SYSUT3     DD   UNIT=(SYSDA,SEP=(SYSLIB,SYSUT1,SYSUT2)),         X
//                SPACE=(1700,(100,50))
//L      EXEC     PGM=DFSILNK0,PARM='XREF,LIST',COND=(0,LT,C),     X
//                REGION=100K
//SYSLIN     DD   DSNAME=*.C.SYSGO,DISP=(OLD,DELETE)
//           DD   DDNAME=SYSIN
//SYSPRINT   DD   SYSOUT=A,DCB=(LRECL=121,RECFM=FBA,BLKSIZE=605), X
//                SPACE=(121,(100,100),RLSE)
//SYSLMOD    DD   DSNAME=IMS.DBDLIB(&MBR),DISP=SHR
//SYSUT1     DD   UNIT=(SYSDA,SEP=(SYSLMOD,SYSLIN)),               X
//                DISP=(,DELETE),SPACE=(1024,(100,10),RLSE)
```

MEMBER NAME IMSCOBOL

```
//       PROC    MBR=,PAGES=60
//C EXEC PGM=IEQCBL00,PARM='SIZE=110000,LINECNT=50',REGION=126K
//SYSLIN DD DSNAME=&&LIN,DISP=(MOD,PASS),UNIT=SYSDA,             X
//              DCB=(LRECL=80,RECFM=FB,BLKSIZE=400),             X
//              SPACE=(CYL,(4,1),RLSE)
//SYSPRINT DD   SYSOUT=A,DCB=(RECFM=FBA,LRECL=121,BLKSIZE=605),  X
//              SPACE=(605,(&PAGES.0,&PAGES),RLSE,,ROUND)
//SYSUT1 DD UNIT=SYSDA,DISP=(NEW,DELETE),SPACE=(CYL,(10,1),RLSE)
//SYSUT2 DD UNIT=SYSDA,DISP=(NEW,DELETE),SPACE=(CYL,(10,1),RLSE)
//SYSUT3 DD UNIT=SYSDA,DISP=(NEW,DELETE),SPACE=(CYL,(10,1),RLSE)
//SYSUT4 DD UNIT=SYSDA,DISP=(NEW,DELETE),SPACE=(CYL,(10,1),RLSE)
//L EXEC PGM=DFSILNK0,REGION=100K,PARM='XREF,LIST,LET',          X
//              COND=(4,LT,C)
//SYSLIB DD DSNAME=SYS1.COBLIB,DISP=SHR
// DD DSNAME=SYS1.PL1LIB,DISP=SHR
//SYSOBJ DD DSNAME=IMS.RESLIB,DISP=SHR  NOTE
//SYSLIN DD DSNAME=&&LIN,DISP=(OLD,DELETE)
//          DD  DSNAME=IMS.PROCLIB(DLITCBL),DISP=SHR
// DD DDNAME=SYSIN
//SYSLMOD DD DSNAME=IMS.PGMLIB(&MBR),DISP=SHR
//SYSPRINT DD  SYSOUT=A,DCB=(RECFM=FBA,LRECL=121,BLKSIZE=605),   X
//              SPACE=(605,(&PAGES.0,&PAGES),RLSE,,ROUND)
//SYSUT1 DD UNIT=SYSDA,DISP=(NEW,DELETE),SPACE=(CYL,(10,1),RLSE)
```

Assumes:

1. User supplies source data from SYSIN.
2. Output is Class A.
3. MBR=NAME, when name is load module name for program.
4. SYSDA is generic device name.
5. RESLIB is cataloged.

MEMBER NAME IMSPLI

```
//       PROC    MBR=,PAGES=50
//C EXEC  PGM=IEMAA,PARM='XREF,ATR,LOAD,NODECK,NOMACRO,OPT=1',   X
//              REGION=114K
//SYSUT1    DD  UNIT=SYSDA,SPACE=(1024,(60,60),RLSE,,ROUND),     X
//              DCB=BLKSIZE=1024,DISP=(NEW,PASS)
//SYSUT3    DD  UNIT=SYSDA,SPACE=(1024,(60,60),RLSE,,ROUND),     X
//              DCB=BLKSIZE=1024,DISP=(NEW,PASS)
//SYSPRINT DD   SYSOUT=A,DCB=(LRECL=125,BLKSIZE=629,RECFM=VBA),  X
//              SPACE=(605,(&PAGES.0,&PAGES),RLSE)
//SYSLIN    DD  UNIT=SYSDA,SPACE=(80,(250,80),RLSE),             X
//              DCB=BLKSIZE=80,DISP=(NEW,PASS)
//L      EXEC   PGM=DFSILNK0,PARM='XREF,LIST,LET',COND=(4,LT,C),X
//              REGION=100K
//SYSLIB    DD  DSNAME=SYS1.PL1LIB,DISP=SHR
//          DD  DSNAME=SYS1.COBLIB,DISP=SHR
//SYSLIN    DD  DSNAME=*.C.SYSLIN,DISP=(OLD,DELETE)
//          DD  DSNAME=IMS.PROCLIB(DLITPLI),DISP=SHR
//          DD  DDNAME=SYSIN
//SYSLMOD   DD  DSNAME=IMS.PGMLIB(&MBR),DISP=SHR
//SYSPRINT  DD  SYSOUT=A,DCB=(LRECL=121,BLKSIZE=605,RECFM=FBA),  X
//              SPACE=(605,(&PAGES.0,&PAGES),RLSE)
//SYSOBJ    DD  DSNAME=IMS.RESLIB,DISP=SHR
//SYSUT1    DD  UNIT=SYSDA,DISP=(NEW,DELETE),                    X
//              SPACE=(CYL,(5,1),RLSE)
```

Same assumptions as IMSCOBOL

MEMBER NAME DLIBATCH

```
//      PROC    PSB=TEMPNAME
//G         EXEC PGM=DFSIRC00,PARM='3,&PSB',REGION=120K
//IMS       DD  DSNAME=IMS.PSBLIB,DISP=SHR
//          DD  DSNAME=IMS.DBDLIB,DISP=SHR
//SYSUDUMP  DD  SYSOUT=A,SPACE=(605,(500,500),RLSE,,ROUND),      X
//              DCB=(RECFM=FBA,LRECL=121,BLKSIZE=605)
```

Assume that user must append DD cards for data sets representing Data Language/I data bases.

MEMBER NAME IMSCOBGO

```
//      PROC    MBR=,PAGES=60
//C     EXEC    PGM=IEQCBL00,                                    X
//              PARM='LINECNT=50,SIZE=110000',REGION=126K
//SYSIN     DD  DSNAME=&&LIN,DISP=(MOD,PASS),UNIT=SYSDA,         X
//              DCB=(LRECL=80,RECFM=FB,BLKSIZE=400),             X
//              SPACE=(CYL,(4,1),RLSE)
//SYSPRINT DD   SYSOUT=A,DCB=(LRECL=121,RECFM=FBA,BLKSIZE=605), X
//              SPACE=(605,(&PAGES.0,&PAGES),RLSE,,ROUND)
//SYSUT1 DD UNIT=SYSDA,DISP=(NEW,DELETE),SPACE=(CYL,(10,1),RLSE)
//SYSUT2 DD UNIT=SYSDA,DISP=(NEW,DELETE),SPACE=(CYL,(10,1),RLSE)
//SYSUT3 DD UNIT=SYSDA,DISP=(NEW,DELETE),SPACE=(CYL,(10,1),RLSE)
//SYSUT4 DD UNIT=SYSDA,DISP=(NEW,DELETE),SPACE=(CYL,(10,1),RLSE)
//L EXEC PGM=DFSILNK0,REGION=100K,PARM='XREF,LIST,LET',          X
//              COND=(4,LT,C)
//SYSLIB DD DSNAME=SYS1,COBLIB,DISP=SHR
//          DD  DSNAME=SYS1.PL1LIB,DISP=SHR
//SYSOBJ DD DSNAME=IMS.RESLIB,DISP=SHR  NOTE 1
//SYSLIN DD DSNAME=&&LIN,DISP=(OLD,DELETE)
//          DD  DSNAME=IMS.PROCLIB(DLITCBL),DISP=SHR
//          DD  DSNAME=SYSIN
//SYSLMOD   DD  DSNAME=IMS.PGMLIB(&MBR),DISP=SHR
//SYSPRINT  DD  SYSOUT=A,DCB=(LRECL=121,RECFM=FBA,BLKSIZE=605), X
//              SPACE=(605,&PAGES.0,RLSE,,ROUND)
//SYSUT1 DD UNIT=SYSDA,DISP=(NEW,DELETE),SPACE=(CYL,(10,1),RLSE)
//G  EXEC  PGM=DFSIRC00,PARM='3,&MBR',REGION=150K,TIME=2,        X
//              COND=(0,LT)
//IMS       DD  DSNAME=IMS.PSBLIB,DISP=SHR
//          DD  DSNAME=IMS.DBDLIB,DISP=SHR
//SYSOUT DD SYSOUT=A,SPACE=(CYL,(1,1)),DCB=(LRECL=133,RECFM=FA)
//SYSUDUMP DD SYSOUT=A,DCB=(LRECL=121,RECFM=FBA,BLKSIZE=3025),  X
//              SPACE=(3025,(200,100),RLSE,,ROUND)
```

Assumes:

1. User supplies source data from SYSIN.
2. Output is Class A.
3. MBR=NAME, when name is load module name for program.
4. User must supply a G.STEPLIB card for IMS.RESLIB and IMS.PROGLIB with DISP=(SHR,PASS).
5. SYSDA is generic device name.
6. RESLIB is cataloged.
7. User must append DD cards in execute set for data sets representing Data Language/I data bases.
8. Execution time limit of two minutes is specified.

MEMBER NAME IMSPLIGO

```
//      PROC    MBR=NAME,PAGES=50
//C EXEC  PGM=IEMAA,PARM='XREF,ATR,LOAD,NODECK,NOMACRO,OPT=1',  X
//              REGION=114K
//SYSUT1    DD  UNIT=SYSDA,SPACE=(1024,(60,60),RLSE,,ROUND),    X
//              DCB=BLKSIZE=1024,DISP=(NEW,PASS)
//SYSUT3    DD  UNIT=SYSDA,SPACE=(1024,(60,60),RLSE,,ROUND),    X
//              DCB=BLKSIZE=1024,DISP=(NEW,PASS)
//SYSPRINT  DD  SYSOUT=A,DCB=(LRECL=125,BLKSIZE=629,RECFM=VBA), X
//              SPACE=(605,(&PAGES.0,&PAGES),RLSE)
//SYSLIN DD     UNIT=SYSDA,SPACE=(80,(250,80),RLSE),            X
//              DCB=BLKSIZE=80,DISP=(NEW,PASS)
//L     EXEC    PGM=DFSILNK0,PARM='XREF,LIST,LET',COND=(4,LT,C),X
//              REGION=100K
//SYSLIB    DD  DSNAME=SYS1.PL1LIB,DISP=SHR
//          DD  DSNAME=SYS1.COBLIB,DISP=SHR
//SYSLIN    DD  DSNAME=*.C.SYSLIN,DISP=(OLD,DELETE)
//          DD  DSNAME=IMS.PROCLIB(DLITPLI),DISP=SHR
//      DD      DDNAME=SYSIN
//SYSLMOD DD    DSNAME=IMS.PGMLIB(&MBR),DISP=SHR
//SYSPRINT DD   SYSOUT=A,DCB=(LRECL=121,BLKSIZE=605,RECFM=FBA), X
//              SPACE=(605,(&PAGES.0,&PAGES),RLSE)
//SYSOBJ    DD  DSNAME=IMS.RESLIB,DISP=SHR
//SYSUT1    DD  UNIT=SYSDA,DISP=(NEW,DELETE),                   X
//              SPACE=(CYL,(5,1),RLSE)
//G     EXEC    PGM=DFSIRC00,PARM='3,&MBR',COND=(4,LT),         X
//              TIME=5,REGION=150K
//IMS       DD  DSNAME=IMS.PSBLIB,DISP=SHR
//          DD  DSNAME=IMS.DBDLIB,DISP=SHR
//SYSPRINT DD   SYSOUT=A,DCB=(LRECL=121,BLKSIZE=605,RECFM=FBA), X
//              SPACE=(605,(500,500),RLSE,,ROUND)
//SYSUDUMP  DD  SYSOUT=A,DCB=(LRECL=121,BLKSIZE=605,RECFM=FBA), X
//              SPACE=(605,(500,500),RLSE,,ROUND)
```

Same assumptions as IMSCOBGO

MEMBER NAME IMS

An example of the IMS procedure is given later in this chapter under "Type 0 Region".

MEMBER NAME IMS0

```
//IMS0  JOB     1,IMS,PRTY=14,MSGLEVEL=1
//NUCLEUS   EXEC PGM=DFSIRC00,                                  X
//          PARM='00DFSINUC0019010010010020'                    X
//          .     ABCCCCCCCCDDDEEEFFFGGGHHH     PARM            X
//          REGION TYPE=0                       A               X
//          DMBSDYNAMIC=0, RESIDENT=1           B               X
//          NUCLEUS MEMBER NAME                 CCCCCCCC        X
//          NUMBER OF QCR BUFFERS(CALCULATED)   DDD             X
//          NUMBER OF MSG BUFFERS(CALCULATED)   EEE             X
//          PSB POOL IN 1K BLOCKS(DEFAULT)      FFF             X
//          DMB POOL IN 1K BLOCKS(DEFAULT)      GGG             X
//          OSAM & TP POOL SIZE(DEFAULT)        HHH
//IMS   DD  DSNAME=IMS.PSBLIB,DISP=SHR
//      DD  DSNAME=IMS.DBDLIB,DISP=SHR
//STEPLIB   DD  DSNAME=IMS.RESLIB,DISP=SHR
//SYSUDUMP DD   SYSOUT=A,DCB=(LRECL=125,RECFM=VBA,              X
//              BLKSIZE=3129),SPACE=(125,(3000,3000),RLSE,,ROUND)
//INQCR     DD  DSNAME=IMS.IQCRDSET,DISP=OLD
//INMSG     DD  DSNAME=IMS.IMSGDSET,DISP=OLD
//OUTQCR    DD  DSNAME=IMS.OQCRDSET,DISP=OLD
//OUTMSG    DD  DSNAME=IMS.OMSGDSET,DISP=OLD
//IMSLOG    DD  DSNAME=IMSLOG,DISP=(,KEEP),                     X
//              DCB=(RECFM=V,BLKSIZE=1408,                      X
//              LRECL=1400,BUFNO=1),VOL=(,,,10),                X
//              UNIT=(2400,,DEFER)
//IMSLOGR   DD  DSNAME=IMSLOG,DISP=OLD,                         X
//              VOLUME=SER=000000,                              X
//              UNIT=(2400,,DEFER)
//DBDUMP    DD  DSNAME=DFSIDUMP,DISP=(NEW,KEEP),                X
//              UNIT=AFF=IMSLOGR
//              FOLLOWING WILL BE TP DEVICE ALLOCATION AS       X
//              SPECIFIED DURING SYSTEM DEFINITION BY USER.     X
//              USER MUST SUPPLY APPLICATION DATA BASE JCL,     X
//              NONE WILL BE GENERATED
```

MEMBER NAME IMS1

```
//NUCLEUS    EXEC PGM=DFSIRC00,REGION=170K,TIME=1440,               X
//                PARM='00DFSINUC0019010010010020'                  X
//                      ABCCCCCCCCDDDEEEFFFGGGHHH     PARM FLD      X
//                REGION TYPE = 0            A
//                DMBSDYNAMIC=0, RESIDENT=1  B
//                NUCLEUS MBR NAME                    CCCCCCCC
//                NUMBER OF QCR BUFFERS(CALCULATED)   DDD           X
//                NUMBER OF MSG BUFFERS(CALCULATED)   EEE           X
//                PSB POOL IN 1K BLOCKS(DEFAULT)      FFF           X
//                DMB POOL IN 1K BLOCKS(DEFAULT)      GGG           X
//                OSAM & TP POOL SIZE(DEFAULT)        HHH           X
//IMS        DD   DSNAME=IMS.PSBLIB,DISP=SHR
//           DD   DSNAME=IMS.DBDLIB,DISP=SHR
//STEPLIB    DD   DSNAME=IMS.RESLIB,DISP=SHR
//SYSUDUMP DD     SYSOUT=A,DCB=(LRECL=125,RECFM=VBA,                X
//                BLKSIZE=3129),SPACE=(125,(3000,3000),RLSE,,ROUND)
//INCR       DD   DSNAME=IMS.IQCRDSET,DISP=OLD
//INMSG      DD   DSNAME=IMS.IMSGDSET,DISP=OLD
//OUTQCR     DD   DSNAME=IMS.OQCRDSET,DISP=OLD
//OUTMSG     DD   DSNAME=IMS.OMSGDSET,DISP=OLD
//IMSLOG     DD   DSNAME=IMSLOG,DISP=(,KEEP),                       X
//                DCB=(RECFM=V,BLKSIZE=1408,                        X
//                LRECL=1400,BUFNO=1),VOL=(,,,10),                  X
//                UNIT=(2400,,DEFER)
//IMSLOGR    DD   DSNAME=IMSLOG,DISP=OLD,                           X
//                VOLUME=SER=000000,                                X
//                UNIT=(2400,,DEFER)
//DBDUMP     DD   DSNAME=DFSIDUMP,DISP=(NEW,KEEP),                  X
//                UNIT=AFF=IMSLOGR
//                FOLLOWING WILL BE TP DEVICE ALLOCATION AS         X
//                SPECIFIED DURING SYSTEM DEFINITION BY USER.       X
//                USER MUST SUPPLY APPLICATION DATA BASE JCL,       X
//                NONE WILL BE GENERATED
```

Assume that embedded STEPLIB allows only one step or first step only in JOB.

MEMBER NAME IMSMSG

```
//MESSAGE    JOB  1,IMS,MSGLEVEL=1
//G          EXEC    PGM=DFSIRC00,PARM=1,REGION=26K
//STEPLIB    DD   DSNAME=IMS.PGMLIB,DISP=SHR
//           DD   DSNAME=IMS.RESLIB,DISP=SHR
//IMS        DD   DSNAME=IMS.PSBLIB,DISP=SHR
//           DD   DSNAME=IMS.DBDLIB,DISP=SHR
//SYSUDUMP   DD   SYSOUT=A,DCB=(LRECL=125,RECFM=VBA,                X
//                BLKSIZE=3129),SPACE=(125,(2500,100),RLSE,,ROUND)
```

MEMBER NAME IMSBATCH

misleading

```
//      PROC      PSB=TEMPNAME
//G          EXEC PGM=DFSIRC00,PARM='2,&PSB',REGION=26K
//IMS        DD   DSNAME=IMS.PSBLIB,DISP=SHR
//           DD   DSNAME=IMS.DBDLIB,DISP=SHR
//SYSUDUMP   DD   SYSOUT=A,DCB=(LRECL=121,RECFM=VBA,                X
//                BLKSIZE=3129),SPACE=(125,(2500,100),RLSE,,ROUND)
```

MEMBER NAME DLITCBL

```
  INCLUDE  SYSOBJ(DFSILI00)
  ENTRY  DLITCBL
```

```
MEMBER NAME DLITPLI
  INCLUDE  SYSOBJ(DFSILI00)
  ENTRY IHESAPD
```

```
MEMBER NAME SECURITY
./       ADD    NAME=SECURITY
./       NUMBER NEW1=10,INCR=10
//       PROC  OPTN=UPDATE,IMS=',0',SOUT=A
//S      EXEC  PGM=DFSISMPO,PARM='&OPTN.&IMS.'
//STEPLIB DD    DSN=IMS.RESLIB,DISP=SHR
//       DD    DSN=IMS.PGMLIB,DISP=SHR
//SYSPRINT DD SYSOUT=&SOUT,DCB=(RECFM=VBA,BLKSIZE=400,BUFL=404
//SYSPUNCH DD  UNIT=SYSDA,SPACE=(80,(800,400),,,ROUND),                X
//             DCB=(RECFM=FB,LRECL=80,BLKSIZE=400),DISP=(,PASS)
//SYSLIN   DD  UNIT=SYSDA,SPACE=(TRK,(1,1)),DCB=(RECFM=F,BLKSIZE=80), X
//             DISP=(,PASS)
//SYSUT1   DD  UNIT=SYSDA,SPACE=(100,(400,400),,,ROUND),               X
//             DCB=(BLKSIZE=500,RECFM=FB)
//SYSUT2   DD  UNIT=(SYSDA,SEP=SYSUT1),SPACE=(100,(400,400),,,ROUND), X
//             DCB=*.S.SYSUT1
//SYSIN    DD  DSN=NO.SYSIN.DD.ASTERISK
//C      EXEC  PGM=IEUASM,PARM='LOAD,NODECK',COND=(12,LT,S),REGION=96K
//SYSPRINT DD  SYSOUT=&SOUT,DCB=(RECFM=FBM,LRECL=121,BLKSIZE=605)
//SYSGO    DD  UNIT=(SYSDA,SEP=SYSPRINT),DISP=(,PASS),                 X
//             DCB=*.S.SYSPUNCH,SPACE=(80,(400,400),,,ROUND)
//SYSUT1   DD  UNIT=SYSDA,SPACE=(CYL,(5,1))
//SYSUT2   DD  UNIT=SYSDA,SPACE=(CYL,(5,1))
//SYSUT3   DD  UNIT=(SYSDA,SEP=(SYSUT1,SYSUT2)),SPACE=(CYL,(5,1))
//SYSIN    DD  DSN=*.S.SYSPUNCH,DISP=(OLD,DELETE)
//L      EXEC  PGM=DFSILNKO,PARM='XREF,NE,OL',REGION=110K,COND=(4,LT,S)
//SYSPRINT DD  SYSOUT=&SOUT,DCB=(RECFM=FBA,LRECL=121,BLKSIZE=605)
//SYSLMOD  DD  DSN=IMS.RESLIB,DISP=SHR
//INPUT    DD  DSN=*.C.SYSGO,DISP=(OLD,DELETE)
//SYSUT1   DD  UNIT=(SYSDA,SEP=INPUT),SPACE=(CYL,(5,1))
//SYSLIN   DD  DSN=*.S.SYSLIN,DISP=(OLD,DELETE)
```

```
MEMBER NAME MFDBLOAD
//         PROC SOUT=A
//LOAD     EXEC PGM=DFSIRC00,PARM='3,DFSSAM01',REGION=110K
//STEPLIB  DD DSN=ICS.CLOD,DISP=SHR
//         DD DSNAME=ICS.CLOD,DISP=SHR
//IMS      DD DSNAME=ICS.PSBLIB,DISP=SHR
//         DD DSNAME=ICS.DBDLIB,DISP=SHR
//SYSUDUMP DD SYSOUT=&SOUT
//DI21PART DD DSNAME=IMS.DI21PART(PRIME),DISP=(,KEEP),DCB=DSORG=IS,     X
//            SPACE=(CYL,3,,CONTIG),VOL=SER=&PSER,UNIT=&PUNIT
//DI21PARO DD DSNAME=IMS.DI21PARO,DISP=(,KEEP),SPACE=(CYL,3,,CONTIG),   X
//            VOL=SER=&OSER,UNIT=&OUNIT
//SYSOUT   DD SYSOUT=&SOUT
//INPUT    DD DSNAME=ICS.BMAC(MFDFSYSN),DISP=SHR
```

```
MEMBER NAME MFDBDUMP
//         PROC SOUT=A
//DUMP     EXEC PGM=DFSIRC00,PARM='3,DFSSAM08',REGION=110K
//STEPLIB  DD DSN=ICS.CLOD,DISP=SHR
//         DD DSNAME=ICS.CLOD,DISP=SHR
//IMS      DD DSNAME=ICS.PSBLIB,DISP=SHR
//         DD DSNAME=ICS.DBDLIB,DISP=SHR
//SYSUDUMP DD SYSOUT=&SOUT
//DI21PART DD DSNAME=IMS.DI21PART,DISP=SHR
//DI21PARO DD DSNAME=IMS.DI21PARO,DISP=SHR
//OUTPUT   DD SYSOUT=&SOUT
```

Operating System/360 Link Pack Modules

Many of the Data Language/I modules, the OSAM modules, and the BISAM modules used by IMS/360 can be placed in the Operating System/360 RAM area (MFT-II) or link pack area (MVT). The following module list indicates those modules whose placement into RAM or link pack is recommended. The next section of this chapter describes the procedure that can be utilized to accomplish the placement of these modules in link pack at Operating System/360 IPL time. The modules to be included must previously exist in either the SYS1.SVCLIB or the SYS1.LINKLIB data set.

From SYS1.LINKLIB

Module Name	Module Definition
DFSIRC00	IMS/360 Region Controller
DFSIDLR0	Data Language/I HISAM Retrieve
DFSIDLH0	Data Language/I HSAM
DFSIDLI0	Data Language/I HISAM Insert
DFSIDLD0	Data Language/I HISAM Delete/Replace
DFSIOS20	OSAM Read/Write
DFSIOS30	OSAM Check
DFSIISM0	Data Language/I ISAM Simulator
DFSIWKN0	Data Language/I Write Key New Simulator

From SYS1.SVCLIB

Module Name	Module Definition
IGG019Z9*	OSAM Channel End Appendage
IGG019GX	BISAM Asynchronous Read/Write
IGG019G9	BISAM Appendage with Write Check
IGG019JV	BISAM Non-Privileged Macro-Time Read/Write
IGG019J7	BISAM Privileged Macro-Time Read/Write

* The last two characters of this module name are determined by the IMS/360 user during system definition.

OS/360 Link Pack Procedures

The following procedures should be utilized to place IMS/360 in MFT-II or MVT link pack. This procedure should be placed in SYS1.PROCLIB using the Operating System/360 utility program IEBUPDTE.

```
./        ADD         NAME=IEAIGG01,LIST=ALL
SYS1.LINKLIB      DFSIDLD0,                                    X
                  DFSIDLH0,                                    X
                  DFSIDLR0,                                    X
                  DFSIDLI0,                                    X
                  DFSIOS20,                                    X
                  DFSIOS30,                                    X
                  DFSIRC00,                                    X
                  DFSIWKN0,                                    X
                  DFSIISM0

./        ADD         NAME=IEAIGG02,LIST=ALL
SYS1.SVCLIB       IGG019GX,                                    X
                  IGG019J7,                                    X
                  IGG019G9,                                    X
                  IGG019JV,                                    X
                  IGG019Z9
```

The module IGG019Z9 is the OSAM channel end appendage, the last two characters of which are user-determined.

When Operating System/360 is IPLed and the system responds with:

```
SPECIFY SYSTEM PARAMETERS
```

the modules described in the two preceding procedures are placed in link pack if the response includes:

```
REPLY   00,'RAM=01,02'
```

and are completely user-dependent.

TYPES OF PROCESSING REGIONS - JCL

This chapter has described the procedures provided by IMS/360 system definition for execution of the various processing region types.

Region Type	Region Function	Procedure Name Used
0	IMS/360 Control Program	IMS0 or IMS1
1	IMS/360 Message Processing Programs	IMSMSG
2	IMS/360 Type 2 Batch Processing	IMSBATCH
3	IMS/360 Type 3 Batch Processing	DLIBATCH

The IMS and DLIBATCH procedures do not contain DD cards for data bases. These DD cards must be supplied, added to the procedure, by the IMS/360 user. The IMSBATCH and DLIBATCH procedures do not include DD cards for SYSIN and SYSOUT or other user-defined data sets. The necessary cards must also be added by the IMS/360 user.

Type 0 Region

IMS1 Procedure

To use the IMS1 procedure, the user should supply the following JCL:

```
//IMS    JOB   MSGLEVEL=1,PRIORITY=13
//JOBLIB DD   DSNAME=IMS.RESLIB,DISP=SHR
//       EXEC  IMS1,REGION=160K,TIME=1440,              X
//             PARM='ABCCCCCCCCDDDEEEFFFGGGHHH'
```

where:

A indicates region type is 0.

B indicates BTAM=0.

CCCCCCCC is IMS/360 control program nucleus member name.

DDD is number of QCR buffers.

EEE is number of message buffers.

FFF is PSB pool size in 1K blocks.

GGG is DMB pool size in 1K blocks.

HHH is OSAM and teleprocessing buffer pool size in 1K blocks.

The region size of 160K is an estimate based upon the size of the user's system. It is assumed that the IMS/360 modules are in the IMS.RESLIB data set.

IMS0 Procedure

The use of the IMS0 procedure does not require a user to supply JCL control cards to an Operating System/360 SYSIN job stream. The IMS0 procedure is invoked by the IMS/360 reader procedure.

The JCL procedure, IMS0, for an IMS/360 Type 0 region may be stored complete in a procedure library. To start the IMS/360 control program (Type 0 region), an operator can override the standard Operating System/360 start reader command as follows:

```
S RDR,2311,RESLIB,DSLIB,DSN=SYS1.PROCLIB(IMS0),DISP=SHR
```

It is more convenient to define a reader procedure that defaults to the IMS/360 job member. An example of such a reader procedure is:

```
//IEFPROC  EXEC   PGM=IEFIRC,     READER FIRST LOAD       C
//              REGION=48K,       READER BASIC REGION     C
//              PARM='00103005001024905010SYSDA   'DEFAULT C
//                   BPPTTT000MMMIIICCCRLSSSSSSSS PARM FLD C
//              DEFINED PROGRAMMER NAME AND     B         C
//              ACCOUNT NUMBER    NOT NEEDED              C
//              PRIORITY=01                     PP        C
//              JOB STEP INTERVAL=30 MINUTES    TTT       C
//              PRIMARY SYSOUT SPACE=50 TRACKS  OOO       C
//              SECONDARY SYSOUT SPACE=10 TRACKS MMM      C
//              READER/INTERPRETER DISPATCHING PRIORITY=249 C
//              JOB STEP DEFAULT REGION=50K  CCC          C
//              DISPLAY & EXECUTE COMMANDS=1    R         C
//              BYPASS LABEL=0                  L         C
//              SYSOUT UNIT NAME=SYSDA         SSSSSSSS   C
//IEFRDER  DD   UNIT=2311,                                C
//              VOLUME=SER=RESLIB,                        C
//              DCB=BUFNO=1,                              C
//              DSNAME=IMS.PROCLIB(IMS0),                 C
//              DISP=SHR
//IEFPDSI  DD   DSNAME=SYS1.PROCLIB,    PROCEDURE LIBRARY C
//              DISP=OLD
//         DD   DSNAME=IMS.PROCLIB,DISP=SHR
//IEFDATA  DD   UNIT=SYSDA,             SPOOL DEVICE      C
//              SPACE=(80,(500,500),RLSE,CONTIG), AMOUNT  C
//              DCB=(BUFNO=2,LRECL=80,BLKSIZE=80,         C
//              RECFM=FB,BUFL=80)
```

A reader procedure, IMS, is included as a part of the IMS/360 package. Using this procedure with the member name of IMS, the IMS/360 online JCL, IMS0, can be read into the Operating System/360 input queue using the command:

```
S IMS
```

The reader procedure, IMS, must be moved from the PDS library described on the PROCLIB card of system definition to the Operating System/360 SYS1.PROCLIB data set.

Type 1 Region

The use of the IMSMSG procedure is provided from the IMS/360 master terminal through the /START REGION command. For this reason, the procedure includes a JOB statement. If the user wishes to start message regions through a SYSIN stream with cards rather than with the /START REGION command, a new message region procedure should be established which does not contain a JOB statement. The /START REGION and /STOP REGION commands are detailed in the _IMS/360 Operations Manual, Volume II - Machine Operations_.

Type 2 Region

The procedure for Type 2 processing region execution is entitled IMSBATCH. To invoke this procedure, the following JCL is required:

```
//TYPE2    JOB     848, name, MSGLEVEL=1
//JOBLIB   DD      DSNAME=IMS.RESLIB,DISP=SHR
//         DD      DSNAME=IMS.PGMLIB,DISP=SHR
//         EXEC    IMSBATCH,                                X
//                 PARM=2,AAAAAAAA,BBBBBBBB,CCCCCCCC,DDDDDDDD
```

where:

2

is the type of processing region.

AAAAAAAA

is an application program name.

BBBBBBBB

is an optional parameter that allows the user to specify a PSB name different from the program name specified in parameter AAAAAAAA.

CCCCCCCC

is an input transaction code. Use of this parameter is required only if the Type 2 program intends to access messages of the specified transaction code from the input queue.

DDDDDDDD

is an output transaction code or logical terminal name. If this optional parameter is specified, it overrides the original output destination for all input messages that are processed by the Type 2 program specified in parameter CCCCCCCC. Even if no CCCCCCCC parameter is specified, the Type 2 program may output to the transaction code or logical terminal name specified in parameter DDDDDDDD.

The user may append DD cards to this procedure for any Operating System/360 data sets that do not represent IMS/360 data bases.

Type 3 Region

The procedure for Type 3 region execution is entitled DLIBATCH. The user must append to this procedure DD cards for the data sets that represent the physical storage of his data bases. The user may append DD cards to this procedure for any Operating System/360 data sets that do not represent IMS/360 data bases. The JCL required for invoking the DLIBATCH procedure is:

```
//DLIBATCH     JOB MSGLEVEL=1
//JOBLIB       DD      DSNAME=IMS.RESLIB,DISP=SHR
//             DD      DSNAME=IMS.PGMLIB,DISP=SHR
```

Where the application program and the PSB have same name:

```
//             EXEC    DLIBATCH,PARM='3,PSBNAME'
```

Where the application program has a different name than the PSB:

```
//             EXEC    DLIBATCH,PARM='3,PGMNAME,PSBNAME'
```

where:

PGMNAME equals the application program name.

PSBNAME equals the PSB name.

SYSTEM DEFINITION - TYPE 3 PROCESSING REGION

The system definition requirements for a Type 3 processing region are a subset of those required for the online IMS/360 system. The necessary information, including a flow of functions to be performed and the necessary JCL statements for system definition execution, appears elsewhere throughout this chapter.

SYSTEM DEFINITION ERROR CONDITIONS

The IMS/360 system definition error conditions are listed in Chapter 7.

SYSTEM DEFINITION EXAMPLES

There are two examples, one for a Type 0, 1, and 2 processing region, and the other for a Type 3 (batch stand-alone) processing region.

The Type 0, 1, and 2 processing region example assumes the following:

- Type 1 programming system being used in MVT
- Three application programs
- Ten transaction codes against those application programs
- Two data bases
- Line groups:

  nonswitched 1050 communication system with one terminal

  switched (dial) 1050 communication system with three terminals

  switched (dial) 2740 communication system with three terminals

  nonswitched 2740 communication systems with four terminals

See the example below for additional assumptions.

Teleprocessing Example

This example illustrates the output from Stage 1 of the IMS/360 system definition utility program. The input to Stage 1 (that is, the control cards) is provided in the output listing followed by a summary of the featgroup specifications, the application specifications, the communication specifications, and the data set specifications. Next are the punch statements, followed by the comments considered warnings.

If the user invokes the alternate IMS/360 system definition for an alternate IMS/360 nucleus, an example would appear as follows:

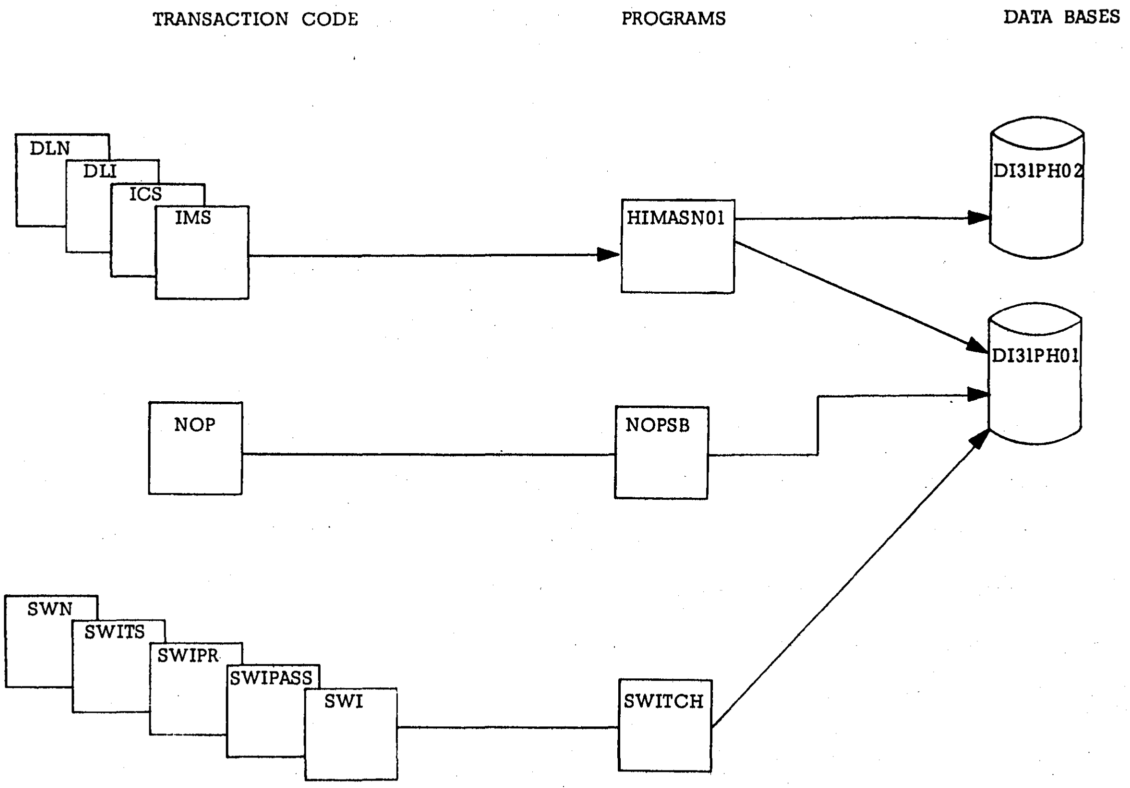
- The IMSTEST control card would precede all of the other system definition cards.

  ```
  IMSTEST GLIB=ICS.MACLIB,LLIB=ICS.LOAD,BLIB=ICS.BLKLIB,CODE=A
  ```

- Referring to the teleprocessing example that follows Figure 23, on Page 17 (upper right of page) is the start of the punch statements. Note that statement 749 starts Step 1 and there are 38 steps generated (through Page 53). The alternate system definition example of the punch statements would not have Steps 1 through 6, Steps 32 through 35, and Step 38. Step 7, statement 1287, would become Step 1 and all steps that follow would be renumbered consecutively.

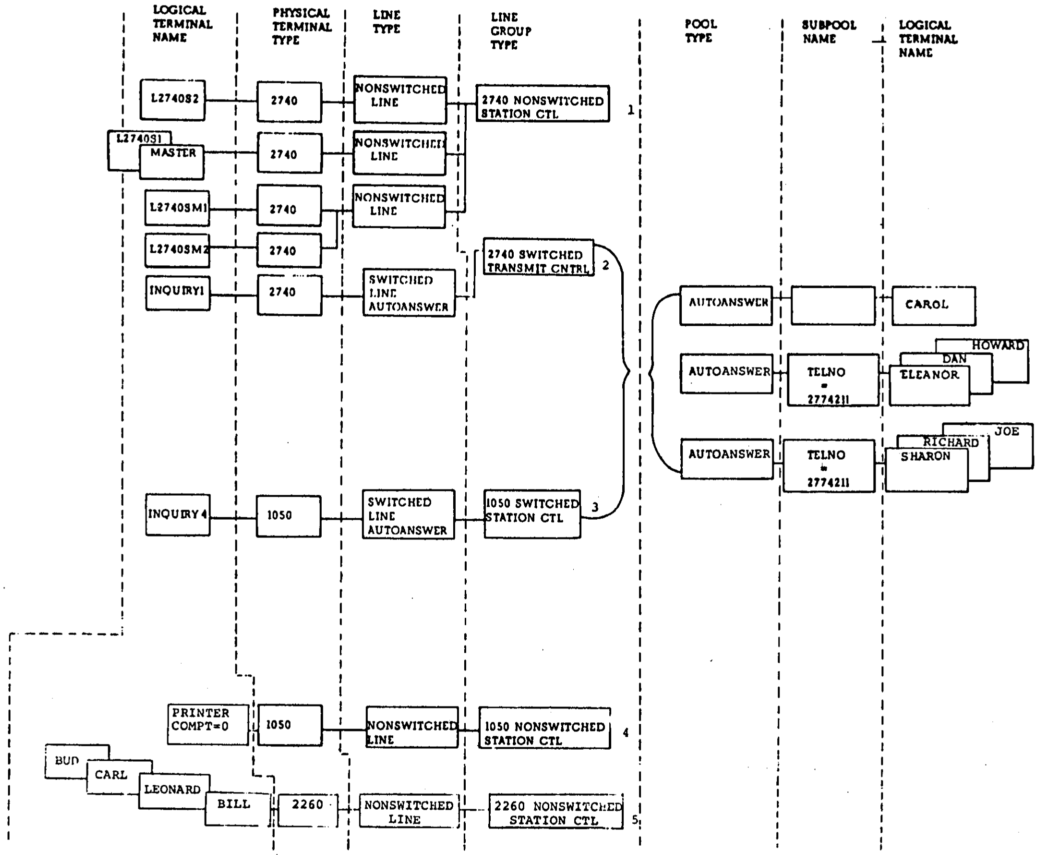
Figure 22 shows, in summary form, the various transaction codes, programs, and data bases, including their relationship to each other, as they exist in the following example of system definition.

Figure 23 shows, in summary form, the teleprocessing relationship as it exists in the following example of system definition. A review of the section titled IMS/360 Telecommunications Considerations, in Chapter 3, is recommended.



Note: A review of the section in Chapter 3 titled "IMS/360 Telecommunications Considerations" is recommended.

Figure 22. System definition example summary - transaction codes, programs, and data bases



Note: A review of the section in Chapter 3 titled "IMS/360 Telecommunications Considerations" is recommended.

Figure 23. System definition example summary - teleprocessing relationship

```
                                                                                    PAGE    1

LCC  CBJECT CCDE    ADDR1 ADDR2  STMT   SOURCE STATEMENT                            F30SEP69   2/12/70

                                   1     IMSCTRL         SYSTEM=(MVT,ALL),MAXIO=10,MAXREGN=3,MSGBUFF=10,      X
                                                   COMMSVC=(244,245),OSAMSVC=243,OCENDA=Z8,CKPT=500


                                   3                     *,     ALL IMS/360 FUNCTIONS ARE SELECTED
                                   4                     *,     MVT PROGRAMMING SYSTEM WILL BE USED
                                   5                     *,     3 REGIONS MAY BE OPERATED SIMULTANEOUSLY
                                   6                     *,     10 SUBTASKS MAY BE IN OPERATION TOGETHER
                                   7                     *,     10 TERMINALS MAY BE OPERATED SIMULTANEOUSLY
                                   8                     *,     OSAM CHANNEL END APPENDAGE - IGG019Z8
                                   9                     *,     CHECKPOINTS OCCUR AFTER FVERY 500 LOG ENTRIES
                                  10                     *,     COMMUNICATION -  ASK SVC NUMBER - 244
                                  11                     *,                    - REPLY SVC NUMBER - 245
                                  12                     *,     SUPERVISCR STATE SVC NUMBER - 243


                                   4    APPLCTN          PSB=DFSSAM02
                                  .5            DATABASE      DBD=DI21PART,INTENT=SHARE
                                  16              TRANSACT   CODE=CSPPN,MSGTYPE=(SNGLSEG,NONRESPONSE)
                                  17              TRANSACT   CODE=PART,MSGTYPE=(SNGLSEG,NONRESPONSE)
                                  18    APPLCTN          PSB=DFSSAM03
                                  19            DATABASE      CBD=CI21PART,INTENT=SHARE
                                  20              TRANSACT   CODE=DSPINV,MSGTYPE=(SNGLSEG,NONRESPONSE)
                                  21              TRANSACT   CODE=INVTORY,MSGTYPE=(SNGLSEG,NONRESPONSE)
                                  22    APPLCTN          PSB=DFSSAM07
                                  23            DATABASE      DBD=DI21PART,INTENT=SHARE
                                  24              TRANSACT   CODE=CSPALLI,MSGTYPE=(SNGLSEG,NONRESPONSE)
                                  25    APPLCTN          PSB=DFSSAM04
                                  26            DATABASE      DBD=DI21PART
                                  27              TRANSACT   CODE=ADDI,PRTY=(7,9,5)
                                  28              TRANSACT   CODE=ADDINV,PRTY=(7,9,5)
                                  29              TRANSACT   CODE=ADDPART,PRTY=(7,9,5)
                                  30              TRANSACT   CODE=ADDPN,PRTY=(7,9,5)
                                  31              TRANSACT   CODE=DLETI,PRTY=(5,7,2)
                                  32              TRANSACT   CODE=DLETINV,PRTY=(5,7,2)
                                  33              TRANSACT   CODE=DLETPART,PRTY=(5,7,2)
                                  34              TRANSACT   CODE=DLETPN,PRTY=(5,7,2)
                                  35    APPLCTN          PSB=DFSSAM05
                                  36            DATABASE      DBD=DI21PART
                                  37              TRANSACT   CODE=CLOSE,MSGTYPE=(SNGLSEG,NONRESPONSE)
                                  38              TRANSACT   CCDE=CLSORD,PRTY=(7,9,5)
                                  39    APPLCTN          PSB=UFSSAM06
                                  40            DATABASE      DBD=DI21PART
                                  41              TRANSACT   CCDE=CISB,PRTY=(9,10,2)
                                  42              TRANSACT   CODE=DISBURSE,MSGTYPE=(SNGLSEG,NONRESPONSE)
                                  43           APPLCTN PSB=DFSLKM00
                                  44           DATABASE CBD=DI31PH01,INTENT=SHARE
                                  45           TRANSACT CODE=DFS,PRTY=(5,12,5),PROCLIM=(8,100),                 C
                                                    INQUIRY=YES
                                  46      APPLCTN          PSB=HIMASN01
```

```
                                                                                    PAGE    2

LOC  OBJECT CODE    ADDR1 ADDR2  STMT   SOURCE STATEMENT                            F30SEP69   2/12/70

                                  47        DATABASF      DBD=DI31PH01
                                  48      DATABASE        CBD=DI31PH02
                                  49           TRANSACT  CODE=DLI,PRTY=(5,10,5),PROCLIM=(10,10),               ,
                                                  MSGTYPE=(SNGLSEG,RESPONSE)
                                  50           TRANSACT  CODE=ICS,PRTY=(5,12,5),PROCLIM=(10,100)
                                  51           TRANSACT  CODE=IMS,PRTY=(2,5,10),PROCLIM=(1,100),               X
                                                  MSGTYPE=(SNGLSEG,NONRESPCNSE)
                                  52           TRANSACT  CODE=DLN,PRTY=(0,8,3),PROCLIM=(10,100)
                                  53      APPLCTN         PSB=NOPSB
                                  54         DATABASE     DBD=DI31PH01,INTENT=SHARE
                                  55      TRANSACT        CODE=NOP,PRTY=(1,1,1),PROCLIM=(5,50),INQUIRY=YES
                                  56      APPLCTN         PSB=SWITCH
                                  57         DATABASE     DBD=DI31PH01,INTENT=SHARE
                                  58      TRANSACT        CODE=SWI,PRTY=(1,7,1000),PROCLIM=(5,1),              X
                                                  INQUIRY=YES       n,l,d
                                  59      TRANSACT        CODE=SWIBR,PRTY=(5,5,4),PROCLIM=(20,100),            X
                                                  INQUIRY=YES
                                  60      TRANSACT        CODE=SWIPASS,PRTY=(4,6,1),PROCLIM=(20,100),          X
                                                  INCUIRY=YES
                                  61      TRANSACT        CODE=SWIPR,PRTY=(14,14,100),PROCLIM=(20,100),        X
                                                  INQUIRY=YES
                                  62      TRANSACT        CODE=SWITS,PRTY=(4,6,1),PROCLIM=(20,100),            X
                                                  :NQUIRY=YES
                                  63      TRANSACT        CODE=SWN,PRTY=(0,4,4),PROCLIM=(5,100),INQUIRY=YES
                                  64            APPLCTN         PSB=HIMARJ01
                                  65              DATABASE      DBD=DI21IRJE
                                  66                TRANSACT   CODE=#,PRTY=(10,14,6)
                                  67                TRANSACT   CODE=RJE,PRTY=(2,4,10)
                                  68     \PPLCTN          PSB=HIMAJC01
                                  69       DATABASE       DBD=DS40JC01
                                  70         TRANSACT     CODE=TPPL1,PRTY=(8,8,65535)
                                  71     APPLCTN          PSB=HIMAJCC2
                                  72       DATABASE       DBD=DS40JC01
                                  73         TRANSACT     CODE=TPPL2,PRTY=(8,8,65535)
                                  74     APPLCTN          PSB=HIMAJC03
                                  75       DATABASE       DBD=DI31PH01
                                  76         TRANSACT     CODE=TUBE,PRTY=(8,8,65535)
                                  77           APPLCTN PSB=HIBLSK01,PGMTYPE=BATCH
                                  78           DATABASE CBD=DI31SK01
                                  79           DATABASE DBD=DI32SKC1
                                  80        TRANSACT CCDE=SW1
                                  81+** WARNING **
                                  82                  2,G312 PRIORITY VALUES FOR TRANSACTICN CCDES
                                  83                  *,     USED BY BATCH PROGRAMS MUST BE NULL;
                                  84                  *,     VALUES ARE RESET TO PRTY=(0,0,65535)
                                  85           APPLCTN PSB=HIBASK01,PGMTYPE=BATCH
                                  86           DATABASE CBD=DI31SK01
                                  87           DATABASE CBD=DI32SK01
                                  88        TRANSACT CODE=SW2,PRTY=(0,2,1000)
                                  89+** WARNING **
```

```
                                                                                                            PAGE    3

LCC  OBJECT CODE    ADDR1 ADDR2  STMT   SOURCE STATEMENT                                           F30SEP69   2/12/70

                                   90                2,G312 PRIORITY VALUES FOR TRANSACTICN CODES
                                   91                *,     USED BY BATCH PROGRAMS MUST BE NULL;
                                   92                *,     VALUES ARE RESET TO PRTY=(0,0,1000)
                                   93          APPLCTN PSB=HSBASK01,PGMTYPE=BATCH
                                   94          DATABASE CBD=DS31SK01
                                   95          APPLCTN PSB=EN00SK01,PGMTYPE=TP
                                   96          TRANSACT CODE=ENQ,PRTY=(8,8,65535)
                                   97          APPLCTN PSB=HITASK01,PGMTYPE=TP
                                   96          DATABASE CBD=DI31SK01
                                   99          TRANSACT CODE=SKI1,PRTY=(8,8,65535)
                                  100          APPLCTN PSB=HITASK02,PGMTYPE=TP
                                  101          DATABASE CBD=DI32SKC1
                                  102          TRANSACT CODE=SKI2,PRTY=(8,8,65535)
                                  103          APPLCTN PSB=HSTASK01,PGMTYPE=TP
                                  104          DATABASE CBD=DS31SK01
                                  105          TRANSACT CCDE=SKH1,PRTY=(8,8,65535)


                                  107   LINEGRP      DDNAME=DD2740S
                                  108     LINE          FEAT=POLL,ACDR=022
                                  109        TERMINAL   ACDR=E2
                                  110           NAME    L2740S2
                                  111     LINE          FEAT=POLL,ADDR=023
                                  112 DFSCTBMT TERMINAL ADDR=E2
                                  113           NAME    MASTER
                                  114           NAME    L2740S1
                                  115     LINE          FEAT=POLL,ACDR=024
                                  116        TERMINAL   ADDR=E2
                                  117           NAME    L2740SM1
                                  118        TERMINAL   ADDR=E4
                                  119           NAME    L2740SM2
                                  120    LINEGRP        DCNAME=DD2740A,FEAT=(TRANSCTL,SWITCHED),UNITYPE=2740
                                  121     LINE          FEAT=AUTCANS,ADDR=026
                                  122        TERMINAL   ACDR=E2
                                  123           NAME    INQUIRY1
                                  124    LINEGRP        DDNAME=DD1050A,FEAT=(STACTL,SWITCHED),UNITYPE=1050
                                  125     LINE          FEAT=AUTOANS,ADDR=027
                                  126        TERMINAL   ACDR=E2
                                  127           NAME    INQUIRY2
                                  128     POOL            FEAT=AUTOANS,ZONE=0
                                  129        SUBPOOL
                                  130           NAME    CAROL
                                  131       SUBPOOL
                                  132           NAME    ELEANOR
                                  133           NAME    DAN
                                  134           NAME    HOWARD
                                  135       SUBPOOL
                                  136           NAME    SHARRON
                                  137          NAME  RICHARD
                                  138          NAME  JOE
```

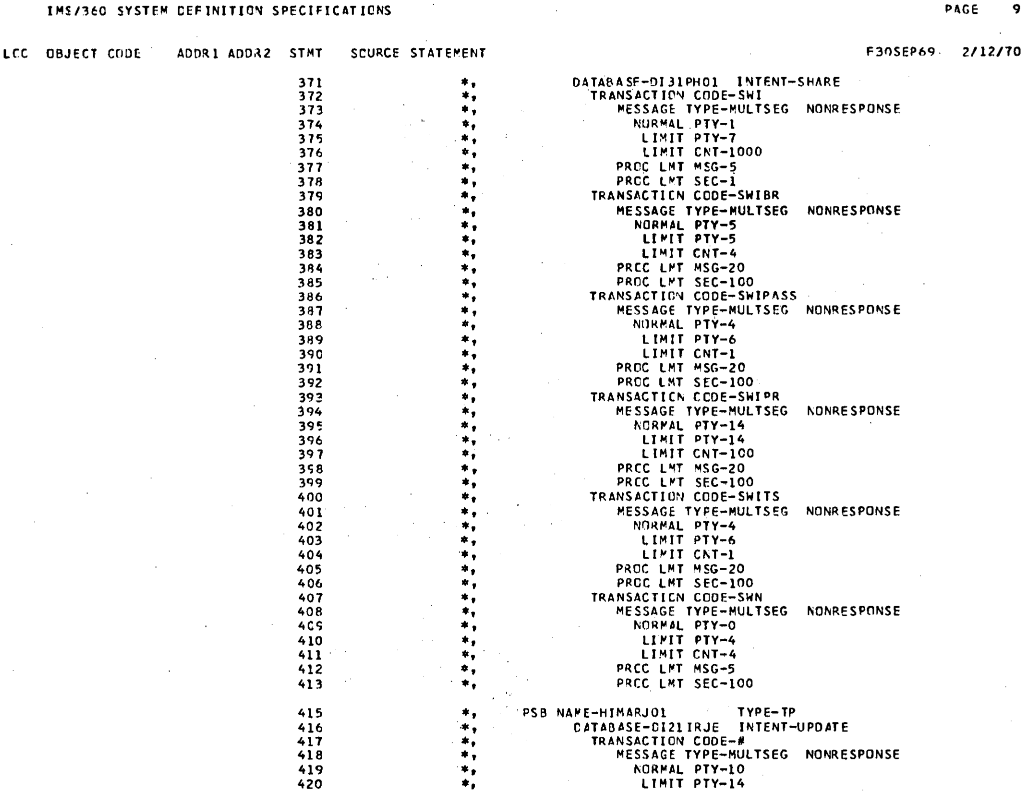
```
                                                                                                            PAGE    4

LCC  CBJECT CCDE    ADDR1 ACDR2  STMT   SOURCE STATEMENT                                           F30SEP69   2/12/70

                                  139   LINEGRP        DCNAME=DD1050,FEAT=(STACTL,NONSWITCH),UNITYPE=1050
                                  140    LINE           FEAT=POLL,ADDR=02A
                                  141        TERMINAL   ACDR=E2
                                  142          NAME  PRINTER,COMPT=FTR1
                                  143          NAME  T2780,COMPT=PTR1
                                  144          NAME  TAPEPNCH,COMPT=PTPCH
                                  145          NAME  MODEL2,COMPT=PTPCH
                                  146          NAME  CARDPNCH,COMPT=3
                                  147          NAME  MODEL2M,COMPT=3
                                  148   LINEGRP      DDNAME=DD2260T,UNITYPE=2260
                                  149     LINE       FEAT=POLL,ADDR=0A2
                                  150       TERMINAL ADDR=A0,UNIT=40
                                  151          NAME  BILL
                                  152       TERMINAL ADDR=A1,UNIT=4C
                                  153          NAME  LECNARD
                                  154          NAME  ERNE
                                  155       TERMINAL ADCR=A2,UNIT=40
                                  156          NAME  CARL
                                  157       TERMINAL ADDR=A3,UNIT=4C
                                  158          NAME  BUD

                                  160     MASTTERM          MASTER


                                  162    PSBLIB      PDS=ICS.PSBLIB,UNIT=2314,VOLNO=IMSLIB
                                  163    OBDLIB      PDS=ICS.OBDLIB,UNIT=2314,VOLNO=IMSLIB
                                  164    PGMLIB      PDS=ICS.CLOD,UNIT=2314,VOLNO=IMSLIB
                                  165   RESLIB       PDS=ICS.CLOD,UNIT=2314,VOLNO=IMSLIB
                                  166   MACLIB       PDS=ICS.BMAC,UNIT=2314,VOLNO=IMSLIB,COPY=ALL
                                  167    PROCLIB     PDS=ICS.PROCLIB,UNIT=2314,VCLNO=STORGE
                                  168     MSGQUEUE   QCPIN=(INQCR,ICS.IQCRDSET,2314,IMSDBS),REUSE=(YES,150), ,
                                                     MSGIN=(INMSG,ICS.IMSGDSET,2314,IMSDBS),                  ,
                                                     QCROUT=(OUTQCR,ICS.OCCRDSET,2314,IMSDBS),                ,
                                                     MSGCUT=(OUTMSG,ICS.OMSGDSET,2314,IMSDBS)


                                  170   IMSGEN       UT1SDS=TEMPSET,ASMPRT=ON,LEPRT=(XREF,LIST)
```

```
     IMS/360 SYSTEM DEFINITION SPECIFICATIONS                                                  PAGE    5

LOC  OBJECT CODE    ADDR1 ADDR2  STMT   SOURCE STATEMENT                                   F30SEP69   2/12/70
                                  172              *,FEATGRP SPECIFICATIONS

                                  174              *,      FEATGRP-
                                  175              *,      CONVERT(INPUT) -UC
                                  176              *,      CONVERT(OUTPUT)-UC


                                  179              *,APPLICATION SPECIFICATIONS

                                  181              *,    PSB NAME-DFSSAM02        TYPE-TP
                                  182              *,          DATABASE-DI21PART  INTENT-SHARE
                                  183              *,            TRANSACTION CODE-DSPPN
                                  184              *,               MESSAGE TYPE-SNGLSEG  NONRESPONSE
                                  185              *,                 NORMAL PTY-1
                                  186              *,                  LIMIT PTY-1
                                  187              *,                  LIMIT CNT-65535
                                  188              *,               PROC LMT MSG-65535
                                  189              *,               PROC LMT SEC-65535
                                  190              *,            TRANSACTION CODE-PART
                                  191              *,               MESSAGE TYPE-SNGLSEG  NONRESPONSE
                                  192              *,                 NORMAL PTY-1
                                  193              *,                  LIMIT PTY-1
                                  194              *,                  LIMIT CNT-65535
                                  195              *,               PROC LMT MSG-65535
                                  196              *,               PROC LMT SEC-65535

                                  198              *,    PSB NAME-DFSSAM03        TYPE-TP
                                  199              *,          DATABASE-DI21PART  INTENT-SHARE
                                  200              *,            TRANSACTION CODE-DSPINV
                                  201              *,               MESSAGE TYPE-SNGLSEG  NONRESPONSE
                                  202              *,                 NORMAL PTY-1
                                  203              *,                  LIMIT PTY-1
                                  204              *,                  LIMIT CNT-65535
                                  205              *,               PROC LMT MSG-65535
                                  206              *,               PROC LMT SEC-65535
                                  207              *,            TRANSACTION CODE-INVTORY
                                  208              *,               MESSAGE TYPE-SNGLSEG  NONRESPONSE
                                  209              *,                 NORMAL PTY-1
                                  210              *,                  LIMIT PTY-1
                                  211              *,                  LIMIT CNT-65535
                                  212              *,               PROC LMT MSG-65535
                                  213              *,               PROC LMT SEC-65535

                                  215              *,    PSB NAME-DFSSAM07        TYPE-TP
                                  216              *,          DATABASE-DI21PART  INTENT-SHARE
                                  217              *,            TRANSACTION CODE-DSPALLI
                                  218              *,               MESSAGE TYPE-SNGLSEG  NONRESPONSE
                                  219              *,                 NORMAL PTY-1
                                  220              *,                  LIMIT PTY-1
```

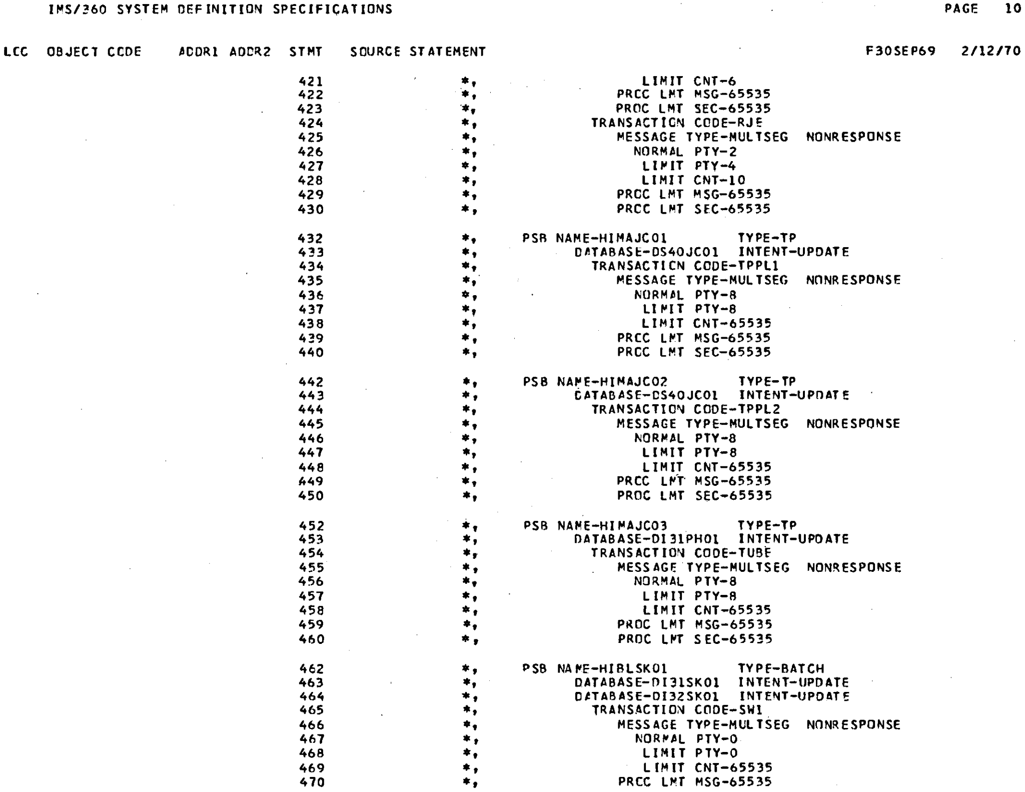
```
     IMS/360 SYSTEM DEFINITION SPECIFICATIONS                                                  PAGE    6

LOC  OBJECT CODE    ADDR1 ADDR2  STMT   SOURCE STATEMENT                                   F30SEP69   2/12/70
                                  221              *,                  LIMIT CNT-65535
                                  222              *,               PROC LMT MSG-65535
                                  223              *,               PROC LMT SEC-65535

                                  225              *,    PSB NAME-DFSSAM04        TYPE-TP
                                  226              *,          DATABASE-DI21PART  INTENT-UPDATE
                                  227              *,            TRANSACTION CODE-ADDI
                                  228              *,               MESSAGE TYPE-MULTSEG  NONRESPONSE
                                  229              *,                 NORMAL PTY-7
                                  230              *,                  LIMIT PTY-9
                                  231              *,                  LIMIT CNT-5
                                  232              *,               PROC LMT MSG-65535
                                  233              *,               PROC LMT SEC-65535
                                  234              *,            TRANSACTION CODE-ADDINV
                                  235              *,               MESSAGE TYPE-MULTSEG  NONRESPONSE
                                  236              *,                 NORMAL PTY-7
                                  237              *,                  LIMIT PTY-9
                                  238              *,                  LIMIT CNT-5
                                  239              *,               PROC LMT MSG-65535
                                  240              *,               PROC LMT SEC-65535
                                  241              *,            TRANSACTION CODE-ADDPART
                                  242              *,               MESSAGE TYPE-MULTSEG  NONRESPONSE
                                  243              *,                 NORMAL PTY-7
                                  244              *,                  LIMIT PTY-9
                                  245              *,                  LIMIT CNT-5
                                  246              *,               PROC LMT MSG-65535
                                  247              *,               PROC LMT SEC-65535
                                  248              *,            TRANSACTION CODE-ADDPN
                                  249              *,               MESSAGE TYPE-MULTSEG  NONRESPONSE
                                  250              *,                 NORMAL PTY-7
                                  251              *,                  LIMIT PTY-9
                                  252              *,                  LIMIT CNT-5
                                  253              *,               PROC LMT MSG-65535
                                  254              *,               PROC LMT SEC-65535
                                  255              *,            TRANSACTION CODE-DLETI
                                  256              *,               MESSAGE TYPE-MULTSEG  NONRESPONSE
                                  257              *,                 NORMAL PTY-5
                                  258              *,                  LIMIT PTY-7
                                  259              *,                  LIMIT CNT-2
                                  260              *,               PROC LMT MSG-65535
                                  261              *,               PROC LMT SEC-65535
                                  262              *,            TRANSACTION CODE-DLETINV
                                  263              *,               MESSAGE TYPE-MULTSEG  NONRESPONSE
                                  264              *,                 NORMAL PTY-5
                                  265              *,                  LIMIT PTY-7
                                  266              *,                  LIMIT CNT-2
                                  267              *,               PROC LMT MSG-65535
                                  268              *,               PROC LMT SEC-65535
                                  269              *,            TRANSACTION CODE-DLETPART
                                  270              *,               MESSAGE TYPE-MULTSEG  NONRESPONSE
```

```
                                 271            *,               NORMAL PTY-5
                                 272            *,                LIMIT PTY-7
                                 273            *,                LIMIT CNT-2
                                 274            *,             PROC LMT MSG-65535
                                 275            *,             PROC LMT SEC-65535
                                 276            *,          TRANSACTION CODE-DLETPN
                                 277            *,             MESSAGE TYPE-MULTSEG  NONRESPONSE
                                 278            *,               NORMAL PTY-5
                                 279            *,                LIMIT PTY-7
                                 280            *,                LIMIT CNT-2
                                 281            *,             PROC LMT MSG-65535
                                 282            *,             PROC LMT SEC-65535

                                 284            *,    PSB NAME-DFSSAM05        TYPE-TP
                                 285            *,          DATABASE-DI21PART  INTENT-UPDATE
                                 286            *,            TRANSACTION CODE-CLOSE
                                 287            *,               MESSAGE TYPE-SNGLSEG  NONRESPONSE
                                 288            *,                 NORMAL PTY-1
                                 289            *,                  LIMIT PTY-1
                                 290            *,                  LIMIT CNT-65535
                                 291            *,               PROC LMT MSG-65535
                                 292            *,               PROC LMT SEC-65535
                                 293            *,            TRANSACTION CODE-CLSORD
                                 294            *,               MESSAGE TYPE-MULTSEG  NONRESPONSE
                                 295            *,                 NORMAL PTY-7
                                 296            *,                  LIMIT PTY-9
                                 297            *,                  LIMIT CNT-5
                                 298            *,               PROC LMT MSG-65535
                                 299            *,               PROC LMT SEC-65535

                                 301            *,    PSB NAME-DFSSAM06        TYPE-TP
                                 302            *,          DATABASE-DI21PART  INTENT-UPDATE
                                 303            *,            TRANSACTION CODE-DISB
                                 304            *,               MESSAGE TYPE-MULTSEG  NONRESPONSE
                                 305            *,                 NORMAL PTY-9
                                 306            *,                  LIMIT PTY-10
                                 307            *,                  LIMIT CNT-2
                                 308            *,               PROC LMT MSG-65535
                                 309            *,               PROC LMT SEC-65535
                                 310            *,            TRANSACTION CODE-DISBURSE
                                 311            *,               MESSAGE TYPE-SNGLSEG  NONRESPONSE
                                 312            *,                 NORMAL PTY-1
                                 313            *,                  LIMIT PTY-1
                                 314            *,                  LIMIT CNT-65535
                                 315            *,               PROC LMT MSG-65535
                                 316            *,               PROC LMT SEC-65535

                                 318            *,    PSB NAME-DFSLKM00        TYPE-TP
                                 319            *,          DATABASE-DI31PH01  INTENT-SHARE
                                 320            *,            TRANSACTION CODE-DFS
```

```
                                 321            *,               MESSAGE TYPE-MULTSEG  NONRESPONSE
                                 322            *,                 NORMAL PTY-5
                                 323            *,                  LIMIT PTY-12
                                 324            *,                  LIMIT CNT-5
                                 325            *,               PROC LMT MSG-8
                                 326            *,               PROC LMT SEC-100

                                 328            *,    PSB NAME-HIMASN01        TYPE-TP
                                 329            *,          DATABASE-DI31PH01  INTENT-UPDATE
                                 330            *,          DATABASE-DI31PH02  INTENT-UPDATE
                                 331            *,            TRANSACTION CODE-DLI
                                 332            *,               MESSAGE TYPE-SNGLSEG  RESPONSE
                                 333            *,                 NORMAL PTY-5
                                 334            *,                  LIMIT PTY-10
                                 335            *,                  LIMIT CNT-5
                                 336            *,               PROC LMT MSG-10
                                 337            *,               PROC LMT SEC-10
                                 338            *,            TRANSACTION CODE-ICS
                                 339            *,               MESSAGE TYPE-MULTSEG  NONRESPONSE
                                 340            *,                 NORMAL PTY-5
                                 341            *,                  LIMIT PTY-12
                                 342            *,                  LIMIT CNT-5
                                 343            *,               PROC LMT MSG-10
                                 344            *,               PROC LMT SEC-100
                                 345            *,            TRANSACTION CODE-IMS
                                 346            *,               MESSAGE TYPE-SNGLSEG  NONRESPONSE
                                 347            *,                 NORMAL PTY-2
                                 348            *,                  LIMIT PTY-5
                                 349            *,                  LIMIT CNT-10
                                 350            *,               PROC LMT MSG-1
                                 351            *,               PROC LMT SEC-100
                                 352            *,            TRANSACTION CODE-DLN
                                 353            *,               MESSAGE TYPE-MULTSEG  NONRESPONSE
                                 354            *,                 NORMAL PTY-0
                                 355            *,                  LIMIT PTY-8
                                 356            *,                  LIMIT CNT-3
                                 357            *,               PROC LMT MSG-10
                                 358            *,               PROC LMT SEC-100

                                 360            *,    PSB NAME-NOPSB         TYPE-TP
                                 361            *,          DATABASE-DI31PH01  INTENT-SHARE
                                 362            *,            TRANSACTION CODE-NOP
                                 363            *,               MESSAGE TYPE-MULTSEG  NONRESPONSE
                                 364            *,                 NORMAL PTY-1
                                 365            *,                  LIMIT PTY-1
                                 366            *,                  LIMIT CNT-1
                                 367            *,               PROC LMT MSG-5
                                 368            *,               PROC LMT SEC-50

                                 370            *,    PSB NAME-SWITCH         TYPE-TP
```



```
     IMS/360 SYSTEM CEFINITION SPECIFICATIONS                                                                  PAGE    9

LCC  OBJECT CODE     ADDR1 ADDR2  STMT    SCURCE STATEMENT                                          F30SEP69   2/12/70

                                   371                  *,              DATABASE-DI31PH01  INTENT-SHARE
                                   372                  *,                TRANSACTION CODE-SWI
                                   373                  *,                   MESSAGE TYPE-MULTSEG  NONRESPONSE
                                   374                  *,                     NORMAL PTY-1
                                   375                  *,                      LIMIT PTY-7
                                   376                  *,                      LIMIT CNT-1000
                                   377                  *,                   PRCC LMT MSG-5
                                   378                  *,                   PRCC LMT SEC-1
                                   379                  *,                TRANSACTICN CODE-SWIBR
                                   380                  *,                   MESSAGE TYPE-MULTSEG  NONRESPONSE
                                   381                  *,                     NORMAL PTY-5
                                   382                  *,                      LIMIT PTY-5
                                   383                  *,                      LIMIT CNT-4
                                   384                  *,                   PRCC LMT MSG-20
                                   385                  *,                   PROC LMT SEC-100
                                   386                  *,                TRANSACTION CODE-SWIPASS
                                   387                  *,                   MESSAGE TYPE-MULTSEG  NONRESPONSE
                                   388                  *,                     NORMAL PTY-4
                                   389                  *,                      LIMIT PTY-6
                                   390                  *,                      LIMIT CNT-1
                                   391                  *,                   PROC LMT MSG-20
                                   392                  *,                   PROC LMT SEC-100
                                   393                  *,                TRANSACTICN CCDE-SWIPR
                                   394                  *,                   MESSAGE TYPE-MULTSEG  NONRESPONSE
                                   395                  *,                     NORMAL PTY-14
                                   396                  *,                      LIMIT PTY-14
                                   397                  *,                      LIMIT CNT-100
                                   398                  *,                   PRCC LMT MSG-20
                                   399                  *,                   PRCC LMT SEC-100
                                   400                  *,                TRANSACTION CODE-SWITS
                                   401                  *,                   MESSAGE TYPE-MULTSEG  NONRESPONSE
                                   402                  *,                     NORMAL PTY-4
                                   403                  *,                      LIMIT PTY-6
                                   404                  *,                      LIMIT CNT-1
                                   405                  *,                   PROC LMT MSG-20
                                   406                  *,                   PROC LMT SEC-100
                                   407                  *,                TRANSACTICN CODE-SWN
                                   408                  *,                   MESSAGE TYPE-MULTSEG  NONRESPONSE
                                   409                  *,                     NORMAL PTY-0
                                   410                  *,                      LIMIT PTY-4
                                   411                  *,                      LIMIT CNT-4
                                   412                  *,                   PRCC LMT MSG-5
                                   413                  *,                   PRCC LMT SEC-100

                                   415                  *,        PSB NAME-HIMARJ01         TYPE-TP
                                   416                  *,              DATABASE-DI21IRJE  INTENT-UPDATE
                                   417                  *,                TRANSACTION CODE-#
                                   418                  *,                   MESSAGE TYPE-MULTSEG  NONRESPONSE
                                   419                  *,                     NORMAL PTY-10
                                   420                  *,                      LIMIT PTY-14
```



```
     IMS/360 SYSTEM DEFINITION SPECIFICATIONS                                                                  PAGE   10

LCC  OBJECT CCDE     ADDR1 ADDR2  STMT    SOURCE STATEMENT                                          F30SEP69   2/12/70

                                   421                  *,                      LIMIT CNT-6
                                   422                  *,                   PRCC LMT MSG-65535
                                   423                  *,                   PROC LMT SEC-65535
                                   424                  *,                TRANSACTICN CODE-RJE
                                   425                  *,                   MESSAGE TYPE-MULTSEG  NONRESPONSE
                                   426                  *,                     NORMAL PTY-2
                                   427                  *,                      LIMIT PTY-4
                                   428                  *,                      LIMIT CNT-10
                                   429                  *,                   PRCC LMT MSG-65535
                                   430                  *,                   PRCC LMT SEC-65535

                                   432                  *,        PSB NAME-HIMAJC01         TYPE-TP
                                   433                  *,              DATABASE-DS40JC01  INTENT-UPDATE
                                   434                  *,                TRANSACTICN CODE-TPPL1
                                   435                  *,                   MESSAGE TYPE-MULTSEG  NONRESPONSE
                                   436                  *,                     NORMAL PTY-8
                                   437                  *,                      LIMIT PTY-8
                                   438                  *,                      LIMIT CNT-65535
                                   439                  *,                   PRCC LMT MSG-65535
                                   440                  *,                   PRCC LMT SEC-65535

                                   442                  *,        PSB NAME-HIMAJC02         TYPE-TP
                                   443                  *,              DATABASE-DS40JC01  INTENT-UPDATE
                                   444                  *,                TRANSACTION CODE-TPPL2
                                   445                  *,                   MESSAGE TYPE-MULTSEG  NONRESPONSE
                                   446                  *,                     NORMAL PTY-8
                                   447                  *,                      LIMIT PTY-8
                                   448                  *,                      LIMIT CNT-65535
                                   449                  *,                   PRCC LMT MSG-65535
                                   450                  *,                   PROC LMT SEC-65535

                                   452                  *,        PSB NAME-HIMAJC03         TYPE-TP
                                   453                  *,              DATABASE-DI31PH01  INTENT-UPDATE
                                   454                  *,                TRANSACTION CODE-TUBE
                                   455                  *,                   MESSAGE TYPE-MULTSEG  NONRESPONSE
                                   456                  *,                     NORMAL PTY-8
                                   457                  *,                      LIMIT PTY-8
                                   458                  *,                      LIMIT CNT-65535
                                   459                  *,                   PROC LMT MSG-65535
                                   460                  *,                   PROC LMT SEC-65535

                                   462                  *,        PSB NAME-HIBLSK01         TYPE-BATCH
                                   463                  *,              DATABASE-DI31SK01  INTENT-UPDATE
                                   464                  *,              DATABASE-DI32SK01  INTENT-UPDATE
                                   465                  *,                TRANSACTION CODE-SW1
                                   466                  *,                   MESSAGE TYPE-MULTSEG  NONRESPONSE
                                   467                  *,                     NORMAL PTY-0
                                   468                  *,                      LIMIT PTY-0
                                   469                  *,                      LIMIT CNT-65535
                                   470                  *,                   PRCC LMT MSG-65535
```

```
 471              *,                PROC LMT SEC-65535

 473              *,     PSB NAME-HIBASK01        TYPE-BATCH
 474              *,           DATABASE-DI31SK01  INTENT-UPDATE
 475              *,           DATABASE-DI32SK01  INTENT-UPDATE
 476              *,             TRANSACTION CODE-SW2
 477              *,                MESSAGE TYPE-MULTSEG  NONRESPONSE
 478              *,                  NORMAL PTY-0
 479              *,                   LIMIT PTY-0
 480              *,                   LIMIT CNT-1000
 481              *,                PROC LMT MSG-65535
 482              *,                PROC LMT SEC-65535

 484              *,     PSB NAME-HSBASK01        TYPE-BATCH
 485              *,           DATABASE-DS31SK01  INTENT-UPDATE
 486*** WARNING **
 487              2,G048 NO TRANSACT SPECIFICATIONS FOR PSB-HSBASK01

 489              *,     PSB NAME-ENQOSK01        TYPE-TP
 490*** WARNING **
 491              2,G047 NO DATABASE SPECIFICATIONS FOR PSB-ENQOSK01
 492              *,             TRANSACTION CODE-ENQ
 493              *,                MESSAGE TYPE-MULTSEG  NONRESPONSE
 494              *,                  NORMAL PTY-8
 495              *,                   LIMIT PTY-8
 496              *,                   LIMIT CNT-65535
 497              *,                PROC LMT MSG-65535
 498              *,                PROC LMT SEC-65535

 500              *,     PSB NAME-HITASK01        TYPE-TP
 501              *,           DATABASE-DI31SK01  INTENT-UPDATE
 502              *,             TRANSACTION CODE-SKI1
 503              *,                MESSAGE TYPE-MULTSEG  NONRESPONSE
 504              *,                  NORMAL PTY-8
 505              *,                   LIMIT PTY-8
 506              *,                   LIMIT CNT-65535
 507              *,                PROC LMT MSG-65535
 508              *,                PROC LMT SEC-65535

 510              *,     PSB NAME-HITASK02        TYPE-TP
 511              *,           DATABASE-DI32SK01  INTENT-UPDATE
 512              *,             TRANSACTION CODE-SKI2
 513              *,                MESSAGE TYPE-MULTSEG  NONRESPONSE
 514              *,                  NORMAL PTY-8
 515              *,                   LIMIT PTY-8
 516              *,                   LIMIT CNT-65535
 517              *,                PROC LMT MSG-65535
 518              *,                PROC LMT SEC-65535

 520              *,     PSB NAME-HSTASK01        TYPE-TP
```

```
 521              *,           DATABASE-DS31SK01  INTENT-UPDATE
 522              *,             TRANSACTION CODE-SKH1
 523              *,                MESSAGE TYPE-MULTSEG  NONRESPONSE
 524              *,                  NORMAL PTY-8
 525              *,                   LIMIT PTY-8
 526              *,                   LIMIT CNT-65535
 527              *,                PROC LMT MSG-65535
 528              *,                PROC LMT SEC-65535
```

```
     IMS/360 SYSTEM CEFINITION SPECIFICATIONS                                                                  PAGE   13

LOC   OBJECT CCDE     ADDR1 ADDR2  STMT   SOURCE STATEMENT                                                   F30SEP69   2/12/70

                                    531                *,        ****************************************
                                    532                *,        *  UPDATE ACTIVITY WILL BE LOGGED
                                    533                *,        *    FOR THE FOLLOWING DATABASES:
                                    534                *,        *
                                    535                *,        *             DI21PART
                                    536                *,        *             DI31PH01
                                    537                *,        *             DI31PH02
                                    538                *,        *             DI21IRJE
                                    539                *,        *             DS40JC01
                                    540                *,        *             DI31SK01
                                    541                *,        *             DI32SK01
                                    542                *,        *             DS31SK01
                                    543                *,        ****************************************

                                    545                *,        ****************************************
                                    546                *,        *    THE FCLLOWING TRANSACTION CODES
                                    547                *,        *      WILL NOT BE REPROCESSED BY
                                    548                *,        *          DATABASE RECCVERY:
                                    549                *,        *
                                    550                *,        *             DFS
                                    551                *,        *             NCP
                                    552                *,        *             SWI
                                    553                *,        *             SWIBR
                                    554                *,        *             SWIPASS
                                    555                *,        *             SWIPR
                                    556                *,        *             SWITS
                                    557                *,        *             SWN
                                    558                *,        ****************************************
```

```
     IMS/360 SYSTEM CEFINITION SPECIFICATIONS                                                                  PAGE   14

LCC   CBJECT CCDE     ADCRI ADDR2  STMT   SOURCE STATEMENT                                                   F30SEP69   2/12/70

                                    560                *,COMMUNICATICN SPECIFICATIONS

                                    562                *,       LINE-1   SYSTEM/360 OPERATOR'S CONSCLE

                                    564                *,            TERMINAL-0 ADDR-N/A FEATGRP-N/A
                                    565                *,               LOGICAL NAME-WTCR  FEATGRP-N/A

                                    567                *,  LINEGRP-1  DDNAME-DD2740S  FEAT-STACTL NONSWITCH
                                    568                *,               UNITYPE-2740

                                    570                *,       LINE-2   FEAT-PCLL    ADDR-022

                                    572                *,            TERMINAL-1 ADDR=E2 FEATGRP-STANDARD
                                    573                *,               LCGICAL NAME-L2740S2  FEATGRP-STANDARD

                                    575                *,       LINE-3   FEAT-PCLL    ADDR-023

                                    577                *,            TERMINAL-2 ADDR=E2 FEATGRP-STANDARD
                                    578                *,               LOGICAL NAME-MASTER  FEATGRP-STANDARD
                                    579                *,                 ******************************
                                    580                *,                 *       MASTER TERMINAL      *
                                    581                *,                 ******************************
                                    582                *,               LOGICAL NAME-L2740S1  FEATGRP-STANDARD

                                    584                *,       LINE-4   FEAT-PCLL    ADDR-024

                                    586                *,            TERMINAL-3 ADDR=E2 FEATGRP-STANDARD
                                    587                *,               LCGICAL NAME-L2740SM1  FEATGRP-STANDARD

                                    589                *,            TERMINAL-4 ACDR=E4 FEATGRP-STANDARD
                                    590                *,               LOGICAL NAME-L2740SM2  FEATGRP-STANDARD

                                    592                *,  LINEGRP-2  DCNAME-DD2740A  FEAT-TRANSCTL SWITCHED
                                    593                *,               UNITYPE-2740

                                    595                *,       LINE-5   FEAT-AUTOANS    ADDR-026
                                    596                *,                  ZCNE CODE IS 0

                                    598                *,            TERMINAL-5 ADDR=E2 FEATGRP-STANDARD
                                    599                *,               LOGICAL NAME-INQUIRY1  FEATGRP-STANDARD

                                    601                *,  LINEGRP-3  DCNAME-DD1050A  FEAT-STACTL SWITCHED
                                    602                *,               UNITYPF-1050

                                    604                *,       LINE-6   FEAT-AUTOANS    ACDR-027
                                    605                *,                  ZCNE CCDE IS 0

                                    607                *,            TERMINAL-6 ADDR=E2 FEATGRP-STANDARD
                                    608                *,               LCGICAL NAME-INQUIRY2  FEATGRP-STANDARD
                                    609                *,                     COMPT-0
```

```
IMS/360 SYSTEM DEFINITION SPECIFICATIONS                                               PAGE   15

LOC  OBJECT CODE     ADDR1 ADDR2  STMT   SOURCE STATEMENT                              F30SEP69   2/12/70

                                  611               *,        LINE-7   FEAT-AUTOANS     ADDR-000
                                  612               *,                 ******************************
                                  613               *,                 * LINE.IS A DIAL ANS POOL
                                  614               *,                 *       ZONE CODE IS 0.
                                  615               *,                 ******************************

                                  617               *,             TERMINAL-7 ADDR=00 FEATGRP-STANDARD
                                  618               *,                LOGICAL NAME-CAROL  FEATGRP-STANDARD
                                  619               *,                       COMPT-0

                                  621               *,             TERMINAL-8 ADDR=00 FEATGRP-STANDARD
                                  622               *,                LOGICAL NAME-ELEANOR  FEATGRP-STANDARD
                                  623               *,                       COMPT-0
                                  624               *,                LOGICAL NAME-DAN  FEATGRP-STANDARD
                                  625               *,                       COMPT-0
                                  626               *,                LOGICAL NAME-HOWARD  FEATGRP-STANDARD
                                  627               *,                       COMPT-0

                                  629               *,             TERMINAL-9 ADDR=00 FEATGRP-STANDARD
                                  630               *,                LOGICAL NAME-SHARRON  FEATGRP-STANDARD
                                  631               *,                       COMPT-0
                                  632               *,                LOGICAL NAME-RICHARD  FEATGRP-STANDARD
                                  633               *,                       COMPT-0
                                  634               *,                LOGICAL NAME-JOE  FEATGRP-STANDARD
                                  635               *,                       COMPT-0

                                  637               *,   LINEGRP-4  DDNAME-DD1050  FEAT-STACTL NONSWITCH
                                  638               *,                UNITYPE-1050

                                  640               *,        LINE-8   FEAT-POLL     ADDR-02A

                                  642               *,             TERMINAL-10 ADDR=E2 FEATGRP-STANDARD
                                  643               *,                LOGICAL NAME-PRINTER. FEATGRP-STANDARD
                                  644               *,                       COMPT-PTR1
                                  645               *,                LOGICAL NAME-T2780  FEATGRP-STANDARD
                                  646               *,                       COMPT-PTR1
                                  647               *,                LOGICAL NAME-TAPEPNCH  FEATGRP-STANDARD
                                  648               *,                       COMPT-PTPCH
                                  649               *,                LOGICAL NAME-MODEL2  FEATGRP-STANDARD
                                  650               *,                       COMPT-PTPCH
                                  651               *,                LOGICAL NAME-CARDPNCH  FEATGRP-STANDARD
                                  652               *,                       COMPT-3
                                  653               *,                LOGICAL NAME-MODEL2M  FEATGRP-STANDARD
                                  654               *,                       COMPT-3

                                  656               *,   LINEGRP-5  DDNAME-DD2260T  FEAT-STACTL NONSWITCH
                                  657               *,                UNITYPE-2260

                                  659               *,        LINE-9   FEAT-POLL     ADDR-0A2
```

```
IMS/360 SYSTEM DEFINITION SPECIFICATIONS                                               PAGE   16

LOC  OBJECT CODE     ADDR1 ADDR2  STMT   SOURCE STATEMENT                              F30SEP69   2/12/70

                                  661               *,             TERMINAL-11 ADDR=A0 FEATGRP-STANDARD
                                  662               *,                   CONTROL UNIT-40
                                  663               *,                LOGICAL NAME-BILL  FEATGRP-STANDARD

                                  665               *,             TERMINAL-12 ADDR=A1 FEATGRP-STANDARD
                                  666               *,                   CONTROL UNIT-40
                                  667               *,                LOGICAL NAME-LEONARD  FEATGRP-STANDARD
                                  668               *,                LOGICAL NAME-ERNE  FEATGRP-STANDARD

                                  670               *,             TERMINAL-13 ADDR=A2 FEATGRP-STANDARD
                                  671               *,                   CONTROL UNIT-40
                                  672               *,                LOGICAL NAME-CARL  FEATGRP-STANDARD

                                  674               *,             TERMINAL-14 ADDR=A3 FEATGRP-STANDARD
                                  675               *,                   CONTROL UNIT-40
                                  676               *,                LOGICAL NAME-BUD  FEATGRP-STANDARD
```

```
     IMS/360 SYSTEM DEFINITION SPECIFICATIONS                                                              PAGE   17

LOC  OBJECT CODE    ADDR1 ADDR2  STMT   SOURCE STATEMENT                                                F30SEP69   2/12/70
                                  679               *,IMS/360 DATA SET SPECIFICATIONS

                                  681               *,   QCR DATA SETS:
                                  682               *,        QCRIN DDNAME-INQCR
                                  683               *,              DSNAME-ICS.IQCRDSET
                                  684               *,                UNIT-2314
                                  685               *,              SERIAL-IMSDBS
                                  686               *,       QCROUT DDNAME-OUTQCR
                                  687               *,              DSNAME-ICS.OQCRDSET
                                  688               *,                UNIT-2314
                                  689               *,              SERIAL-IMSDBS

                                  691               *,   MESSAGE QUEQUE DATA SETS:
                                  692               *,        MSGIN DDNAME-INMSG
                                  693               *,              DSNAME-ICS.IMSGDSET
                                  694               *,                UNIT-2314
                                  695               *,              SERIAL-IMSDBS
                                  696               *,       MSGOUT DDNAME-OUTMSG
                                  697               *,              DSNAME-ICS.OMSGDSET
                                  698               *,                UNIT-2314
                                  699               *,              SERIAL-IMSDBS


                                  701               *,   RESLIB SPECIFICATION:
                                  702               *,      DSNAME-ICS.CLOD  VOLUME-IMSLIB UNIT-2314

                                  704               *,   MACLIB SPECIFICATION:
                                  705               *,      DSNAME-ICS.BMAC  VOLUME-IMSLIB
                                  706               *,        UNIT-2314        COPY-ALL

                                  708               *,   PROCLIB SPECIFICATION:
                                  709               *,      DSNAME-ICS.PROCLIB VOLUME-STORGE UNIT-2314

                                  711               *,   PGMLIB SPECIFICATION:
                                  712               *,      DSNAME-ICS.CLOD  VOLUME-IMSLIB UNIT-2314

                                  714               *,   PSBLIB SPECIFICATION:
                                  715               *,      DSNAME-ICS.PSBLIB VOLUME-IMSLIB UNIT-2314

                                  717               *,   DBDLIB SPECIFICATION:
                                  718               *,      DSNAME-ICS.DBDLIB VOLUME-IMSLIB UNIT-2314
```

```
     IMS/360 SYSTEM DEFINITION SPECIFICATIONS                                                              PAGE   18

LOC  OBJECT CODE    ADDR1 ADDR2  STMT   SOURCE STATEMENT                                                F30SEP69   2/12/70
                                 721+         PUNCH '//IMSGEN JOB 1,''IMSGEN STAGE II'',MSGCLASS=A,MSGLEVEL=X
                                    +               1'
                                 722+         PUNCH '//STEP1 EXEC PGM=IEHMOVE,REGION=100K'
                                 723+         PUNCH '//SYSPRINT DD SYSOUT=A'
                                 724+         PUNCH '//SYSUT1 DD DSNAME=TEMPSET,DISP=(OLD,PASS)'
                                 725+         PUNCH '//DD2 DD DSNAME=IMS.LOAD,DISP=(OLD,PASS)'
                                 726+         PUNCH '//DD3    DD     VOLUME=SER=IMSLIB,DISP=(OLD,PASS),      X
                                    +                            CONTINUE'
                                 727+         PUNCH '//             DSNAME=ICS.CLOD,UNIT=2314'
                                 728+         PUNCH '//SYSIN DD *'
                                 729+         PUNCH '  COPY PDS=IMS.LOAD,TO=2314=IMSLIB,RENAME=ICS.CLOD'
                                 730+         PUNCH '  SELECT MEMBER=DFSIRA00   REGION ANALYZER MODULE'
                                 731+         PUNCH '  SELECT MEMBER=DFSIRCC0   REGION CONTROLLER MODULE'
                                 732+         PUNCH '  SELECT MEMBER=DFSIPCC0   PROG. CONTROLLER MODULE'
                                 733+         PUNCH '  SELECT MEMBER=DFSIPR00   PROG. REQUEST HANDLER'
                                 734+         PUNCH '  SELECT MEMBER=DFSILNK0   IMS/360 LINKAGE EDITOR'
                                 735+         PUNCH '  SELECT MEMBER=DFSILI00   DL/I LANGUAGE INTERFACE'
                                 736+         PUNCH '  SELECT MEMBER=DFSIDLR0   DL/I RETRIEVE MODULE'
                                 737+         PUNCH '  SELECT MEMBER=DFSIDLI0   DL/I INSERT MODULE'
                                 738+         PUNCH '  SELECT MEMBER=DFSIDLD0   DL/I DELETE/REPLACE MODULE'
                                 739+         PUNCH '  SELECT MEMBER=DFSIDLE0   DL/I DATA BASE LOAD MODULE'
                                 740+         PUNCH '  SELECT MEMBER=DFSIDLN0   DL/I BATCH INITIALIZATION'
                                 741+         PUNCH '  SELECT MEMBER=DFSIDLH0   DL/I HSAM MODULE'
                                 742+         PUNCH '  SELECT MEMBER=DFSIDLT0   DL/I PROGRAM TEST MODULE'
                                 743+         PUNCH '  SELECT MEMBER=DFSISNAP   DL/I BLOCK SNAP ROUTINE'
                                 744+         PUNCH '  SELECT MEMBER=DFSIISM0   DL/I ISAM SIMULATOR'
                                 745+         PUNCH '  SELECT MEMBER=DFSIWKN0   DL/I WRITE KEY NEW MODULE'
                                 746+         PUNCH '  SELECT MEMBER=DFSIDLK0   DL/I BLOCK LOADER MODULE'
                                 747+         PUNCH '  SELECT MEMBER=DFSIOS20   OSAM READ/WRITE MODULE'
                                 748+         PUNCH '  SELECT MEMBER=DFSIOS30   OSAM CHECK ROUTINE'
                                 749+         PUNCH '  SELECT MEMBER=DFSIOS60   OSAM OPEN/CLOSE(OVFW)'
                                 750+         PUNCH '  SELECT MEMBER=DFSIOS10   OSAM OPEN ROUTINE'
                                 751+         PUNCH '  SELECT MEMBER=DFSISMM0   STORAGE MANAGEMENT MODULE'
                                 752+         PUNCH '  SELECT MEMBER=DFSIDLO0   DL/I OPEN MODULE'
                                 753+         PUNCH '  SELECT MEMBER=DFSIDLC0   DL/I CLOSE MODULE'
                                 754+         PUNCH '  SELECT MEMBER=DFSIDBA0   DL/I BATCH ANALYZER'
                                 755+         PUNCH '  SELECT MEMBER=DFSIBKB0   DL/I BATCH BLOCK MODULE'
                                 756+         PUNCH '  SELECT MEMBER=DFSIINL0   INIT - MODULE LOADER'
                                 757+         PUNCH '  SELECT MEMBER=DFSIIN10   INIT - JOBLIB MODULE LOADER'
                                 758+         PUNCH '  SELECT MEMBER=DFSIIN20   INIT - SVCLIB MODULE LOADER'
                                 759+         PUNCH '  SELECT MEMBER=((DFSIOCE0,IGG019Z8))   OSAM CH. END APX
                                    +               PENDAGE'
                                 760+         PUNCH '  SELECT MEMBER=DFSIXXX0     RESIDENT MODULE MAP'
                                 761+         PUNCH '  SELECT MEMBER=DFSISVV0     INTER-REGION SVC RTNES'
                                 762+         PUNCH '  SELECT MEMBER=DFSIDSP0     IMS SUBTASK DISPATCHER'
                                 763+         PUNCH '  SELECT MEMBER=DFSIRST0     IMS RESTART'
                                 764+         PUNCH '  SELECT MEMBER=DFSIRSI0     RESTART INITIALIZATION'
                                 765+         PUNCH '  SELECT MEMBER=DFSICP00     IMS CHECKPOINT'
                                 766+         PUNCH '  SELECT MEMBER=DFSIINT0     INIT - CONTROL && MISC'
                                 767+         PUNCH '  SELECT MEMBER=DFSIIND0     INIT - DMB DIRECTORY'
```

```
                                  768+          PUNCH '  SELECT MEMBER=DFSIINSO      INIT - STORAGE POOL MGMT'
                                  769+          PUNCH '  SELECT MEMBER=DFSIINQC      INIT - QUEUE MANAGEMENT'
                                  770+          PUNCH '  SELECT MEMBER=DFSIINPO      INIT - COMMUNICATIONS'
                                  771+          PUNCH '  SELECT MEMBER=DFSIINXO      INIT - RESIDENT XFR CTL'
                                  772+          PUNCH '  SELECT MEMBER=DFSISMNC       STORAGE POOL MGMT(O/L)'
                                  773+          PUNCH '  SELECT MEMBER=DFSICBLO    DATABASE LOGGER MODULE'
                                  774+          PUNCH '  SELECT MEMBER=DFSIBDRO    DATABASE RECOVERY MODULE'
                                  775+          PUNCH '  SELECT MEMBER=DFSIBDPO    DB RECOVERY PSB MODULE'
                                  776+          PUNCH '  SELECT MEMBER=DFSIDBHO      DBDUMP - DBD ANALYZER'
                                  777+          PUNCH '  SELECT MEMBER=DFSIASIO      SCHEDULER - INITIATION'
                                  778+          PUNCH '  SELECT MEMBER=DFSIASTO      SCHEDULER - TERMINATION'
                                  779+          PUNCH '  SELECT MEMBER=DFSICLIO      COMM INPUT PROCESSOR'
                                  780+          PUNCH '  SELECT MEMBER=DFSICLOC      COMM OUTPUT PROCESSOR'
                                  781+          PUNCH '  SELECT MEMBER=DFSICLPO      COMMAND MSGE PROCESSOR'
                                  782+          PUNCH '  SELECT MEMBER=DFSICLRO      MESSAGE ROUTER'
                                  783+          PUNCH '  SELECT MEMBER=DFSICLMO      MESSAGE GENERATOR'
                                  784+          PUNCH '  SELECT MEMBER=DFSICM10      COMM MESSAGE TABLE'
                                  785+          PUNCH '  SELECT MEMBER=DFSICLTO      COMM TRANSLATION MODULE'
                                  786+          PUNCH '  SELECT MEMBER=DFSICLBC      COMM BACKSPACE EDIT'
                                  787+          PUNCH '  SELECT MEMBER=DFSICLFO      SYMBOLIC DEST FINDER'
                                  788+          PUNCH '  SELECT MEMBER=DFSICLSC      SECURITY PROCESSOR'
                                  789+          PUNCH '  SELECT MEMBER=DFSICLXO      RESET POLL'
                                  790+          PUNCH '  SELECT MEMBER=DFSICLIO      /BROADCAST COMMAND'
                                  791+          PUNCH '  SELECT MEMBER=DFSICL20      /CHE /RES COMMAND'
                                  792+          PUNCH '  SELECT MEMBER=DFSICLAO      /IAM COMMAND'
                                  793+          PUNCH '  SELECT MEMBER=DFSICL30      EDIT COMMAND MSGE'
                                  794+          PUNCH '  SELECT MEMBER=DFSICL40      /STA /STO /PST COMMAND'
                                  795+          PUNCH '  SELECT MEMBER=DFSICL50      /TES /END /EXC COMMAND'
                                  796+          PUNCH '  SELECT MEMBER=DFSICL60      /CHANGE COMMAND'
                                  797+          PUNCH '  SELECT MEMBER=DFSICL7C      /ASSIGN COMMAND'
                                  798+          PUNCH '  SELECT MEMBER=DFSICL80      /DELETE COMMAND'
                                  799+          PUNCH '  SELECT MEMBER=DFSICL9C      /LOCK /UNLOCK COMMAND'
                                  800+          PUNCH '  SELECT MEMBER=DFSICLEO      /SET /RESET COMMANDS'
                                  801+          PUNCH '  SELECT MEMBER=DFSICLDO    /DISPLAY CONTROL MODULE'
                                  802+          PUNCH '  SELECT MEMBER=DFSIDP10        "     STATUS'
                                  803+          PUNCH '  SELECT MEMBER=DFSIDP20        "     ACTIVE'
                                  804+          PUNCH '  SELECT MEMBER=DFSIDP3C        "     QUEUES'
                                  805+          PUNCH '  SELECT MEMBER=DFSIDP40        "     TRAN && LTERM'
                                  806+          PUNCH '  SELECT MEMBER=DFSIDP50        "     PGM && DATABASE'
                                  807+          PUNCH '  SELECT MEMBER=DFSIDP60        "     LINE && PTERM'
                                  808+          PUNCH '  SELECT MEMBER=DFSIDP70        "     ASSIGNMENT'
                                  809+          PUNCH '  SELECT MEMBER=DFSIRD10        "     MASTER'
                                  810+          PUNCH '  SELECT MEMBER=DFSIDLAC      DL/I CALL ANALYZER'
                                  811+          PUNCH '  SELECT MEMBER=DFSIDLMO      DL/I BLOCK MOVER'
                                  812+          PUNCH '  SELECT MEMBER=DFSIIDEO      BLOCK DEQUEUE'
                                  813+          PUNCH '  SELECT MEMBER=DFSIIENO      BLOCK ENQUEUE'
                                  814+          PUNCH '  SELECT MEMBER=DFSIMBEO      SCHEDULER - SMB ENQUEUE'
                                  815+          PUNCH '  SELECT MEMBER=DFSIMBDO      SCHEDULER - SMB DEQUEUE'
                                  816+          PUNCH '  SELECT MEMBER=DFSIPREO      MSGE AND LOG PREFIX BLDR'
                                  817+          PUNCH '  SELECT MEMBER=DFSILOOO      WRITE LOG ROUTINE'
```

```
                                  818+          PUNCH '  SELECT MEMBER=DFSISTPO      START REGION'
                                  819+          PUNCH '  SELECT MEMBER=DFSIPTPO      STOP REGION'
                                  820+          PUNCH '  SELECT MEMBER=DFSIASEO      SIM REGION TERMINATION'
                                  821+          PUNCH '  SELECT MEMBER=DFSIRWOO      READ/WRITE MSGE QUEUE'
                                  822+          PUNCH '  SELECT MEMBER=DFSIQMSO      QUEUE REUSE MODULE'
                                  823+          PUNCH '  SELECT MEMBER=DFSISTO1      IMS STATISTICS MODULE'
                                  824+          PUNCH '  SELECT MEMBER=DFSISTO2      IMS STATISTICS MODULE'
                                  825+          PUNCH '  SELECT MEMBER=DFSISTO3      IMS STATISTICS MODULE'
                                  826+          PUNCH '  SELECT MEMBER=DFSISTO4      IMS STATISTICS MODULE'
                                  827+          PUNCH '  SELECT MEMBER=DFSISMIO      SECURITY MAINT INIT'
                                  828+          PUNCH '  SELECT MEMBER=DFSISMPO      SECURITY MAINT'
                                  829+          PUNCH '/*'
                                  830+          PUNCH '//STEP2 EXEC PGM=IEWL,PARM=''RENT,NCAL,XREF,LIST'',REGIX
                                     +                ON=110K'
                                  831+          PUNCH '//SYSPRINT DD SYSOUT=A'
                                  832+          PUNCH '//SYSLIN DD DDNAME=SYSIN'
                                  833+          PUNCH '//SYSLMOD DD    VOLUME=SER=IMSLIB,DISP=(OLD,PASS),       X
                                     +                               CONTINUE'
                                  834+          PUNCH '//              DSNAME=ICS.CLOD,UNIT=2314'
                                  835+          PUNCH '//SYSOBJ DD DSNAME=IMS.LOAD,DISP=(OLD,PASS)'
                                  836+          PUNCH '//SYSUT1 DD     UNIT=(SYSDA,SEP=(SYSLIN,SYSLMOD)),DISP=(X
                                     +                ,DELETE),        X'
                                  837+          PUNCH '//              SPACE=(1700,(100,50),RLSE)'
                                  838+          PUNCH '//SYSIN DD *'
                                  839+          PUNCH '  CHANGE IGC255(IGC243)'
                                  840+          PUNCH '  INCLUDE SYSOBJ(DFSIOSVO)     OSAM SVC ROUTINE'
                                  841+          PUNCH '  NAME IGC243(R)'
                                  842+          PUNCH '/*'
                                  843+          PUNCH '//STEP3 EXEC PGM=IEHMOVE,REGION=100K'
                                  844+          PUNCH '//SYSPRINT DD SYSOUT=A'
                                  845+          PUNCH '//SYSUT1 DD DSNAME=TEMPSET,DISP=(OLD,PASS)'
                                  846+          PUNCH '//DD2 DD DSNAME=IMS.GENLIB,DISP=(OLD,PASS)'
                                  847+          PUNCH '//DD3 DD    VOLUME=SER=IMSLIB,DISP=OLD,                    X
                                     +                               CONTINUE'
                                  848+          PUNCH '//              DSNAME=ICS.BMAC,UNIT=2314'
                                  849+          PUNCH '//SYSIN DD *'
                                  850+          PUNCH '  COPY PDS=IMS.GENLIB,TO=2314=IMSLIB,RENAME=ICS.BMAC'
                                  851+          PUNCH '/*'
                                  852+          PUNCH '//STEP4 EXEC PGM=IEBUPDTE,PARM=NEW,REGION=90K'
                                  853+          PUNCH '//SYSPRINT DD SYSOUT=A'
                                  854+          PUNCH '//SYSUT2 DD     VOLUME=SER=STORGE,DISP=(OLD,PASS),       X
                                     +                               CONTINUE'
                                  855+          PUNCH '//              DSNAME=ICS.PROCLIB,UNIT=2314'
                                  856+          PUNCH '//SYSIN DD DATA'
                                  857+          PUNCH './         ADD    NAME=DLITCBL'
                                  858+          PUNCH '  INCLUDE SYSOBJ(DFSILIOO)'
                                  859+          PUNCH '  ENTRY DLITCBL'
                                  860+          PUNCH './         ADD    NAME=DLITPLI'
                                  861+          PUNCH '  INCLUDE SYSOBJ(DFSILIOO)'
                                  862+          PUNCH '  ENTRY IHESAPD'
```

```
    IMS/360 SYSTEM DEFINITION SPECIFICATIONS                                                          PAGE   21

LOC  OBJECT CODE     ADDR1 ADDR2  STMT    SOURCE STATEMENT                                          F30SEP69   2/12/70

                                  863+         PUNCH './        ADD   NAME=DLIBATCH'
                                  864+         PUNCH './        NUMBER NEW1=00000010,INCR=00000010'
                                  865+         PUNCH '//        PROC  PSB=TEMPNAME'
                                  866+         PUNCH '//G     EXEC PGM=DFSIRC00,PARM=''3,&&PSB'',REGION=120K'
                                  867+         PUNCH '//IMS DD    VOLUME=SER=IMSLIB,DISP=SHR,                      X
                                     +                             CONTINUE'
                                  868+         PUNCH '//                DSNAME=ICS.PSBLIB,UNIT=2314'
                                  869+         PUNCH '//     DD    VOLUME=SER=IMSLIB,DISP=SHR,                      X
                                     +                             CONTINUE'
                                  870+         PUNCH '//                DSNAME=ICS.DBDLIB,UNIT=2314'
                                  871+         PUNCH '//SYSUDUMP DD SYSOUT=A,SPACE=(605,(500,500),RLSE,,ROUNDX
                                     +               ),                 X'
                                  872+         PUNCH '//                DCB=(RECFM=FBA,LRECL=121,BLKSIZE=605)'
                                  873+         PUNCH './        ADD   NAME=IMSCOBOL'
                                  874+         PUNCH './        NUMBER NEW1=00000010,INCR=00000010'
                                  875+         PUNCH '//        PROC MBR=,PAGES=60'
                                  876+         PUNCH '//C         EXEC PGM=IEQCBL00,PARM=''SIZE=110000,LINECNT=5X
                                     +               0'',REGION=126K'
                                  877+         PUNCH '//SYSLIN DD    DSNAME=&&&&LIN,DISP=(MOD,PASS),UNIT=SYSDAX
                                     +               ,DCB=(LRECL=80,    X'
                                  878+         PUNCH '//                RECFM=FB,BLKSIZE=400),SPACE=(CYL,(4,1),RX
                                     +               LSE)'
                                  879+         PUNCH '//SYSPRINT DD SYSOUT=A,DCB=(RECFM=FBA,LRECL=121,BLKSIZEX
                                     +               =605),             X'
                                  880+         PUNCH '//                SPACE=(605,(&&PAGES.0,&&PAGES),RLSE,,ROUX
                                     +               ND)'
                                  881+         PUNCH '//SYSUT1 DD    UNIT=SYSDA,DISP=(NEW,DELETE),SPACE=(CYL,(X
                                     +               10,1),RLSE)'
                                  882+         PUNCH '//SYSUT2 DD    UNIT=SYSDA,DISP=(NEW,DELETE),SPACE=(CYL,(X
                                     +               10,1),RLSE)'
                                  883+         PUNCH '//SYSUT3 DD    UNIT=SYSDA,DISP=(NEW,DELETE),SPACE=(CYL,(X
                                     +               10,1),RLSE)'
                                  884+         PUNCH '//SYSUT4 DD    UNIT=SYSDA,DISP=(NEW,DELETE),SPACE=(CYL,(X
                                     +               10,1),RLSE)'
                                  885+         PUNCH '//L         EXEC PGM=DFSILNK0,PARM=''XREF,LIST,LET'',REGIOX
                                     +               N=100K,            X'
                                  886+         PUNCH '//                COND=(4,LT,C)'
                                  887+         PUNCH '//SYSLIB DD    DSNAME=SYS1.COBLIB,DISP=SHR'
                                  888+         PUNCH '//        DD    DSNAME=SYS1.PL1LIB,DISP=SHR'
                                  889+         PUNCH '//SYSOBJ DD    DSNAME=ICS.CLOD,DISP=SHR'
                                  890+         PUNCH '//SYSLIN DD    DSNAME=&&&&LIN,DISP=(OLD,DELETE)'
                                  891+         PUNCH '//        DD    VOLUME=SER=STORGE,DISP=SHR,                   X
                                     +                             CONTINUE'
                                  892+         PUNCH '//                DSNAME=ICS.PROCLIB(DLITCBL),UNIT=2314'
                                  893+         PUNCH '//        DD    DDNAME=SYSIN'
                                  894+         PUNCH '//SYSLMOD DD   VOLUME=SER=IMSLIB,DISP=SHR,                   X
                                     +                             CONTINUE'
                                  895+         PUNCH '//                DSNAME=ICS.CLOD(&&MBR),UNIT=2314'
                                  896+         PUNCH '//SYSPRINT DD SYSOUT=A,DCB=(RECFM=FBA,LRECL=121,BLKSIZEX
                                     +               =605),             X'
```

```
    IMS/360 SYSTEM DEFINITION SPECIFICATIONS                                                          PAGE   22

LOC  OBJECT CODE     ADDR1 ADDR2  STMT    SOURCE STATEMENT                                          F30SEP69   2/12/70

                                  897+         PUNCH '//                SPACE=(605,&&PAGES.0,RLSE,,ROUND)'
                                  898+         PUNCH '//SYSUT1 DD    UNIT=SYSDA,DISP=(NEW,DELETE),SPACE=(CYL,(X
                                     +               10,1),RLSE)'
                                  899+         PUNCH './        ADD   NAME=IMSPLI'
                                  900+         PUNCH './        NUMBER NEW1=10,INCR=10'
                                  901+         PUNCH '//        PROC  MBR=,PAGES=50'
                                  902+         PUNCH '//C       EXEC    PGM=IEMAA,PARM=''XREF,ATR,LOAD,NODECK,NX
                                     +               OMACRO,OPT=1'',    X'
                                  903+         PUNCH '//                REGION=114K'
                                  904+         PUNCH '//SYSUT1 DD       UNIT=SYSDA,SPACE=(1024,(60,60),RLSE,,ROUX
                                     +               ND),               X'
                                  905+         PUNCH '//                DCB=(BLKSIZE=1024),DISP=(NEW,PASS)'
                                  906+         PUNCH '//SYSUT3 DD       UNIT=SYSDA,SPACE=(1024,(60,60),RLSE,,ROUX
                                     +               ND),               X'
                                  907+         PUNCH '//                DCB=(BLKSIZE=1024),DISP=(NEW,PASS)'
                                  908+         PUNCH '//SYSPRINT DD     SYSOUT=A,DCB=(LRECL=125,BLKSIZE=629,RECFX
                                     +               M=VBA),            X'
                                  909+         PUNCH '//                SPACE=(605,(&&PAGES.0,&&PAGES),RLSE)'
                                  910+         PUNCH '//SYSLIN DD       UNIT=SYSDA,SPACE=(80,(250,80),RLSE),DCB=X
                                     +               BLKSIZE=80,        X'
                                  911+         PUNCH '//                DISP=(NEW,PASS)'
                                  912+         PUNCH '//L       EXEC    PGM=DFSILNK0,PARM=''XREF,LIST,LET'',CONDX
                                     +               =(4,LT,C),          X'
                                  913+         PUNCH '//                REGION=100K'
                                  914+         PUNCH '//SYSLIB DD       DSNAME=SYS1.PL1LIB,DISP=SHR'
                                  915+         PUNCH '//        DD      DSNAME=SYS1.COBLIB,DISP=SHR'
                                  916+         PUNCH '//SYSLIN DD       DSNAME=*.C.SYSLIN,DISP=(OLD,DELETE)'
                                  917+         PUNCH '//        DD      VOLUME=SER=STORGE,DISP=SHR,                X
                                     +                             CONTINUE'
                                  918+         PUNCH '//                DSNAME=ICS.PROCLIB(DLITPLI),UNIT=2314'
                                  919+         PUNCH '//        DD      DDNAME=SYSIN'
                                  920+         PUNCH '//SYSLMOD DD      VOLUME=SER=IMSLIB,DISP=SHR,                X
                                     +                             CONTINUE'
                                  921+         PUNCH '//                DSNAME=ICS.CLOD(&&MBR),UNIT=2314'
                                  922+         PUNCH '//SYSPRINT DD     SYSOUT=A,DCB=(LRECL=121,BLKSIZE=605,RECFX
                                     +               M=FBA),            X'
                                  923+         PUNCH '//                SPACE=(605,(&&PAGES.0,&&PAGES),RLSE)'
                                  924+         PUNCH '//SYSOBJ DD       DSNAME=ICS.CLOD,DISP=SHR'
                                  925+         PUNCH '//SYSUT1 DD       UNIT=SYSDA,DISP=(NEW,DELETE),SPACE=(CYL,X
                                     +               (5,1),RLSE)'
                                  926+         PUNCH './        ADD     NAME=IMSCOBGO'
                                  927+         PUNCH './        NUMBER NEW1=00000010,INCR=00000010'
                                  928+         PUNCH '//        PROC MBR=,PAGES=60'
                                  929+         PUNCH '//C       EXEC PGM=IEQCBL00,PARM=''SIZE=110000,LINECNT=5X
                                     +               0'',REGION=126K'
                                  930+         PUNCH '//SYSLIN DD    DSNAME=&&&&LIN,DISP=(MOD,PASS),UNIT=SYSDAX
                                     +               ,DCB=(LRECL=80,    X'
                                  931+         PUNCH '//                RECFM=FB,BLKSIZE=400),SPACE=(CYL,(4,1),RX
                                     +               LSE)'
                                  932+         PUNCH '//SYSPRINT DD SYSOUT=A,DCB=(RECFM=FBA,LRECL=121,BLKSIZEX
    +                =605),          X'
 933+          PUNCH '//             SPACE=(6C5,(&&PAGES.O,&&PAGES),RLSE,,RCUX
    +                ND)'
 934+          PUNCH '//SYSUT1 DC   UNIT=SYSDA,DISP=(NEW,DELETE),SPACE=(CYL,(X
    +                10,1),RLSE)'
 935+          PUNCH '//SYSUT2 DC   UNIT=SYSDA,CISP=(NEW,CELETE),SPACE=(CYL,(X
    +                10,1),RLSE)'
 936+          PUNCH '//SYSUT3 DD   UNIT=SYSDA,DISP=(NEW,DELETE),SPACE=(CYL,(X
    +                10,1),RLSE)'
 937+          PUNCH '//SYSUT4 DD   UNIT=SYSDA,DISP=(NEW,DELETE),SPACE=(CYL,(X
    +                10,1),RLSE)'
 938+          PUNCH '//L      EXEC PGM=DFSILNKO,PARM=''XREF,LIST,LET'',REGIOX
    +                N=100K,           X'
 939+          PUNCH '//             COND=(4,LT,C)'
 940+          PUNCH '//SYSLIB DD   DSNAME=SYS1.COBLIB,DISP=SHR'
 941+          PUNCH '//       DC   DSNAME=SYS1.PL1LIB,DISP=SHR'
 942+          PUNCH '//SYSOBJ DC   DSNAME=ICS.CLCO,DISP=SHR'
 943+          PUNCH '//SYSLIN DD   DSNAME=&&&&LIN,DISP=(CLD,DELETE)'
 944+          PUNCH '//       DD   VOLUME=SER=STORGE,DISP=SHR,              X
    +                            CONTINUE'
 945+          PUNCH '//             DSNAME=ICS.PROCLIB(CLITCBL),UNIT=2314'
 946+          PUNCH '//       DC   DCNAME=SYSIN'
 947+          PUNCH '//SYSLMOD DD  VCLUME=SER=IMSLIB,CISP=SHR,              X
    +                            CONTINUE'
 948+          PUNCH '//             DSNAME=ICS.CLOD(&&MBR),UNIT=2314'
 949+          PUNCH '//SYSPRINT DD SYSOUT=A,DCB=(RECFM=FBA,LRECL=121,BLKSIZEX
    +                =605),          X'
 95C+          PUNCH '//             SPACE=(605,&&PAGES.O,RLSE,,ROUND)'
 951+          PUNCH '//SYSUT1 DD   UNIT=SYSDA,DISP=(NEW,DELETE),SPACE=(CYL,(X
    +                10,1),RLSE)'
 952+          PUNCH '//G      EXEC  PGM=DFSIRC00,PARM=''3,&&MBR'',REGION=150X
    +                K,COND=(0,LT),    X'
 953+          PUNCH '//             TIME=2'
 954+          PUNCH '//IMS    DD    VOLUME=SER=IMSLIB,CISP=SHR,             X
    +                            CONTINUE'
 955+          PUNCH '//             CSNAME=ICS.PSBLIB,UNIT=2314'
 956+          PUNCH '//       DD    VOLUME=SER=IMSLIB,DISP=SHR,             X
    +                            CONTINUE'
 957+          PUNCH '//             DSNAME=ICS.DBDLIB,UNIT=2314'
 958+          PUNCH '//SYSOUT DD    SYSOUT=A,SPACE=(CYL,(1,1)),DCB=(LRECL=13X
    +                3,RECFM=FA)'
 959+          PUNCH '//SYSUDUMP DD  SYSOUT=A,DCB=(LRECL=121,RECFM=FBA,BLKSIZX
    +                E=3025),        X'
 960+          PUNCH '//             SPACE=(3025,(2C0,100),RLSE,,ROUND)'
 961+          PUNCH './       ADD   NAME=IMSPLIGO'
 962+          PUNCH './       NUMBER NEW1=10,INCR=10'
 963+          PUNCH '//       PROC  MBR=,PAGES=50'
 964+          PUNCH '//C      EXEC   PGM=IEMAA,PARM=''XREF,ATR,LOAD,NODECK,NX
    +                OMACRO,OPT=1'',   X'
 965+          PUNCH '//             REGICN=114K'
```

```
 966+          PUNCH '//SYSUT1 DD    UNIT=SYSDA,SPACE=(1024,(60,60),RLSE,,ROUX
    +                ND),            X'
 967+          PUNCH '//             DCB=(BLKSIZE=1024),DISP=(NEW,PASS)'
 968+          PUNCH '//SYSUT3 DD    UNIT=SYSDA,SPACE=(1024,(60,60),RLSE,,RCUX
    +                ND),            X'
 969+          PUNCH '//             DCB=(BLKSIZE=1024),CISP=(NEW,PASS)'
 970+          PUNCH '//SYSPRINT DD  SYSOUT=A,DCB=(LRECL=125,BLKSIZE=629,RECFX
    +                M=VBA),         X'
 971+          PUNCH '//             SPACE=(605,(&&PAGES.O,&&PAGES),RLSE)'
 972+          PUNCH '//SYSLIN DD    UNIT=SYSDA,SPACE=(80,(250,80),RLSE),DCB=X
    +                BLKSIZE=80,     X'
 973+          PUNCH '//             DISP=(NEW,PASS)'
 974+          PUNCH '//L      EXEC  PGM=DFSILNKO,PARM=''XREF,LIST,LET'',CONDX
    +                =(4,LT,C),        X'
 975+          PUNCH '//             REGICN=1C0K'
 976+          PUNCH '//SYSLIB DD    DSNAME=SYS1.PL1LIB,DISP=SHR'
 977+          PUNCH '//       DD    DSNAME=SYS1.COBLIB,CISP=SHR'
 978+          PUNCH '//SYSLIN DD    DSNAME=*.C.SYSLIN,DISP=(OLD,DELETE)'
 979+          PUNCH '//       DD    VOLUME=SER=STORGE,DISP=SHR,             X
    +                            CONTINUE'
 980+          PUNCH '//             DSNAME=ICS.PROCLIB(DLITPLI),UNIT=2314'
 981+          PUNCH '//       DD    DDNAME=SYSIN'
 982+          PUNCH '//SYSLMOD DD   VOLUME=SER=IMSLIB,CISP=SHR,             X
    +                            CONTINUE'
 983+          PUNCH '//             DSNAME=ICS.CLOD(&&MBR),UNIT=2314'
 984+          PUNCH '//SYSPRINT DD  SYSOUT=A,DCB=(LRECL=121,BLKSIZE=605,RECFX
    +                M=FBA),         X'
 985+          PUNCH '//             SPACE=(605,(&&PAGES.O,&&PAGES),RLSE)'
 986+          PUNCH '//SYSOBJ DD    DSNAME=ICS.CLOD,DISP=SHR'
 987+          PUNCH '//SYSUT1 DD    UNIT=SYSDA,DISP=(NEW,DELETE),SPACE=(CYL,X
    +                (5,1),RLSE)'
 988+          PUNCH '//G      EXEC  PGM=DFSIRC00,PARM=''3,&&MBR'',COND=(4,LTX
    +                ),REGION=150K,    X'
 989+          PUNCH '//             TIME=5'
 990+          PUNCH '//IMS    DD    VOLUME=SER=IMSLIB,DISP=SHR,             X
    +                            CONTINUE'
 991+          PUNCH '//             DSNAME=ICS.PSBLIB,UNIT=2314'
 992+          PUNCH '//       DD    VOLUME=SER=IMSLIB,DISP=SHR,             X
    +                            CONTINUE'
 993+          PUNCH '//             DSNAME=ICS.DBDLIB,UNIT=2314'
 994+          PUNCH '//SYSPRINT DC  SYSOUT=A,DCB=(LRECL=121,BLKSIZE=605,RECFX
    +                M=FBA),         X'
 995+          PUNCH '//             SPACF=(605,(500,500),RLSE,,ROUND)'
 996+          PUNCH '//SYSUDUMP DC  SYSOUT=A,DCB=(LRECL=121,BLKSIZE=605,RECFX
    +                M=FBA),         X'
 997+          PUNCH '//             SPACE=(605,(500,500),RLSE,,ROUND)'
 998+          PUNCH './       ADD   NAME=MFDBDUMP'
 999+          PUNCH './       NUMBER NEW1=10,INCR=10 '
1000+          PUNCH '//       PROC SOUT=A'
1001+          PUNCH '//DUMP EXEC PGM=DFSIRC00,PARM=''3,CFSSAM08'',REGION=110X
```

```
     IMS/36C SYSTEM DEFINITICN SPECIFICATICNS                                                          PAGE   25

LCC  CBJECT CCDE    ADDR1 ADDR2  STMT   SOURCE STATEMENT                                             F30SEP69   2/12/70

                                    +          K'
                                 1002+        PUNCH '//STEPLIB DD CSN=ICS.CLCD,DISP=SHR'
                                 1003+        PUNCH '//          DD   DSNAME=ICS.CLCD,DISP=SHR'
                                 1004+        PUNCH '//IMS       DD   DSNAME=ICS.PSBLIB,DISP=SHR'
                                 1005+        PUNCH '//          DD   DSNAME=ICS.DBDLIB,DISP=SHR'
                                 1006+        PUNCH '//SYSUDUMP DD SYSOUT=&&SOUT'
                                 1007+        PUNCH '//DI21PART DD DSNAME=IMS.DI21PART,DISP=SHR'
                                 1008+        PUNCH '//DI21PARO DD DSNAME=IMS.DI21PARO,CISP=SHR'
                                 1009+        PUNCH '//OUTPUT DD    SYSOUT=&&SOUT'
                                 1010+        PUNCH './          ADD  NAME=MFCBLOAD'
                                 1011+        PUNCH './          NUMBER NEW1=10,INCR=10'
                                 1012+        PUNCH '//          PROC SOUT=A'
                                 1013+        PUNCH '//LCAD EXEC PGM=DFSIRCOO,PARM=''3,DFSSAM01'',REGION=110X
                                    +          K'
                                 1014+        PUNCH '//STEPLIB DD CSN=ICS.CLCD,DISP=SHR'
                                 1015+        PUNCH '//          DD   DSNAME=ICS.CLCD,DISP=SHR'
                                 1016+        PUNCH '//IMS       DD   DSNAME=ICS.PSBLIB,DISP=SHR'
                                 1017+        PUNCH '//          DD   DSNAME=ICS.DBDLIB,DISP=SHR'
                                 1018+        PUNCH '//SYSUDUMP DC SYSOUT=&&SOUT'
                                 1019+        PUNCH '//DI21PART DD DSNAME=IMS.DI21PART(PRIME),DISP=(,KEEP),DX
                                    +          CB=CSORG=IS,     X'
                                 1020+        PUNCH '//                 SPACE=(CYL,3,,CONTIG),VOL=SER=&&PSER,UNIX
                                    +          T=&&PUNIT'
                                 1021+        PUNCH '//DI21PARO DC CSNAME=IMS.DI21PARO,DISP=(,KEEP),SPACE=(CX
                                    +          YL,3,,CONTIG),   X'
                                 1022+        PUNCH '//                 VOL=SER=&&OSER,UNIT=&&CUNIT'
                                 1023+        PUNCH '//SYSOUT DD    SYSCUT=&&SOUT'
                                 1024+        PUNCH '//INPUT  DD    DSNAME=ICS.BMAC(MFDFSYSN),DISP=SHR'
                                 1025+        PUNCH './          ADD   NAME=PSBGEN'
                                 1026+        PUNCH './          NUMBER NEW1=10,INCR=10'
                                 1027+        PUNCH '//          PROC  MBR=TEMPNAME'
                                 1028+        PUNCH '//C         EXEC  PGM=IEUASM,PARM=''LOAD,NODECK'',REGION=1X
                                    +          OOK'
                                 1029+        PUNCH '//SYSLIB DD      VOLUME=SER=IMSLIB,DISP=SHR,              X
                                    +                      CONTINUE'
                                 1030+        PUNCH '//                 DSNAME=ICS.BMAC,UNIT=2314'
                                 1031+        PUNCH '//          DD    DSNAME=SYS1.MACLIB,DISP=SHR'
                                 1032+        PUNCH '//SYSGO    DC    UNIT=SYSDA,DISP=(,PASS),DCB=(BLKSIZE=400X
                                    +          ,                X'
                                 1033+        PUNCH '//                 RECFM=FB,LRECL=80),SPACE=(80,(100,10C),RX
                                    +          LSE)'
                                 1034+        PUNCH '//SYSPRINT DD    SYSOUT=A,DCB=(LRECL=121,RECFM=FBA,BLKSIZX
                                    +          E=605),          X'
                                 1035+        PUNCH '//                 SPACF=(121,(500,500),RLSE,,ROUND)'
                                 1036+        PUNCH '//SYSUT1 DD      UNIT=SYSDA,DISP=(,DELETE),SPACE=(17CO,(1X
                                    +          00,50))'
                                 1037+        PUNCH '//SYSUT2 DD      UNIT=SYSDA,DISP=(,DELETE),SPACE=(1700,(1X
                                    +          00,50))'
                                 1038+        PUNCH '//SYSUT3 DC      UNIT=(SYSDA,SEP=(SYSLIB,SYSUT1,SYSUT2)),X
                                    +                           X'
```

```
     IMS/360 SYSTEM DEFINITION SPECIFICATIONS                                                          PAGE   26

LCC  CBJECT CODE    ADDR1 ADDR2  STMT   SOURCE STATEMENT                                             F30SEP69   2/12/70

                                 1039+        PUNCH '//                 SPACE=(1700,(100,50))'
                                 1040+        PUNCH '//L        EXEC  PGM=DFSILNKO,PARM=''XREF,LIST'',COND=(0,X
                                    +          LT,C),             X'
                                 1041+        PUNCH '//                 REGION=100K'
                                 1042+        PUNCH '//SYSLIN DD      DSNAME=*.C.SYSGO,DISP=(OLD,DELETE)'
                                 1043+        PUNCH '//          DD    DDNAME=SYSIN'
                                 1044+        PUNCH '//SYSPRINT DC    SYSOUT=A,DCB=(LRECL=121,RECFM=FBA,BLKSIZX
                                    +          E=605),          X'
                                 1045+        PUNCH '//                 SPACE=(121,(100,100),RLSE)'
                                 1046+        PUNCH '//SYSLMOD DD     VOLUME=SER=IMSLIB,CISP=SHR,              X
                                    +                      CONTINUE'
                                 1047+        PUNCH '//                 CSNAME=ICS.PSBLIB(&&MBR),UNIT=2314'
                                 1048+        PUNCH '//SYSUT1 DD      UNIT=(SYSDA,SEP=(SYSLMOD,SYSLIN)),DISP=(X
                                    +          ,DELETE),        X'
                                 1049+        PUNCH '//                 SPACE=(1024,(100,10),RLSE)'
                                 1050+        PUNCH './          ADD   NAME=DBDGEN'
                                 1051+        PUNCH './          NUMBER NEW1=10,INCR=10'
                                 1052+        PUNCH '//          PROC  MBR=TEMPNAME'
                                 1053+        PUNCH '//C         EXEC  PGM=IEUASM,PARM=''LCAD,NODECK'',REGION=1X
                                    +          OOK'
                                 1054+        PUNCH '//SYSLIB DD      VOLUME=SER=IMSLIB,CISP=SHR,              X
                                    +                      CONTINUE'
                                 1055+        PUNCH '//                 DSNAME=ICS.BMAC,UNIT=2314'
                                 1056+        PUNCH '//          DD    DSNAME=SYS1.MACLIB,DISP=SHR'
                                 1057+        PUNCH '//SYSGO    DD    UNIT=SYSDA,DISP=(,PASS),DCB=(BLKSIZE=400X
                                    +          ,                X'
                                 1058+        PUNCH '//                 RECFM=FB,LRECL=80),SPACE=(80,(100,100),RX
                                    +          LSE)'
                                 1059+        PUNCH '//SYSPRINT CC    SYSOUT=A,DCB=(LRECL=121,RECFM=FBA,BLKSIZX
                                    +          E=605),          X'
                                 1060+        PUNCH '//                 SPACE=(121,(500,500),RLSE,,ROUND)'
                                 1061+        PUNCH '//SYSUT1 DD      UNIT=SYSDA,DISP=(,DELETE),SPACE=(1700,(1X
                                    +          00,50))'
                                 1062+        PUNCH '//SYSUT2 DD      UNIT=SYSDA,DISP=(,DELETE),SPACE=(17CO,(1X
                                    +          00,50))'
                                 1063+        PUNCH '//SYSUT3 DD      UNIT=(SYSDA,SEP=(SYSLIB,SYSUT1,SYSUT2)),X
                                    +                           X'
                                 1064+        PUNCH '//                 SPACE=(1700,(100,50))'
                                 1065+        PUNCH '//L        EXEC  PGM=DFSILNKO,PARM=''XREF,LIST'',COND=(0,X
                                    +          LT,C),             X'
                                 1066+        PUNCH '//                 REGICN=100K'
                                 1067+        PUNCH '//SYSLIN DD      DSNAME=*.C.SYSGO,DISP=(OLD,DELETE)'
                                 1068+        PUNCH '//          DD    DDNAME=SYSIN'
                                 1069+        PUNCH '//SYSPRINT DD    SYSOUT=A,DCB=(LRECL=121,RECFM=FBA,BLKSIZX
                                    +          E=605),          X'
                                 107C+        PUNCH '//                 SPACE=(121,(100,100),RLSE)'
                                 1071+        PUNCH '//SYSLMOD DD     VOLUME=SER=IMSLIB,DISP=SHR,              X
                                    +                      CONTINUE'
                                 1072+        PUNCH '//                 CSNAME=ICS.DBDLIB(&&MBR),UNIT=2314'
                                 1073+        PUNCH '//SYSUT1 DC      UNIT=(SYSDA,SEP=(SYSLMOD,SYSLIN)),DISP=(X
     IMS/360 SYSTEM DEFINITION SPECIFICATIONS                                                                   PAGE   27

LOC  OBJECT CODE    ADDR1 ADDR2  STMT   SOURCE STATEMENT                                            F30SEP69   2/12/70

                                     +               ,DELETE),       X'
                                 1074+         PUNCH '//             SPACE=(1024,(100,10),RLSE)'
                                 1075+         PUNCH './       ADD   NAME=IMS'
                                 1076+         PUNCH './      NUMBER NEW1=10,INCR=10'
                                 1077+         PUNCH '//IEFPROC EXEC PGM=IEFIRC,       READER FIRST LOAD     X
                                     +                               X'
                                 1078+         PUNCH '//             REGION=48K,       READER BASIC REGION   X
                                     +                               X'
                                 1079+         PUNCH '//             PARM=''00103005001024905010SYSDA   ''    X
                                     +                                 X'
                                 1080+         PUNCH '//             .     BPPTTTCCCMMMIIICCCRLSSSSSSSS  DEFAX
                                     +               ULT PARM FIELDS X'
                                 1081+         PUNCH '//             PROGRAMMER NAME AND ACCT NBR NOT NEEDED X
                                     +                      B        X'
                                 1082+         PUNCH '//             PRIORITY=01                             X
                                     +                      PP       X'
                                 1083+         PUNCH '//             JOB STEP INTERVAL=30 MINUTES            X
                                     +                      TTT      X'
                                 1084+         PUNCH '//             PRIMARY SYSOUT SPACE=50 TRACKS          X
                                     +                      OOO      X'
                                 1085+         PUNCH '//             SECONDARY SYSOUT SPACE=10 TRACKS        X
                                     +                      MMM      X'
                                 1086+         PUNCH '//             READER/INTERPRETER DISPATCHING PRIORITY=X
                                     +               249    III      X'
                                 1087+         PUNCH '//             JOB STEP DEFAULT REGION=50K             X
                                     +                      CCC      X'
                                 1088+         PUNCH '//             DISPLAY AND EXECUTE COMMANDS=1          X
                                     +                      R        X'
                                 1089+         PUNCH '//             BASIC LABEL=0                           X
                                     +                      L        X'
                                 1090+         PUNCH '//             SYSPUT UNIT NAME=SYSDA                  X
                                     +                      SSSSSSSS'
                                 1091+         PUNCH '//IEFRDER DD   UNIT=2314,                              X
                                     +                         CONTINUE'
                                 1092+         PUNCH '//             VOLUME=SER=STORGE,                      X
                                     +                            CONTINUE'
                                 1093+         PUNCH '//             DISP=SHR,                               X
                                     +                               X'
                                 1094+         PUNCH '//            DSNAME=ICS.PROCLIB(IMSO),DCB=BUFNO=1'
                                 1095+         PUNCH '//IEFPDSI DD   DSNAME=SYS1.PROCLIB,DISP=OLD   PROCEDUREX
                                     +                LIBRARY'
                                 1096+         PUNCH '//       DD    VOLUME=SER=STORGE,DISP=SHR,             X
                                     +                            CONTINUE'
                                 1097+         PUNCH '//             DSNAME=ICS.PROCLIB,UNIT=2314'
                                 1098+         PUNCH '//IEFDATA DD   UNIT=SYSDA,       SPOOL DEVICE          X
                                     +                               X'
                                 1099+         PUNCH '//             SPACE=(80,(500,500),RLSE,CONTIG),   AMOUX
                                     +               NT              X'
                                 1100+         PUNCH '//             DCB=(BUFNO=2,LRECL=80,BLKSIZE=80,RECFM=FX
                                     +               B,BUFL=80)'
```

```
     IMS/360 SYSTEM DEFINITION SPECIFICATIONS                                                                   PAGE   28

LOC  OBJECT CODE    ADDR1 ADDR2  STMT   SOURCE STATEMENT                                            F30SEP69   2/12/70

                                 1101+         PUNCH './       ADD   NAME=IMSO'
                                 1102+         PUNCH './      NUMBER NEW1=10,INCR=10'
                                 1103+         PUNCH '//IMSO   JOB   1,IMS,PRTY=12,MSGLEVEL=1'
                                 1104+         PUNCH '//NUCLEUS EXEC PGM=DFSIRC00,REGION=178K,TIME=1440,    X
                                     +                               X'
                                 1105+         PUNCH '//             PARM=''00DFSINUC0014010010010020''      X
                                     +                                 X'
                                 1106+         PUNCH '//             .     ABCCCCCCCCDDDEEEFFFGGGHHH   DEFAULX
                                     +               T PARM FIELD    X'
                                 1107+         PUNCH '//             REGION TYPE=0                      A    X
                                     +                               X'
                                 1108+         PUNCH '//             BTAM=0                             B    X
                                     +                               X'
                                 1109+         PUNCH '//             NUCLEUS MEMBER NAME                CCCCX
                                     +               CCCC            X'
                                 1110+         PUNCH '//             NUMBER OF QCR BUFFERS(CALCULATED)   DDD X
                                     +                               X'
                                 1111+         PUNCH '//             NUMBER OF MSG BUFFERS(CALCULATED)   EEE X
                                     +                               X'
                                 1112+         PUNCH '//             PSB POOL IN 1K BLOCKS(DEFAULT)      FFF X
                                     +                               X'
                                 1113+         PUNCH '//             DMB POOL IN 1K BLOCKS(DEFAULT)      GGG X
                                     +                               X'
                                 1114+         PUNCH '//             OSAM && TP POOL SIZE(DEFAULT)       HHH'
                                 1115+         PUNCH '//IMS    DD    VOLUME=SER=IMSLIB,DISP=SHR,             X
                                     +                             CONTINUE'
                                 1116+         PUNCH '//             DSNAME=ICS.PSBLIB,UNIT=2314'
                                 1117+         PUNCH '//       DD    VOLUME=SER=IMSLIB,DISP=SHR,             X
                                     +                             CONTINUE'
                                 1118+         PUNCH '//             DSNAME=ICS.DBDLIB,UNIT=2314'
                                 1119+         PUNCH '//STEPLIB DD DSN=ICS.CLCD,DISP=SHR'
                                 1120+         PUNCH '//SYSUDUMP DD  SYSOUT=A,DCB=(LRECL=125,RECFM=VBA,BLKSIZX
                                     +               E=3129),        X'
                                 1121+         PUNCH '//             SPACE=(125,(3000,3000),RLSE,,ROUND)'
                                 1122+         PUNCH '//INQCR DD DSNAME=ICS.IQCRDSET,DISP=OLD'
                                 1123+         PUNCH '//INMSG  DD  DSNAME=ICS.IMSGDSET,DISP=OLD'
                                 1124+         PUNCH '//OUTQCR DD DSNAME=ICS.OQCRDSET,DISP=OLD'
                                 1125+         PUNCH '//OUTMSG DD DSNAME=ICS.OMSGDSET,DISP=OLD'
                                 1126+         PUNCH '//IMSLOG DD    DSNAME=IMSLOG,DISP=(,KEEP),UNIT=(2400,,DX
                                     +               EFER),          X'
                                 1127+         PUNCH '//             DCB=(RECFM=VB,BLKSIZE=1408,LRECL=1400,BUX
                                     +               FNO=1),         X'
                                 1128+         PUNCH '//             VOL=(,,,10)'
                                 1129+         PUNCH '//IMSLOGR DD   DSNAME=IMSLOG,DISP=OLD,VOLUME=SER=000000X
                                     +               ,               X'
                                 1130+         PUNCH '//             UNIT=(2400,,DEFER)'
                                 1131+         PUNCH '//DBDUMP DD    DSNAME=DFSIDUMP,DISP=(NEW,KEEP),UNIT=AFFX
                                     +               =IMSLOGR'
                                 1132+         PUNCH '//DD2740S  DD    UNIT=022      ** IMS LINE 2'
                                 1133+         PUNCH '//         DD    UNIT=023      ** IMS LINE 3'
```

```
LOC  OBJECT CODE    ADDR1 ADDR2  STMT   SOURCE STATEMENT                                     F30SEP69   2/12/70

                                1134+        PUNCH '//            DD     UNIT=024      ** IMS LINE 4'
                                1135+        PUNCH '//DD2740A  DD     UNIT=026      ** IMS LINE 5'
                                1136+        PUNCH '//DD1050A  DD     UNIT=027      ** IMS LINE 6'
                                1137+        PUNCH '//DD1050   DD     UNIT=024      ** IMS LINE 8'
                                1138+        PUNCH '//DD2260T  DD     UNIT=0A2      ** IMS LINE 9'
                                1139+        PUNCH './         ADD    NAME=IMS1'
                                1140+        PUNCH './         NUMBER NEW1=10,INCR=10'
                                1141+        PUNCH '//NUCLEUS EXEC PGM=DFSIRC00,REGION=178K,TIME=1440,       X
                                   +                           X'
                                1142+        PUNCH '//                 PARM=''00DFSINUCC014C10010010020''    X
                                   +                             X'
                                1143+        PUNCH '//                 .     ABCCCCCCCCDDDEEEFFFGGGHHH    DEFAULX
                                   +               T PARM FIELD     X'
                                1144+        PUNCH '//                 REGION TYPE=0                        A    X
                                   +                                X'
                                1145+        PUNCH '//                 BTAM=0                               B    X
                                   +                                X'
                                1146+        PUNCH '//                 NUCLEUS MEMBER NAME                  CCCCX
                                   +               CCCC             X'
                                1147+        PUNCH '//                 NUMBER OF QCR BUFFERS(CALCULATED)    DDD X
                                   +                                X'
                                1148+        PUNCH '//                 NUMBER OF MSG BUFFERS(CALCULATED)    EEE X
                                   +                                X'
                                1149+        PUNCH '//                 PSB POOL IN 1K BLOCKS(DEFAULT)       FFF X
                                   +                                X'
                                1150+        PUNCH '//                 DMB POOL IN 1K BLOCKS(DEFAULT)       GGG X
                                   +                                X'
                                1151+        PUNCH '//                 CSAM && TP POOL SIZE(DEFAULT)        HHH'
                                1152+        PUNCH '//IMS     DD       VOLUME=SER=IMSLIB,DISP=SHR,                X
                                   +                             CONTINUE'
                                1153+        PUNCH '//                 DSNAME=ICS.PSBLIB,UNIT=2314'
                                1154+        PUNCH '//        DD       VOLUME=SER=IMSLIB,DISP=SHR,                X
                                   +                             CONTINUE'
                                1155+        PUNCH '//                 DSNAME=ICS.DBDLIB,UNIT=2314'
                                1156+        PUNCH '//STEPLIB DD DSN=ICS.CLOD,DISP=SHR'
                                1157+        PUNCH '//SYSUDUMP DD  SYSOUT=A,DCB=(LRECL=125,RECFM=VBA,BLKSIZX
                                   +               E=3129),         X'
                                1158+        PUNCH '//                 SPACE=(125,(3000,3000),RLSE,,ROUND)'
                                1159+        PUNCH '//INQCR DD DSNAME=ICS.IQCRDSET,DISP=OLD'
                                1160+        PUNCH '//INMSG  DD DSNAME=ICS.IMSGDSET,DISP=OLD'
                                1161+        PUNCH '//OUTQCR DD DSNAME=ICS.OQCRDSET,DISP=OLD'
                                1162+        PUNCH '//OUTMSG DD DSNAME=ICS.OMSGDSET,DISP=OLD'
                                1163+        PUNCH '//IMSLOG DD      DSNAME=IMSLOG,DISP=(,KEEP),UNIT=(2400,,DX
                                   +               EFER),           X'
                                1164+        PUNCH '//                 DCB=(RECFM=VB,BLKSIZE=1408,LRECL=1400,BUX
                                   +               FNO=1),          X'
                                1165+        PUNCH '//                 VOL=(,,,10)'
                                1166+        PUNCH '//IMSLOGR DD     DSNAME=IMSLOG,DISP=OLD,VOLUME=SER=000000X
                                   +               ,                X'
                                1167+        PUNCH '//                 UNIT=(2400,,DEFER)'
```

```
LOC  OBJECT CODE    ADDR1 ADDR2  STMT   SOURCE STATEMENT                                     F30SEP69   2/12/70

                                1168+        PUNCH '//DBDUMP DD      DSNAME=DFSIDUMP,DISP=(NEW,KEEP),UNIT=AFFX
                                   +               =IMSLOGR'
                                1169+        PUNCH '//DD2740S  DD     UNIT=022      ** IMS LINE 2'
                                1170+        PUNCH '//            DD     UNIT=023      ** IMS LINE 3'
                                1171+        PUNCH '//            DD     UNIT=024      ** IMS LINE 4'
                                1172+        PUNCH '//DD2740A  DD     UNIT=026      ** IMS LINE 5'
                                1173+        PUNCH '//DD1050A  DD     UNIT=027      ** IMS LINE 6'
                                1174+        PUNCH '//DD1050   DD     UNIT=024      ** IMS LINE 8'
                                1175+        PUNCH '//DD2260T  DD     UNIT=0A2      ** IMS LINE 9'
                                1176+        PUNCH './         ADD    NAME=IMSBATCH'
                                1177+        PUNCH './         NUMBER NEW1=10,INCR=10'
                                1178+        PUNCH '//         PROC   PSB=TEMPNAME'
                                1179+        PUNCH '//G        EXEC   PGM=DFSIRC00,PARM=''2,&&PSB'',REGION=26KX
                                   +               '
                                1180+        PUNCH '//IMS     DD       VOLUME=SER=IMSLIB,DISP=SHR,                X
                                   +                             CONTINUE'
                                1181+        PUNCH '//                 DSNAME=ICS.PSBLIB,UNIT=2314'
                                1182+        PUNCH '//        DD       VOLUME=SER=IMSLIB,DISP=SHR,                X
                                   +                             CONTINUE'
                                1183+        PUNCH '//                 DSNAME=ICS.DBDLIB,UNIT=2314'
                                1184+        PUNCH '//SYSUDUMP DD  SYSOUT=A,DCB=(LRECL=121,RECFM=VBA,BLKSIZX
                                   +               E=3129),         X'
                                1185+        PUNCH '//                 SPACE=(125,(2500,100),RLSE,,ROUND)'
                                1186+        PUNCH './         ADD    NAME=IMSMSG'
                                1187+        PUNCH './         NUMBER NEW1=10,INCR=10'
                                1188+        PUNCH '//MESSAGE JOB   1,IMS,MSGLEVEL=1,PRTY=11'
                                1189+        PUNCH '//G        EXEC   PGM=DFSIRC00,PARM=1,REGION=26K'
                                1190+        PUNCH '//STEPLIB DD DSN=ICS.CLOD,DISP=SHR'
                                1191+        PUNCH '//        DD       VOLUME=SER=IMSLIB,DISP=SHR,                X
                                   +                             CONTINUE'
                                1192+        PUNCH '//                 DSNAME=ICS.CLOD,UNIT=2314'
                                1193+        PUNCH '//IMS     DD       VOLUME=SER=IMSLIB,DISP=SHR,                X
                                   +                             CONTINUE'
                                1194+        PUNCH '//                 DSNAME=ICS.PSBLIB,UNIT=2314'
                                1195+        PUNCH '//        DD       VOLUME=SER=IMSLIB,DISP=SHR,                X
                                   +                             CONTINUE'
                                1196+        PUNCH '//                 DSNAME=ICS.DBDLIB,UNIT=2314'
                                1197+        PUNCH '//SYSUDUMP DD  SYSOUT=A,DCB=(LRECL=125,RECFM=VBA,BLKSIZX
                                   +               E=3129),         X'
                                1198+        PUNCH '//                 SPACE=(125,(2500,100),RLSE,,ROUND)'
                                1199+        PUNCH './         ADD    NAME=SECURITY'
                                1200+        PUNCH './         NUMBER NEW1=10,INCR=10'
                                1201+        PUNCH '//         PROC   OPTN=UPDATE,IMS='',0'',SOUT=A'
                                1202+        PUNCH '//S        EXEC   PGM=DFSISMP0,PARM=''&&OPTN.&&IMS.'''
                                1203+        PUNCH '//STEPLIB DD DSN=ICS.CLOD,DISP=SHR'
                                1204+        PUNCH '//SYSPRINT DD SYSOUT=&&SOUT,DCB=(RECFM=VBA,BLKSIZE=400,X
                                   +               BUFL=404)'
                                1205+        PUNCH '//SYSPUNCH DD  UNIT=SYSDA,SPACE=(80,(800,400),,,ROUND),X
                                   +                                X'
                                1206+        PUNCH '//                 DCB=(RECFM=FB,LRECL=80,BLKSIZE=400),DISPX
```

```
LOC  OBJECT CODE    ADDR1 ADDR2  STMT   SOURCE STATEMENT                                      F30SEP69   2/12/70

                                     +               =(,PASS)'
                                  1207+         PUNCH '//SYSLIN   DD  UNIT=SYSDA,SPACE=(TRK,(1,1)),DCB=(RECFM=X
                                     +               F,BLKSIZE=80),'
                                  1208+         PUNCH '//                 DISP=(,PASS)'
                                  1209+         PUNCH '//SYSUT1   DD  UNIT=SYSDA,SPACE=(100,(400,400),,,ROUND)X
                                     +               ,'
                                  1210+         PUNCH '//                 DCB=(BLKSIZE=500,RECFM=FB)'
                                  1211+         PUNCH '//SYSUT2   DD  UNIT=(SYSDA,SEP=SYSUT1),SPACE=(100,(400,X
                                     +               400),,,ROUND),'
                                  1212+         PUNCH '//                 DCB=*.S.SYSUT1'
                                  1213+         PUNCH '//SYSIN    DC  DSN=NO.SYSIN.DD.ASTERISK'
                                  1214+         PUNCH '//C        EXEC  PGM=IEUASM,PARM=''LOAD,NODECK'',COND=(12X
                                     +               ,LT,S),REGION=96K'
                                  1215+         PUNCH '//SYSPRINT DD  SYSOUT=&&SOUT,DCB=(RECFM=FBM,LRECL=121,BX
                                     +               LKSIZE=605)'
                                  1216+         PUNCH '//SYSGO    DD  UNIT=(SYSDA,SEP=SYSPRINT),DISP=(,PASS),'
                                  1217+         PUNCH '//                 DCB=*.S.SYSPUNCH,SPACE=(80,(400,400),,,RX
                                     +               OUND)'
                                  1218+         PUNCH '//SYSUT1   DC  UNIT=SYSDA,SPACE=(CYL,(5,1))'
                                  1219+         PUNCH '//SYSUT2   DD  UNIT=SYSDA,SPACE=(CYL,(5,1))'
                                  1220+         PUNCH '//SYSUT3   DC  UNIT=(SYSDA,SEP=(SYSUT1,SYSUT2)),SPACE=(X
                                     +               CYL,(5,1))'
                                  1221+         PUNCH '//SYSIN    DD  DSN=*.S.SYSPUNCH,DISP=(OLD,DELETE)'
                                  1222+         PUNCH '//L        EXEC  PGM=DFSILNKO,PARM=''XREF,NE,OL'',REGION=X
                                     +               110K,COND=(4,LT,S)'
                                  1223+         PUNCH '//SYSPRINT DD  SYSOUT=&&SOUT,DCB=(RECFM=FBA,LRECL=121,BX
                                     +               LKSIZE=605)'
                                  1224+         PUNCH '//SYSLMOD  DC  DSN=ICS.CLOD,DISP=SHR'
                                  1225+         PUNCH '//INPUT    DD  DSN=*.C.SYSGO,DISP=(OLD,DELETE)'
                                  1226+         PUNCH '//SYSUT1   DC  UNIT=(SYSDA,SEP=INPUT),SPACE=(CYL,(5,1))X
                                     +               '
                                  1227+         PUNCH '//SYSLIN   DD  DSN=*.S.SYSLIN,DISP=(OLD,DELETE)'
                                  1228+         PUNCH './       ENDUP'
                                  1229+         PUNCH '/*'
                                  1230+         PUNCH '//STEP5  EXEC  PGM=IEUASM,PARM=''LOAD,NODECK'',REGION=9X
                                     +               2K'
                                  1231+         PUNCH '//SYSLIB DD    DSNAME=IMS.GENLIB,DISP=(OLD,PASS)'
                                  1232+         PUNCH '//       DD    DSNAME=SYS1.MACLIB,DISP=SHR'
                                  1233+         PUNCH '//SYSGO  DD    UNIT=SYSDA,DISP=(,PASS),DCB=(LRECL=80,BLX
                                     +               KSIZE=400,     X'
                                  1234+         PUNCH '//             RECFM=FB),SPACE=(TRK,(10,10),RLSE)'
                                  1235+         PUNCH '//SYSPRINT DC  SYSOUT=A,DCB=(LRECL=121,BLKSIZE=605,RECFX
                                     +               M=FBA),        X'
                                  1236+         PUNCH '//             SPACE=(605,(100,50),RLSE,,ROUND)'
                                  1237+         PUNCH '//SYSUT1 DD    UNIT=SYSDA,DISP=(,DELETE),SPACE=(1700,(1X
                                     +               00,50))'
                                  1238+         PUNCH '//SYSUT2 DD    UNIT=SYSDA,DISP=(,DELETE),SPACE=(1700,(1X
                                     +               00,50))'
                                  1239+         PUNCH '//SYSUT3 DD    UNIT=(SYSDA,SEP=(SYSLIB,SYSUT1,SYSUT2)),X
                                     +               DISP=(,DELETE), X'
```

```
LOC  OBJECT CODE    ADDR1 ADDR2  STMT   SOURCE STATEMENT                                      F30SEP69   2/12/70

                                  1240+         PUNCH '//                SPACE=(1700,(100,50))'
                                  1241+         PUNCH '//SYSIN  DD       *'
                                  1242+         PUNCH 'DFSISCD  CSECT'
                                  1243+         PUNCH '         PRINT ON'
                                  1244+         PUNCH '         IMSBATCH CENDA=Z8,SPVSVC=243'
                                  1245+         PUNCH '         ISCD     SECTYPE=CSECT'
                                  1246+         PUNCH '         END'
                                  1247+         PUNCH '/*'
                                  1248+         PUNCH '//STEP6 EXEC PGM=IEWL,PARM=''OVLY,NCAL,XREF,LIST'',REGIX
                                     +               ON=110K'
                                  1249+         PUNCH '//SYSPRINT DC  SYSOUT=A,DCB=(LRECL=121,BLKSIZE=605,RECFX
                                     +               M=FBA),        X'
                                  1250+         PUNCH '//               SPACE=(605,(10,10),RLSE,,ROUND)'
                                  1251+         PUNCH '//SYSLIN DD      DSNAME=*.STEP5.SYSGO,DISP=(OLD,DELETE)'
                                  1252+         PUNCH '//       DD      DDNAME=SYSIN'
                                  1253+         PUNCH '//SYSOBJ DD      VOLUME=SER=IMSLIB,DISP=(OLD,PASS),       X
                                     +                         CONTINUE'
                                  1254+         PUNCH '//               DSNAME=ICS.CLOD,UNIT=2314'
                                  1255+         PUNCH '//SYSLMOD DD     VOLUME=SER=IMSLIB,DISP=(OLD,PASS),       X
                                     +                         CONTINUE'
                                  1256+         PUNCH '//               DSNAME=ICS.CLOD,UNIT=2314'
                                  1257+         PUNCH '//SYSUT1 DD      UNIT=(SYSDA,SEP=(SYSLIN,SYSLMOD)),DISP=(X
                                     +               ,DELETE),        X'
                                  1258+         PUNCH '//               SPACE=(1700,(100,50))'
                                  1259+         PUNCH '//SYSIN  DD      *'
                                  1260+         PUNCH '         SETSSI 05012090'
                                  1261+         PUNCH '  INCLUDE SYSOBJ(DFSIDBA0)   DL/I BATCH ANALYZER'
                                  1262+         PUNCH '  INCLUDE SYSOBJ(DFSIDLN0)   DL/I BATCH INITIALIZATION'
                                  1263+         PUNCH '  INCLUDE SYSOBJ(DFSISMM0)   STORAGE MANAGEMENT'
                                  1264+         PUNCH '  INCLUDE SYSOBJ(DFSIOS10)   USAM OPEN ROUTINE'
                                  1265+         PUNCH '  INCLUDE SYSOBJ(DFSIOS60)   OSAM CLOSE ROUTINE'
                                  1266+         PUNCH '  INCLUDE SYSOBJ(DFSIBKB0)   BATCH CONTROL BLOCKS'
                                  1267+         PUNCH '  INCLUDE SYSOBJ(DFSIINL0)   INIT - MODULE LOADER'
                                  1268+         PUNCH '  INCLUDE SYSOBJ(DFSIIN10)   INIT - JOBLIB MODULE LOADEX
                                     +               R'
                                  1269+         PUNCH '  INCLUDE SYSOBJ(DFSIIN20)   INIT - SVCLIB MODULE LOADEX
                                     +               R'
                                  1270+         PUNCH '  INCLUDE SYSOBJ(DFSIDLO0)   DL/I OPEN MODULE'
                                  1271+         PUNCH '  CHANGE DFSICS60(DFSIOS70)  CHG EP TO OSAM CLOSE RTNE'
                                  1272+         PUNCH '  INCLUDE SYSOBJ(DFSIOS60)   OSAM CLOSE RTNE(2ND COPY)'
                                  1273+         PUNCH '  CHANGE DFSIOS60(DFSIOS70)  CHG DLCO REFERENCE'
                                  1274+         PUNCH '  INCLUDE SYSOBJ(DFSIDLC0)   DL/I CLOSE MODULE'
                                  1275+         PUNCH '                 OVERLAY IMSA'
                                  1276+         PUNCH '  INSERT DFSISMM0'
                                  1277+         PUNCH '                 OVERLAY IMSB'
                                  1278+         PUNCH '  INSERT DFSIDLN0'
                                  1279+         PUNCH '  INSERT DFSIINL0'
                                  1280+         PUNCH '  INSERT DFSIIN10'
                                  1281+         PUNCH '  INSERT DFSIIN20'
                                  1282+         PUNCH '                 OVERLAY IMSA'
```

```
LOC  OBJECT CODE    ADDR1 ADDR2  STMT   SOURCE STATEMENT                                                  F30SEP69   2/12/70

                                 1283+        PUNCH '  INSERT DFSIDLCO'
                                 1284+        PUNCH '                  OVERLAY IMSC'
                                 1285+        PUNCH '  INSERT DFSICS60'
                                 1286+        PUNCH '                  OVERLAY IMSD'
                                 1287+        PUNCH '  INSERT DFSIOS10'
                                 1288+        PUNCH '                  OVERLAY IMSA'
                                 1289+        PUNCH '  INSERT DFSIDLCO'
                                 1290+        PUNCH '                  OVERLAY IMSE'
                                 1291+        PUNCH '  INSERT DFSIOS70'
                                 1292+        PUNCH '  ENTRY DFSSTART'
                                 1293+        PUNCH '  NAME DFSIDLB0(R)             DL/I BATCH NUCLEUS'
                                 1294+        PUNCH '/*'
                                 1295+        PUNCH '//STEP7  EXEC  PGM=IEUASM,PARM=''LOAD,NODECK'',REGION=9X
                                   +                2K'
                                 1296+        PUNCH '//SYSLIB DD   DSNAME=IMS.GENLIB,DISP=(SHR,PASS)'
                                 1297+        PUNCH '//       DD    DSNAME=SYS1.MACLIB,DISP=(SHR,PASS)'
                                 1298+        PUNCH '//SYSGO  DD    UNIT=SYSDA,DISP=(,PASS),DCB=(LRECL=80,BLX
                                   +                KSIZE=400,      X'
                                 1299+        PUNCH '//             RECFM=FB),SPACE=(CYL,(1,1),RLSE)'
                                 1300+        PUNCH '//SYSPRINT DD  SYSOUT=A,DCB=(LRECL=121,BLKSIZE=605,RECFX
                                   +                M=FBA),         X'
                                 1301+        PUNCH '//             SPACE=(605,(100,50),RLSE,,ROUND)'
                                 1302+        PUNCH '//SYSUT1 DD    UNIT=SYSDA,DISP=(,DELETE),SPACE=(1700,(1X
                                   +                00,50))'
                                 1303+        PUNCH '//SYSUT2 DD    UNIT=SYSDA,DISP=(,DELETE),SPACE=(1700,(1X
                                   +                00,50))'
                                 1304+        PUNCH '//SYSUT3 DD    UNIT=(SYSDA,SEP=(SYSLIB,SYSUT1,SYSUT2)),X
                                   +                DISP=(,DELETE), X'
                                 1305+        PUNCH '//             SPACE=(1700,(100,50),RLSE)'
                                 1306+        PUNCH '//SYSIN  DD    *'
                                 1307+        PUNCH '         COPY  PCHSSI'
                                 1308+        PUNCH '         PRINT ON'
                                 1309+        PUNCH '         DFSPSBD DFSIBDR0,100,DMBL1,0'
                                 1310+        PUNCH '         DFSPSBD DFSLKM00,100,DMBL2,0'
                                 1311+        PUNCH '         DFSPSBD DFSSAM02,100,DMBL3,0'
                                 1312+        PUNCH '         DFSPSBD DFSSAM03,100,DMBL4,0'
                                 1313+        PUNCH '         DFSPSBD DFSSAM04,100,DMBL5,0'
                                 1314+        PUNCH '         DFSPSBD DFSSAM05,100,DMBL6,0'
                                 1315+        PUNCH '         DFSPSBD DFSSAM06,100,DMBL7,0'
                                 1316+        PUNCH '         DFSPSBD DFSSAM07,100,DMBL8,0'
                                 1317+        PUNCH '         DFSPSBD ENQOSK01,100,0,0'
                                 1318+        PUNCH '         DFSPSBD HIBASK01,010,DMBL10,0'   Batch
                                 1319+        PUNCH '         DFSPSBD HIBLSK01,010,DMBL11,0'
                                 1320+        PUNCH '         DFSPSBD HIMAJC01,100,DMBL12,0'
                                 1321+        PUNCH '         DFSPSBD HIMAJC02,100,DMBL13,0'
                                 1322+        PUNCH '         DFSPSBD HIMAJC03,100,DMBL14,0'
                                 1323+        PUNCH '         DFSPSBD HIMARJC1,100,DMBL15,0'
                                 1324+        PUNCH '         DFSPSBD HIMASN01,100,DMBL16,0'
                                 1325+        PUNCH '         DFSPSBD HITASK01,100,DMBL17,0'
                                 1326+        PUNCH '         DFSPSBD HITASK02,100,DMBL18,0'
```

```
LOC  OBJECT CODE    ADDR1 ADDR2  STMT   SOURCE STATEMENT                                                  F30SEP69   2/12/70

                                 1327+        PUNCH '         DFSPSBD HSBASK01,010,DMBL19,0'
                                 1328+        PUNCH '         DFSPSBD HSTASK01,100,DMBL20,0'
                                 1329+        PUNCH '         DFSPSBD NOPSB,100,DMBL21,0'
                                 1330+        PUNCH '         DFSPSBD SWITCH,100,DMBL22,1'
                                 1331+        PUNCH 'DMBL1      DFSDMBL DFSIBDRT,01,1'
                                 1332+        PUNCH 'DMBL2      DFSDMBL DI31PH01,00,1'
                                 1333+        PUNCH 'DMBL3      DFSDMBL DI21PART,00,1'  share
                                 1334+        PUNCH 'DMBL4      DFSDMBL DI21PART,00,1'  "
                                 1335+        PUNCH 'DMBL5      DFSDMBL DI21PART,10,1'  update
                                 1336+        PUNCH 'DMBL6      DFSDMBL DI21PART,10,1'
                                 1337+        PUNCH 'DMBL7      DFSDMBL DI21PART,10,1'
                                 1338+        PUNCH 'DMBL8      DFSDMBL DI21PART,00,1'  share
                                 1339+        PUNCH 'DMBL10     DFSDMBL DI31SK01,10,0'
                                 1340+        PUNCH '           DFSDMBL DI32SK01,10,1'
                                 1341+        PUNCH 'DMBL11     DFSDMBL DI31SK01,10,0'
                                 1342+        PUNCH '           DFSDMBL DI32SK01,10,1'
                                 1343+        PUNCH 'DMBL12     DFSDMBL DS40JC01,10,1'
                                 1344+        PUNCH 'DMBL13     DFSDMBL DS40JC01,10,1'
                                 1345+        PUNCH 'DMBL14     DFSDMBL DI31PH01,10,1'
                                 1346+        PUNCH 'DMBL15     DFSDMBL DI21IRJE,10,1'
                                 1347+        PUNCH 'DMBL16     DFSDMBL DI31PH01,10,0'
                                 1348+        PUNCH '           DFSDMBL DI31PH02,10,1'
                                 1349+        PUNCH 'DMBL17     DFSDMBL DI31SK01,10,1'
                                 1350+        PUNCH 'DMBL18     DFSDMBL DI32SK01,10,1'
                                 1351+        PUNCH 'DMBL19     DFSDMBL DS31SK01,10,1'
                                 1352+        PUNCH 'DMBL20     DFSDMBL DS31SK01,10,1'
                                 1353+        PUNCH 'DMBL21     DFSDMBL DI31PH01,00,1'
                                 1354+        PUNCH 'DMBL22     DFSDMBL DI31PH01,00,1'
                                 1355+        PUNCH '          ENTRY DFSIDMD0'
                                 1356+        PUNCH 'DFSIBDRT  DFSDMD DFSIBDRT,00000000'
                                 1357+        PUNCH 'DI21IRJE  DFSDMD DI21IRJE,00000010'
                                 1358+        PUNCH 'DI21PART  DFSDMD DI21PART,00000010'
                                 1359+        PUNCH 'DI31PH01  DFSDMD DI31PH01,00000010'
                                 1360+        PUNCH 'DI31PH02  DFSDMD DI31PH02,00000010'
                                 1361+        PUNCH 'DI31SK01  DFSDMD DI31SK01,00000010'
                                 1362+        PUNCH 'DI32SK01  DFSDMD DI32SK01,00000010'
                                 1363+        PUNCH 'DS31SK01  DFSDMD DS31SK01,00000010'
                                 1364+        PUNCH 'DS40JC01  DFSDMD DS40JC01,00000010'
                                 1365+        PUNCH '          END '
                                 1366+        PUNCH '/*'
                                 1367+        PUNCH '//STEP8 EXEC PGM=IEWL,PARM=''REUS,NCAL,XREF,LIST'',REGIX
                                   +                ON=110K'
                                 1368+        PUNCH '//SYSPRINT DD  SYSOUT=A,DCB=(LRECL=121,BLKSIZE=605,RECFX
                                   +                M=FBA),         X'
                                 1369+        PUNCH '//             SPACE=(605,(10,10),RLSE,,ROUND)'
                                 1370+        PUNCH '//SYSLIN DD DSNAME=*.STEP7.SYSGO,DISP=(OLD,DELETE)'
                                 1371+        PUNCH '//SYSOBJ DD   DSNAME=ICS.CLOD,DISP=(OLD,PASS)'
                                 1372+        PUNCH '//SYSLMOD DD  DSNAME=ICS.CLOD(DFSIDIR0),DISP=(OLD,PASS)X
                                   +                '
                                 1373+        PUNCH '//SYSUT1 DD    UNIT=(SYSDA,SEP=(SYSLIN,SYSLMOD)),DISP=(X
LOC  OBJECT CODE    ADDR1 ADDR2  STMT   SOURCE STATEMENT                                      F30SEP69   2/12/70

                                   +              ,DELETE),          X'
                                1374+          PUNCH '//                 SPACE=(1700,(100,50))'
                                1375+          PUNCH '//STEP9  EXEC   PGM=IEUASM,PARM=''LOAD,NODECK'',REGION=9X
                                   +              2K'
                                1376+          PUNCH '//SYSLIB DD   DSNAME=IMS.GENLIB,DISP=(SHR,PASS)'
                                1377+          PUNCH '//          DD    DSNAME=SYS1.MACLIB,DISP=(SHR,PASS)'
                                1378+          PUNCH '//SYSGO   DD    UNIT=SYSDA,DISP=(,PASS),DCB=(LRECL=80,BLX
                                   +              KSIZE=400,         X'
                                1379+          PUNCH '//                 RECFM=FB),SPACE=(CYL,(1,1),RLSE)'
                                1380+          PUNCH '//SYSPRINT DD  SYSOUT=A,DCB=(LRECL=121,BLKSIZE=605,RECFX
                                   +              M=FBA),            X'
                                1381+          PUNCH '//                 SPACE=(605,(100,50),RLSE,,ROUND)'
                                1382+          PUNCH '//SYSUT1 DD    UNIT=SYSDA,DISP=(,DELETE),SPACE=(1700,(1X
                                   +              00,50))'
                                1383+          PUNCH '//SYSUT2 DD    UNIT=SYSDA,DISP=(,DELETE),SPACE=(1700,(1X
                                   +              00,50))'
                                1384+          PUNCH '//SYSUT3 DD    UNIT=(SYSDA,SEP=(SYSLIB,SYSUT1,SYSUT2)),X
                                   +              DISP=(,DELETE), X'
                                1385+          PUNCH '//                 SPACE=(1700,(100,50),RLSE)'
                                1386+          PUNCH '//SYSIN   DD      *'
                                1387+          PUNCH '           COPY   PCHSSI'
                                1388+          PUNCH '           PRINT  ON'
                                1389+          PUNCH '           DFSSMB 10,6,01000101,14,448,#,65535,65535'
                                1390+          PUNCH '           DFSSMB 7,5,01000101,9,128,ADDI,65535,65535'
                                1391+          PUNCH '           DFSSMB 7,5,01000101,9,128,ADDINV,65535,65535'
                                1392+          PUNCH '           DFSSMB 7,5,01000101,9,128,ADDPART,65535,65535'
                                1393+          PUNCH '           DFSSMB 7,5,01000101,9,128,ADDPN,65535,65535'
                                1394+          PUNCH '           DFSSMB 1,65535,00000101,1,160,CLOSE,65535,6553X
                                   +              5'
                                1395+          PUNCH '           DFSSMB 7,5,01000101,9,160,CLSORD,65535,65535'
                                1396+          PUNCH '           DFSSMB 5,5,01000001,12,32,DFS,8,100'
                                1397+          PUNCH '           DFSSMB 0,65535,01000101,15,0,DFSIBDRS,65535,65X
                                   +              535'
                                1398+          PUNCH '           DFSSMB 9,2,01000101,10,192,DISB,65535,65535'
                                1399+          PUNCH '           DFSSMB 1,65535,00000101,1,192,DISBURSE,65535,6X
                                   +              5535'
                                1400+          PUNCH '           DFSSMB 5,2,01000101,7,128,DLETI,65535,65535'
                                1401+          PUNCH '           DFSSMB 5,2,01000101,7,128,DLETINV,65535,65535'
                                1402+          PUNCH '           DFSSMB 5,2,01000101,7,128,DLETPART,65535,65535X
                                   +              '
                                1403+          PUNCH '           DFSSMB 5,2,01000101,7,128,DLETPN,65535,65535'
                                1404+          PUNCH '           DFSSMB 5,5,10000101,10,480,DLI,10,10'
                                1405+          PUNCH '           DFSSMB 0,3,01000101,8,480,DLN,10,100'
                                1406+          PUNCH '           DFSSMB 1,65535,00000101,1,224,DSPALLI,65535,65X
                                   +              535'
                                1407+          PUNCH '           DFSSMB 1,65535,00000101,1,96,DSPINV,65535,6553X
                                   +              5'
                                1408+          PUNCH '           DFSSMB 1,65535,00000101,1,64,DSPPN,65535,65535X
                                   +              '
                                1409+          PUNCH '           DFSSMB 8,65535,01000101,8,256,ENQ,65535,65535'
```

```
LOC  OBJECT CODE    ADDR1 ADDR2  STMT   SOURCE STATEMENT                                      F30SEP69   2/12/70

                                1410+          PUNCH '           DFSSMB 5,5,01000101,12,480,ICS,10,100'
                                1411+          PUNCH '           DFSSMB 2,10,00000101,5,480,IMS,1,100'
                                1412+          PUNCH '           DFSSMB 1,65535,00000101,1,96,INVTORY,65535,655X
                                   +              35'
                                1413+          PUNCH '           DFSSMB 1,1,01000001,1,640,NOP,5,50'
                                1414+          PUNCH '           DFSSMB 1,65535,00000101,1,64,PART,65535,65535'
                                1415+          PUNCH '           DFSSMB 2,10,01000101,4,448,RJE,65535,65535'
                                1416+          PUNCH '           DFSSMB 8,65535,01000101,8,608,SKH1,65535,65535X
                                   +              '
                                1417+          PUNCH '           DFSSMB 8,65535,01000101,8,512,SKI1,65535,65535X
                                   +              '
                                1418+          PUNCH '           DFSSMB 8,65535,01000101,8,544,SKI2,65535,65535X
                                   +              '
                                1419+          PUNCH '           DFSSMB 1,1000,01000001,7,672,SWI,5,1'
                                1420+          PUNCH '           DFSSMB 5,4,01000001,5,672,SWIBR,20,100'
                                1421+          PUNCH '           DFSSMB 4,1,01000001,6,672,SWIPASS,20,100'
                                1422+          PUNCH '           DFSSMB 14,100,01000001,14,672,SWIPR,20,100'
                                1423+          PUNCH '           DFSSMB 4,1,01000001,6,672,SWITS,20,100'
                                1424+          PUNCH '           DFSSMB 0,4,01000001,4,672,SWN,5,100'
                                1425+          PUNCH '           DFSSMB 0,65535,01000101,0,320,SW1,65535,65535'
                                1426+          PUNCH '           DFSSMB 0,1000,01000101,0,288,SW2,65535,65535'
                                1427+          PUNCH '           DFSSMB 8,65535,01000101,8,352,TPPL1,65535,6553X
                                   +              5'
                                1428+          PUNCH '           DFSSMB 8,65535,01000101,8,384,TPPL2,65535,6553X
                                   +              5'
                                1429+          PUNCH '           DFSSMB 8,65535,01000101,8,416,TUBE,65535,65535X
                                   +              '
                                1430+          PUNCH '           END'
                                1431+          PUNCH '/*'
                                1432+          PUNCH '//STEP10 EXEC PGM=IEWL,PARM=''REUS,NCAL,XREF,LIST'',REGX
                                   +              ION=110K'
                                1433+          PUNCH '//SYSPRINT DD  SYSOUT=A,DCB=(LRECL=121,BLKSIZE=605,RECFX
                                   +              M=FBA),            X'
                                1434+          PUNCH '//                 SPACE=(605,(10,10),RLSE,,ROUND)'
                                1435+          PUNCH '//SYSLIN DD DSNAME=*.STEP9.SYSGO,DISP=(OLD,DELETE)'
                                1436+          PUNCH '//SYSOBJ DD   DSNAME=ICS.CLOD,DISP=(OLD,PASS)'
                                1437+          PUNCH '//SYSLMOD DD  DSNAME=ICS.CLOD(DFSISMBO),DISP=(OLD,PASS)X
                                   +              '
                                1438+          PUNCH '//SYSUT1 DD    UNIT=(SYSDA,SEP=(SYSLIN,SYSLMOD)),DISP=(X
                                   +              ,DELETE),          X'
                                1439+          PUNCH '//                 SPACE=(1700,(100,50))'
                                1440+          PUNCH '//STEP11  EXEC  PGM=IEUASM,PARM=''LOAD,NODECK'',REGION=X
                                   +              92K'
                                1441+          PUNCH '//SYSLIB DD   DSNAME=IMS.GENLIB,DISP=(SHR,PASS)'
                                1442+          PUNCH '//          DD    DSNAME=SYS1.MACLIB,DISP=(SHR,PASS)'
                                1443+          PUNCH '//SYSGO   DD    UNIT=SYSDA,DISP=(,PASS),DCB=(LRECL=80,BLX
                                   +              KSIZE=400,         X'
                                1444+          PUNCH '//                 RECFM=FB),SPACE=(CYL,(1,1),RLSE)'
                                1445+          PUNCH '//SYSPRINT DD  SYSOUT=A,DCB=(LRECL=121,BLKSIZE=605,RECFX
                                   +              M=FBA),            X'
```

```
     IMS/360 SYSTEM DEFINITION SPECIFICATIONS                                                                 PAGE  37

LCC  CBJECT CCDE    ADDR1 ADDR2  STMT   SOURCE STATEMENT                                                 F30SEP69   2/12/70

                                 1446+         PUNCH '//                SPACF=(605,(100,50),RLSE,,ROUND)'
                                 1447+         PUNCH '//SYSUT1 DD       UNIT=SYSDA,DISP=(,DELETE),SPACE=(1700,(1X
                                     +               00,50))'
                                 1448+         PUNCH '//SYSUT2 CC       UNIT=SYSCA,DISP=(,DELETE),SPACE=(17CC,(1X
                                     +               00,50))'
                                 1449+         PUNCH '//SYSUT3 DD       UNIT=(SYSDA,SEP=(SYSLIB,SYSUT1,SYSUT2)),X
                                     +               DISP=(,DELETE), X'
                                 1450+         PUNCH '//                SPACE=(1700,(100,50),RLSE)'
                                 1451+         PUNCH '//SYSIN  CC       *'
                                 1452+         PUNCH '          CCPY  PCHSSI'
                                 1453+         PUNCH '          PRINT ON'
                                 1454+         PUNCH '          DFSCLB 1,1,1,1,8008,1,00,0   SYSTEM CONSOLE CLX
                                     +               B'
                                 1455+         PUNCH '.DFCL1  ANOP'
                                 1456+         PUNCH '          DFSCLB 2,1,1,1,80E8,0,28,0'
                                 1457+         PUNCH '          AGO   .DFCL2'
                                 1458+         PUNCH '.DFPL1 ANOP'
                                 1459+         PUNCH 'DFPL1   DFTRMLST  WRAPLST,(E2) '
                                 1460+         PUNCH '          AGO   .DFPL2'
                                 1461+         PUNCH '.CFCL2  ANOP'
                                 1462+         PUNCH '          DFSCLB 3,1,2,2,8008,0,56,0'
                                 1463+         PUNCH '          AGO   .DFCL3'
                                 1464+         PUNCH '.DFPL2 ANOP'
                                 1465+         PUNCH 'DFPL2   DFTRMLST  WRAPLST,(E2) '
                                 1466+         PUNCH '          AGO   .DFPL3'
                                 1467+         PUNCH '.DFCL3  ANOP'
                                 1468+         PUNCH '          DFSCLB 4,1,3,3,80E8,0,84,0'
                                 1469+         PUNCH '          AGO   .DFCL4'
                                 1470+         PUNCH '.DFPL3 ANCP'
                                 1471+         PUNCH 'DFPL3   DFTRMLST  WRAPLST,(E2,E4) '
                                 1472+         PUNCH '          AGO   .DFPL4'
                                 1473+         PUNCH '.LERB1 ANOP'
                                 1474+         PUNCH 'LERB1  LERB  3,(200,10,5,5)'
                                 1475+         PUNCH '          AGO   .LERB2'
                                 1476+         PUNCH '.DFCL4  ANCP'
                                 1477+         PUNCH '          DFSCLB 5,2,4,1,8CE8,0,140,0'
                                 1478+         PUNCH '          AGC   .DFCL5'
                                 1479+         PUNCH '.DFPL4 ANOP'
                                 1480+         PUNCH 'DFPL4  DFTRMLST DIALST,0,(E2)'
                                 1481+         PUNCH '          AGC   .DFPL5'
                                 1482+         PUNCH '.LERB2 ANOP'
                                 1483+         PUNCH 'LERB2  LERB  1,(200,10,5,5)'
                                 1484+         PUNCH '          AGO   .LERB3'
                                 1485+         PUNCH '.DFCL5  ANOP'
                                 1486+         PUNCH '          DFSCLB 6,3,5,1,8CE8,0,168,0'
                                 1487+         PUNCH '          AGC   .DFCL6'
                                 1488+         PUNCH '.DFPL5 ANOP'
                                 1489+         PUNCH 'DFPL5  DFTRMLST DIALST,0,(E215)'
                                 1490+         PUNCH '          AGC   .DFPL6'
                                 1491+         PUNCH '.DFCL6  ANOP'
```

```
     IMS/360 SYSTEM DEFINITION SPECIFICATIONS                                                                 PAGE  38

LCC  OBJECT CCDE    ADDR1 ADDR2  STMT   SOURCE STATEMENT                                                 F30SEP69   2/12/70

                                 1492+         PUNCH '          DFSCLB 7,3,6,2,80E8,0,196,0'
                                 1493+         PUNCH '          AGO   .DFCL7'
                                 1494+         PUNCH '.DFPL6 ANOP'
                                 1495+         PUNCH 'DFPL6  DFTRMLST DIALST,0,(0015)'
                                 1496+         PUNCH '          AGO   .DFPL7'
                                 1497+         PUNCH '.LERB3 ANOP'
                                 1498+         PUNCH 'LERB3  LERB  2,(200,10,5,5)'
                                 1499+         PUNCH '          AGO   .LERB4'
                                 1500+         PUNCH '.DFCL7  ANOP'
                                 1501+         PUNCH '          DFSCLB 8,4,7,1,80E8,0,280,0'
                                 1502+         PUNCH '          AGO   .DFCL8'
                                 1503+         PUNCH '.DFPL7 ANOP'
                                 1504+         PUNCH 'DFPL7   DFTRMLST  WRAPLST,(E215) '
                                 1505+         PUNCH '          AGO   .DFPL8'
                                 1506+         PUNCH '.LERB4 ANOP'
                                 1507+         PUNCH 'LERB4  LERB  1,(200,10,5,5)'
                                 1508+         PUNCH '          AGO   .LERB5'
                                 1509+         PUNCH '.DFCL8  ANOP'
                                 1510+         PUNCH '          DFSCLB 9,5,8,1,80E8,0,308,0'
                                 1511+         PUNCH '          TITLE ''DFSICLLO - COMMUNICATION LINE POLLING X
                                     +               LISTS '' '
                                 1512+         PUNCH '          AGO   .DFPL1'
                                 1513+         PUNCH '.DFPL8 ANOP'
                                 1514+         PUNCH 'DFPL8   DFTRMLST  WRAPLST,(40FF) '
                                 1515+         PUNCH '          TITLE ''DFSICLLO - COMMUNICATION LINE ERROR BLX
                                     +               OCKS'' '
                                 1516+         PUNCH '          AGO   .LERB1'
                                 1517+         PUNCH '.LERB5 ANOP'
                                 1518+         PUNCH 'LERB5  LERB  1,(200,10,5,5)'
                                 1519+         PUNCH '          TITLE ''DFSICLLO - COMMUNICATION LINE GROUP DCX
                                     +               B''''S '' '
                                 1520+         PUNCH 'DFSDCB1 DCB CSORG=CX,MACRF=(R,W),ERROPT=CTRW,LERB=LERB1X
                                     +               ,        CONTINUE'
                                 1521+         PUNCH '                CCNAME=CC2740S'
                                 1522+         PUNCH '                EJECT'
                                 1523+         PUNCH 'DFSDCB2 CCB DSORG=CX,MACRF=(R,W),ERROPT=CTRW,LERB=LERB2X
                                     +               ,        CONTINUE'
                                 1524+         PUNCH '                DDNAME=DD2740A'
                                 1525+         PUNCH '                EJECT'
                                 1526+         PUNCH 'DFSDCB3 DCB CSORG=CX,MACRF=(R,W),ERROPT=CTRW,LERB=LERB3X
                                     +               ,        CONTINUE'
                                 1527+         PUNCH '                DDNAME=DD1C50A'
                                 1528+         PUNCH '                EJECT'
                                 1529+         PUNCH 'DFSCCB4 DCB DSORG=CX,MACRF=(R,W),ERROPT=CTRW,LERB=LERB4X
                                     +               ,        CONTINUE'
                                 1530+         PUNCH '                DDNAME=DD1050'
                                 1531+         PUNCH '                EJECT'
                                 1532+         PUNCH 'DFSCCB5 CCB CSORG=CX,MACRF=(R,W),FRROPT=CTRW,LERB=LERB5X
                                     +               ,        CONTINUE'
                                 1533+         PUNCH '                DDNAME=DD2260T'
```

```
LOC  OBJECT CODE    ADDR1 ADDR2  STMT   SOURCE STATEMENT                                          F30SEP69   2/12/70

                                 1534+           PUNCH '           TITLE ''DFSICCBO - COMMUNICATION LINE OPEN LISX
                                    +                  T'' '
                                 1535+           PUNCH 'DFSICDBO CSECT'
                                 1536+           PUNCH '           ENTRY DFSICDB'
                                 1537+           PUNCH 'DFSICDB  DS     OD'
                                 1538+           PUNCH '           DC     AL1(0)'
                                 1539+           PUNCH '           DC     AL3(DFSDCB1)'
                                 1540+           PUNCH '           DC     AL1(0)'
                                 1541+           PUNCH '           DC     AL3(CFSDCB2)'
                                 1542+           PUNCH '           DC     AL1(0)'
                                 1543+           PUNCH '           DC     AL3(DFSDCB3)'
                                 1544+           PUNCH '           DC     AL1(0)'
                                 1545+           PUNCH '           DC     AL3(DFSDCB4)'
                                 1546+           PUNCH '           DC     AL1(128)'
                                 1547+           PUNCH '           DC     AL3(CFSDCB5)' .
                                 1548+           PUNCH '                  END'
                                 1549+           PUNCH '/*'
                                 1550+           PUNCH '//STEP12 EXEC PGM=IEWL,PARM=''REUS,NCAL,XREF,LIST'',REGX
                                    +                  ION=110K'
                                 1551+           PUNCH '//SYSPRINT DC  SYSOUT=A,DCB=(LRECL=121,BLKSIZE=605,RECFX
                                    +                  M=FBA),         X'
                                 1552+           PUNCH '//             SPACE=(605,(10,10),RLSE,,RCUND)'
                                 1553+           PUNCH '//SYSLIN DD DSNAME=*.STEP11.SYSGO,DISP=(OLD,DELETE)'
                                 1554+           PUNCH '//SYSOBJ DD   DSNAME=ICS.CLCD,DISP=(OLD,PASS)'
                                 1555+           PUNCH '//SYSLMOD DD  DSNAME=ICS.CLOD(DFSICLLO),DISP=(OLD,PASS)X
                                    +                  '
                                 1556+           PUNCH '//SYSUT1 DD     UNIT=(SYSDA,SEP=(SYSLIN,SYSLMOD)),DISP=(X
                                    +                  ,DELETE),        X'
                                 1557+           PUNCH '//              SPACE=(1700,(100,50))'
                                 1558+           PUNCH '//STEP13  EXEC  PGM=IEUASM,PARM=''LOAD,NODECK'',REGION=X
                                    +                  92K'
                                 1559+           PUNCH '//SYSLIB DD   DSNAME=IMS.GENLIB,DISP=(SHR,PASS)'
                                 1560+           PUNCH '//       DD     DSNAME=SYS1.MACLIB,DISP=(SHR,PASS)'
                                 1561+           PUNCH '//SYSGO  DD     UNIT=SYSDA,DISP=(,PASS),DCB=(LRECL=80,BLX
                                    +                  KSIZE=400,       X'
                                 1562+           PUNCH '//              RECFM=FB),SPACE=(CYL,(1,1),RLSE)'
                                 1563+           PUNCH '//SYSPRINT DC   SYSOUT=A,DCB=(LRECL=121,BLKSIZE=605,RECFX
                                    +                  M=FBA),          X'
                                 1564+           PUNCH '//              SPACE=(6C5,(1C0,50),RLSE,,ROUND)'
                                 1565+           PUNCH '//SYSUT1 DD     UNIT=SYSDA,DISP=(,DELETE),SPACE=(1700,(1X
                                    +                  00,50))'
                                 1566+           PUNCH '//SYSUT2 DD     UNIT=SYSCA,DISP=(,DELETE),SPACE=(17C0,(1X
                                    +                  00,50))'
                                 1567+           PUNCH '//SYSUT3 DD     UNIT=(SYSDA,SEP=(SYSLIB,SYSUT1,SYSUT2)),X
                                    +                  DISP=(,DELETE),  X'
                                 1568+           PUNCH '//              SPACE=(1700,(100,50),RLSE)'
                                 1569+           PUNCH '//SYSIN  DD     *'
                                 1570+           PUNCH '           CCPY  PCHSSI'
                                 1571+           PUNCH '           PRINT CN'
                                 1572+           PUNCH 'BILL       DFSCNT 0000,0,308,BILL,65535'
```

```
LCC  OBJECT CODE    ADDR1 ADDR2  STMT   SCURCE STATEMENT                                          F30SEP69   2/12/70

                                 1573+           PUNCH 'BUD      DFSCNT 0000,0,392,BUD,65535'
                                 1574+           PUNCH 'CARDPNCH DFSCNT 0000,3,280,CARDPNCH,MODEL2M-DFSICNT'
                                 1575+           PUNCH 'CARL     DFSCNT 0000,0,364,CARL,65535'
                                 1576+           PUNCH 'CAROL    DFSCNT 0002,0,196,CAROL,65535'
                                 1577+           PUNCH 'DAN      DFSCNT 0002,0,224,DAN,HOWARD-DFSICNT'
                                 1578+           PUNCH 'ELEANOR  DFSCNT 0002,0,224,ELEANOR,DAN-DFSICNT'
                                 1579+           PUNCH 'ERNE     DFSCNT 0000,0,336,ERNE,65535'
                                 1580+           PUNCH 'HOWARD   DFSCNT 0002,0,224,HOWARD,65535'
                                 1581+           PUNCH 'INQUIRY1 DFSCNT 0001,0,140,INQUIRY1,65535'
                                 1582+           PUNCH 'INQUIRY2 DFSCNT 0001,0,168,INQUIRY2,65535'
                                 1583+           PUNCH 'JOE      DFSCNT 0002,0,252,JOE,65535'
                                 1584+           PUNCH 'LEONARD  DFSCNT 0000,0,336,LEONARD,ERNE-DFSICNT'
                                 1585+           PUNCH 'L2740SM1 DFSCNT 0000,0,84,L2740SM1,65535'
                                 1586+           PUNCH 'L2740SM2 DFSCNT 0000,0,112,L2740SM2,65535'
                                 1587+           PUNCH 'L2740S1  DFSCNT 0000,0,56,L2740S1,65535'
                                 1588+           PUNCH 'L2740S2  DFSCNT 000C,C,28,L2740S2,65535'
                                 1589+           PUNCH 'MASTER   DFSCNT 4000,0,56,MASTER,L2740S1-DFSICNT'
                                 1590+           PUNCH 'MODEL2   DFSCNT 0000,2,280,MODEL2,CARDPNCH-DFSICNT'
                                 1591+           PUNCH 'MODEL2M  DFSCNT 0000,3,280,MODEL2M,65535'
                                 1592+           PUNCH 'PRINTER  DFSCNT 0000,0,280,PRINTER,T2780-DFSICNT'
                                 1593+           PUNCH 'RICHARD  DFSCNT 0002,0,252,RICHARD,JOE-DFSICNT'
                                 1594+           PUNCH 'SHARRON  DFSCNT 0002,0,252,SHARRON,RICHARD-DFSICNT'
                                 1595+           PUNCH 'TAPEPNCH DFSCNT 0000,2,280,TAPEPNCH,MODEL2-DFSICNT'
                                 1596+           PUNCH 'T2780    DFSCNT 0000,0,280,T2780,TAPEPNCH-DFSICNT'
                                 1597+           PUNCH 'WTOR     DFSCNT 0000,0,0,WTOR,65535'
                                 1598+           PUNCH '         TITLE ''DFSICTMO - COMMUNICATION TERMINAL MATRX
                                    +                  IX'' '
                                 1599+           PUNCH 'DFSICTMO CSECT'
                                 1600+           PUNCH '         ENTRY DFSICTM'
                                 1601+           PUNCH 'DFSICTM  DS    CD'
                                 1602+           PUNCH 'CTMROW1  DC    B''00000000'' '
                                 1603+           PUNCH '         DC    B''0C000000'' '
                                 1604+           PUNCH '         DC    B''01000000'' '
                                 1605+           PUNCH '         DC    B''01000000'' '
                                 1606+           PUNCH 'CTMROW2  DC    B''11111111'' '
                                 1607+           PUNCH '         DC    B''11111111'' '
                                 1608+           PUNCH '         DC    B''11111111'' '
                                 1609+           PUNCH '         DC    B''1C000000'' '
                                 1610+           PUNCH '         END'
                                 1611+           PUNCH '/*'
                                 1612+           PUNCH '//STEP14 EXEC PGM=IEWL,PARM=''REUS,NCAL,XREF,LIST'',REGX
                                    +                  ION=110K'
                                 1613+           PUNCH '//SYSPRINT DC  SYSOLT=A,DCB=(LRECL=121,BLKSIZE=605,RECFX
                                    +                  M=FBA),         X'
                                 1614+           PUNCH '//             SPACE=(605,(10,10),RLSE,,ROUND)'
                                 1615+           PUNCH '//SYSLIN DD DSNAME=*.STEP13.SYSGO,DISP=(OLD,DELETE)'
                                 1616+           PUNCH '//SYSOBJ DD   DSNAME=ICS.CLOD,DISP=(OLD,PASS)'
                                 1617+           PUNCH '//SYSLMCO DD  DSNAME=ICS.CLCD(DFSICNTO),DISP=(OLD,PASS)X
                                    +                  '
                                 1618+           PUNCH '//SYSUT1 DD     UNIT=(SYSDA,SEP=(SYSLIN,SYSLMOD)),DISP=(X
LOC  OBJECT CODE    ADDR1 ADDR2  STMT   SOURCE STATEMENT                                           F30SEP69   2/12/70

                                     +               ,DELETE),       X'
                                 1619+         PUNCH '//             SPACE=(1700,(100,50))'
                                 1620+         PUNCH '//STEP15  EXEC  PGM=IEUASM,PARM=''LOAD,NODECK'',REGION=X
                                     +               92K'
                                 1621+         PUNCH '//SYSLIB DD   DSNAME=IMS.GENLIB,DISP=(SHR,PASS)'
                                 1622+         PUNCH '//       CD    DSNAME=SYS1.MACLIB,DISP=(SHR,PASS)'
                                 1623+         PUNCH '//SYSGO  DD    UNIT=SYSDA,DISP=(,PASS),DCB=(LRECL=80,BLX
                                     +               KSIZE=400,      X'
                                 1624+         PUNCH '//             RECFM=FB),SPACE=(CYL,(1,1),RLSE)'
                                 1625+         PUNCH '//SYSPRINT DD  SYSOLT=A,DCB=(LRECL=121,BLKSIZE=605,RECFX
                                     +               M=FBA),         X'
                                 1626+         PUNCH '//             SPACE=(605,(100,50),RLSE,,ROUND)'
                                 1627+         PUNCH '//SYSUT1 DD    UNIT=SYSDA,DISP=(,DELETE),SPACE=(1700,(1X
                                     +               00,50))'
                                 1628+         PUNCH '//SYSUT2 DD    UNIT=SYSCA,CISP=(,CELETE),SPACE=(1700,(1X
                                     +               00,50))'
                                 1629+         PUNCH '//SYSUT3 CC    UNIT=(SYSDA,SEP=(SYSLIB,SYSUT1,SYSUT2)),X
                                     +               DISP=(,DELETE), X'
                                 1630+         PUNCH '//             SPACE=(17CC,(100,50),RLSE)'
                                 1631+         PUNCH '//SYSIN  DD    *'
                                 1632+         PUNCH '         COPY  PCHSSI'
                                 1633+         PUNCH '         PRINT ON'
                                 1634+         PUNCH '         DFSCTB 0,1,E215,0000,1600,1,1050'
                                 1635+         PUNCH '      DFSCTB 2,2,E281,0000,1024,2,2740'
                                 1636+         PUNCH '         ENTRY DFSCTBMT'
                                 1637+         PUNCH 'DFSCTBMT EQU   *'
                                 1638+         PUNCH '      DFSCTB 2,3,E281,4000,1088,3,2740'
                                 1639+         PUNCH '      DFSCTB 2,4,E281,0C0C,832,4,2740'
                                 164C+         PUNCH '      DFSCTB 2,4,E481,0000,896,5,2740'
                                 1641+         PUNCH '      DFSCTB 4,5,E281,CC00,576,6,2740'
                                 1642+         PUNCH '      DFSCTB 5,6,E202,0000,640,7,1050'
                                 1643+         PUNCH '      DFSCTB 5,7,0002,2000,256,8,1050,NONE'
                                 1644+         PUNCH '      DFSCTB 5,7,0002,2000,384,9,1050,NONE'
                                 1645+         PUNCH '      DFSCTB 5,7,0002,2000,1408,10,1050,NONE'
                                 1646+         PUNCH '      DFSCTB 7,8,E2C2,0000,1280,11,1050'
                                 1647+         PUNCH '      DFSCTB 0,9,40A0,0000,0,12,2260'
                                 1648+         PUNCH '      DFSCTB C,9,40A1,0000,768,13,2260'
                                 1649+         PUNCH '      DFSCTB C,9,4CA2,0000,192,14,2260'
                                 165C+         PUNCH '      DFSCTB 0,9,40A3,0000,64,15,2260'
                                 1651+         PUNCH '         END'
                                 1652+         PUNCH '/*'
                                 1653+         PUNCH '//STEP16 EXEC PGM=IEWL,FARM=''REUS,NCAL,XREF,LIST'',REGX
                                     +               ION=110K'
                                 1654+         PUNCH '//SYSPRINT DC  SYSOUT=A,DCB=(LRECL=121,BLKSIZE=605,RECFX
                                     +               M=FBA),         X'
                                 1655+         PUNCH '//             SPACE=(605,(10,10),RLSE,,RCUND)'
                                 1656+         PUNCH '//SYSLIN DD DSNAME=*.STEP15.SYSGO,DISP=(OLD,DELETE)'
                                 1657+         PUNCH '//SYSOBJ DD   DSNAME=ICS.CLCD,DISP=(OLD,PASS)'
                                 1658+         PUNCH '//SYSLMOD DD  DSNAME=ICS.CLCD(DFSICTBO),DISP=(OLD,PASS)X
                                     +               '
```

```
LOC  OBJECT CODE    ADDR1 ADDR2  STMT   SOURCE STATEMENT                                           F30SEP69   2/12/70

                                 1659+         PUNCH '//SYSUT1 DD    UNIT=(SYSDA,SEP=(SYSLIN,SYSLMOD)),DISP=(X
                                     +               ,DELETE),       X'
                                 166C+         PUNCH '//             SPACE=(1700,(100,50))'
                                 1661+         PUNCH '//STEP17  EXEC  PGM=IEUASM,PARM=''LOAD,NODECK'',REGION=X
                                     +               92K'
                                 1662+         PUNCH '//SYSLIB DD   DSNAME=IMS.GENLIB,DISP=(SHR,PASS)'
                                 1663+         PUNCH '//       DD    DSNAME=SYS1.MACLIB,DISP=(SHR,PASS)'
                                 1664+         PUNCH '//SYSGO  DD    UNIT=SYSDA,DISP=(,PASS),DCB=(LRECL=80,BLX
                                     +               KSIZE=400,      X'
                                 1665+         PUNCH '//             RECFM=FB),SPACE=(CYL,(1,1),RLSE)'
                                 1666+         PUNCH '//SYSPRINT DC  SYSOUT=A,DCB=(LRECL=121,BLKSIZE=605,RECFX
                                     +               M=FBA),         X'
                                 1667+         PUNCH '//             SPACE=(605,(1C0,50),RLSE,,ROUND)'
                                 1668+         PUNCH '//SYSUT1 DD    UNIT=SYSDA,DISP=(,DELETE),SPACE=(1700,(1X
                                     +               00,50))'
                                 1669+         PUNCH '//SYSUT2 DD    UNIT=SYSCA,DISP=(,DELETE),SPACE=(1700,(1X
                                     +               00,50))'
                                 1670+         PUNCH '//SYSUT3 DD    UNIT=(SYSDA,SEP=(SYSLIB,SYSUT1,SYSUT2)),X
                                     +               DISP=(,DELETE), X'
                                 1671+         PUNCH '//             SPACE=(1700,(100,50),RLSE)'
                                 1672+         PUNCH '//SYSIN  DD    *'
                                 1673+         PUNCH '         COPY  PCHSSI'
                                 1674+         PUNCH '         PRINT CN'
                                 1675+         PUNCH 'CTT2260  ICTTG IDLTAB=(40,80),IDLNL=(0A,01),IDLLF=(40,8X
                                     +               0),LLINE=84,    X'
                                 1676+         PUNCH '               LTC=C,OPT=(TERMINAL,VALID,FIRST,TUBE,VALX
                                     +               ID,NOCARRG,     X'
                                 1677+         PUNCH '               STACTL,NOSWITCH),TREC=IECTRSCI,TSND=IECTX
                                     +               SSCI,NTRYL=3,   X'
                                 1678+         PUNCH '               SPACE=SPCRT,BUFSZ=1000'
                                 1679+         PUNCH 'CTT7770 ICTTG OPT=(TERMINAL,VALID,FIRST,NOTUBE,VALID,NOX
                                     +               CARRG,STACTL,   X'
                                 168C+         PUNCH '               SWITCHED),LTC=2,TREC=IECTRF40,BUFSZ=132'
                                 1681+         PUNCH 'CTT2740S ICTTG TREC=IECTRF40,NTRYL=2'
                                 1682+         PUNCH 'CTT2740N ICTTG OPT=(TERMINAL,VALID,FIRST,NOTUBE,INVALIDX
                                     +               ,CARRG,         X'
                                 1683+         PUNCH '               NOSTACTL,NOSWITCH),TREC=IECTRF40,NTRYL=0X
                                     +               '
                                 1684+         PUNCH 'CTT2740A ICTTG OPT=(TERMINAL,INVALID,FIRST,NOTUBE,INVALX
                                     +               ID,CARRG,       X'
                                 1685+         PUNCH '               NOSTACTL,SWITCHED),TREC=IECTRF40,NTRYL=2X
                                     +               '
                                 1686+         PUNCH 'CTT1050A ICTTG OPT=(COMPONT,INVALID,FIRST,NOTUBE,INVALIX
                                     +               D,CARRG,STACTL, X'
                                 1687+         PUNCH '               SWITCHED),TREC=IECTRF50,TSND=IECTSD50,NTX
                                     +               RYL=3'
                                 1688+         PUNCH 'CTT1050N ICTTG OPT=(COMPONT,INVALID,FIRST,NOTUBE,INVALIX
                                     +               D,CARRG,        X'
                                 1689+         PUNCH '               NOSTACTL,NOSWITCH),TREC=IECTRF50,TSND=IEX
                                     +               CTSD50,NTRYL=3'
```

```
LOC  OBJECT CODE    ADDR1 ADDR2  STMT   SOURCE STATEMENT                                           F30SEP69   2/12/70

                                 1690+         PUNCH 'CTT1050S ICTTG OPT=(CCMPONT,INVALID,FIRST,NOTUBE,INVALIX
                                     +               D,CARRG,STACTL, X'
                                 1691+         PUNCH '               NOSWITCH),TREC=IECTRF50,TSND=IECTSD50,NTX
                                     +               RYL=3'
                                 1692+         PUNCH 'CTT2740B ICTTG TREC=IECTRF40,NTRYL=3'
                                 1693+         PUNCH '         ASMTFTAB RF40,SD50,RF50,RSCI '
                                 1694+         PUNCH '         COPY  TRATABLE '
                                 1695+         PUNCH '         END '
                                 1696+         PUNCH '/*'
                                 1697+         PUNCH '//STEP18 EXEC PGM=IEWL,PARM=''REUS,NCAL,XREF,LIST'',REGX
                                     +               ION=110K'
                                 1698+         PUNCH '//SYSPRINT DD  SYSOUT=A,DCB=(LRECL=121,BLKSIZE=605,RECFX
                                     +               M=FBA),         X'
                                 1699+         PUNCH '//             SPACE=(605,(10,10),RLSE,,ROUND)'
                                 1700+         PUNCH '//SYSLIN DD DSNAME=*.STEP17.SYSGO,DISP=(OLD,DELETE)'
                                 1701+         PUNCH '//SYSOBJ DD   DSNAME=ICS.CLOD,DISP=(OLD,PASS)'
                                 1702+         PUNCH '//SYSLMOD DD  DSNAME=ICS.CLOD(DFSICTTO),DISP=(OLD,PASS)X
                                     +               '
                                 1703+         PUNCH '//SYSUT1 DD    UNIT=(SYSDA,SEP=(SYSLIN,SYSLMOD)),DISP=(X
                                     +               ,DELETE),       X'
                                 1704+         PUNCH '//             SPACE=(1700,(100,50))'
                                 1705+         PUNCH '//STEP19  EXEC  PGM=IEUASM,PARM=''LOAD,NODECK'',REGION=X
                                     +               92K'
                                 1706+         PUNCH '//SYSLIB DD   DSNAME=IMS.GENLIB,DISP=(SHR,PASS)'
                                 1707+         PUNCH '//       DD    DSNAME=SYS1.MACLIB,DISP=(SHR,PASS)'
                                 1708+         PUNCH '//SYSGO  DD    UNIT=SYSDA,DISP=(,PASS),DCB=(LRECL=80,BLX
                                     +               KSIZE=400,      X'
                                 1709+         PUNCH '//             RECFM=FB),SPACE=(CYL,(1,1),RLSE)'
                                 1710+         PUNCH '//SYSPRINT DD  SYSOUT=A,DCB=(LRECL=121,BLKSIZE=605,RECFX
                                     +               M=FBA),         X'
                                 1711+         PUNCH '//             SPACE=(605,(100,50),RLSE,,ROUND)'
                                 1712+         PUNCH '//SYSUT1 DD    UNIT=SYSDA,DISP=(,DELETE),SPACE=(1700,(1X
                                     +               00,50))'
                                 1713+         PUNCH '//SYSUT2 DD    UNIT=SYSDA,DISP=(,DELETE),SPACE=(1700,(1X
                                     +               00,50))'
                                 1714+         PUNCH '//SYSUT3 DD    UNIT=(SYSDA,SEP=(SYSLIB,SYSUT1,SYSUT2)),X
                                     +               DISP=(,DELETE), X'
                                 1715+         PUNCH '//             SPACE=(1700,(100,50),RLSE)'
                                 1716+         PUNCH '//SYSIN  DD    *'
                                 1717+         PUNCH '         COPY  PCHSSI'
                                 1718+         PUNCH '         PRINT ON'
                                 1719+         PUNCH '         CVBG  /START,CL40,CTM=0'
                                 1720+         PUNCH '         CVBG  /STOP,CL40,CTM=0'
                                 1721+         PUNCH '         CVBG  /PSTOP,CL40,CTM=0'
                                 1722+         PUNCH '         CVBG  /PURGE,CL40,CTM=0'
                                 1723+         PUNCH '         CVBG  /DISPLAY,CLD0,CTM=0'
                                 1724+         PUNCH '         CVBG  /RDISPLAY,CLD0'
                                 1725+         PUNCH '         CVBG  /CHANGE,CL60,CTM=0'
                                 1726+         PUNCH '         CVBG  /ASSIGN,CL70,CTM=0'
                                 1727+         PUNCH '         CVBG  /DELETE,CL80,CTM=0'
```

```
LOC  OBJECT CODE    ADDR1 ADDR2  STMT   SOURCE STATEMENT                                           F30SEP69   2/12/70

                                 1728+         PUNCH '         CVBG  /BROADCAST,CL10,CTM=4'
                                 1729+         PUNCH '         CVBG  /CHECKPOINT,CL20,CTM=0'
                                 1730+         PUNCH '         CVBG  /DBDUMP,CL20,CTM=0'
                                 1731+         PUNCH '         CVBG  /LOCK,CL90'
                                 1732+         PUNCH '         CVBG  /UNLOCK,CL90'
                                 1733+         PUNCH '         CVBG  /TEST,CL50,TP=80'
                                 1734+         PUNCH '         CVBG  /EXCLUSIVE,CL50,TP=80 '
                                 1735+         PUNCH '         CVBG  /END,CL50,TP=80'
                                 1736+         PUNCH '         CVBG  /LOG,CLP0'
                                 1737+         PUNCH '         CVBG  /CANCEL,CLP0,TP=C0 '
                                 1738+         PUNCH '         CVBG  /DBLOG,CL40,CTM=0'
                                 1739+         PUNCH '         CVBG  /DBNOLOG,CL40,CTM=0'
                                 1740+         PUNCH '         CVBG  /NRESTART,CL20,TP=20,CTM=0'
                                 1741+         PUNCH '         CVBG  /ERESTART,CL20,TP=20,CTM=0'
                                 1742+         PUNCH '         CVBG  /DBRECOVERY,CL20,CTM=0'
                                 1743+         PUNCH '         CVBG  /IAM,CLA0'
                                 1744+         PUNCH '         CVBG  /SET,CLE0'
                                 1745+         PUNCH '         CVBG  /RESET,CLE0,TP=80 '
                                 1746+         PUNCH '         END'
                                 1747+         PUNCH '/*'
                                 1748+         PUNCH '//STEP20 EXEC PGM=IEWL,PARM=''REUS,NCAL,XREF,LIST'',REGX
                                     +               ION=110K'
                                 1749+         PUNCH '//SYSPRINT DD  SYSOUT=A,DCB=(LRECL=121,BLKSIZE=605,RECFX
                                     +               M=FBA),         X'
                                 1750+         PUNCH '//             SPACE=(605,(10,10),RLSE,,ROUND)'
                                 1751+         PUNCH '//SYSLIN DD DSNAME=*.STEP19.SYSGO,DISP=(OLD,DELETE)'
                                 1752+         PUNCH '//SYSOBJ DD   DSNAME=ICS.CLOD,DISP=(OLD,PASS)'
                                 1753+         PUNCH '//SYSLMOD DD  DSNAME=ICS.CLOD(DFSICVBO),DISP=(OLD,PASS)X
                                     +               '
                                 1754+         PUNCH '//SYSUT1 DD    UNIT=(SYSDA,SEP=(SYSLIN,SYSLMOD)),DISP=(X
                                     +               ,DELETE),       X'
                                 1755+         PUNCH '//             SPACE=(1700,(100,50))'
                                 1756+         PUNCH '//STEP21  EXEC  PGM=IEUASM,PARM=''LOAD,NODECK'',REGION=X
                                     +               92K'
                                 1757+         PUNCH '//SYSLIB DD   DSNAME=IMS.GENLIB,DISP=(SHR,PASS)'
                                 1758+         PUNCH '//       DD    DSNAME=SYS1.MACLIB,DISP=(SHR,PASS)'
                                 1759+         PUNCH '//SYSGO  DD    UNIT=SYSDA,DISP=(,PASS),DCB=(LRECL=80,BLX
                                     +               KSIZE=400,      X'
                                 1760+         PUNCH '//             RECFM=FB),SPACE=(CYL,(1,1),RLSE)'
                                 1761+         PUNCH '//SYSPRINT DD  SYSOUT=A,DCB=(LRECL=121,BLKSIZE=605,RECFX
                                     +               M=FBA),         X'
                                 1762+         PUNCH '//             SPACE=(605,(100,50),RLSE,,ROUND)'
                                 1763+         PUNCH '//SYSUT1 DD    UNIT=SYSDA,DISP=(,DELETE),SPACE=(1700,(1X
                                     +               00,50))'
                                 1764+         PUNCH '//SYSUT2 DD    UNIT=SYSDA,DISP=(,DELETE),SPACE=(1700,(1X
                                     +               00,50))'
                                 1765+         PUNCH '//SYSUT3 DD    UNIT=(SYSDA,SEP=(SYSLIB,SYSUT1,SYSUT2)),X
                                     +               DISP=(,DELETE), X'
                                 1766+         PUNCH '//             SPACE=(1700,(100,50),RLSE)'
                                 1767+         PUNCH '//SYSIN  DD    *'
```

```
LOC  OBJECT CODE    ADDR1 ADDR2  STMT   SOURCE STATEMENT                                            F30SEP69   2/12/70

                                 1768+         PUNCH '          COPY  PCHSSI'
                                 1769+         PUNCH '          PRINT ON'
                                 1770+         PUNCH '         TITLE ''DFSISDB0 - SECURITY DIRECTORY BLOCKS'''
                                 1771+         PUNCH 'DFSISDB0 CSECT'
                                 1772+         PUNCH '          ENTRY DFSISDB'
                                 1773+         PUNCH 'DFSISDB  DS    0D'
                                 1774+         PUNCH '          DC    A(SMBL-DFSISDB)     OFFSET TO SMB LIST'
                                 1775+         PUNCH '          DC    A(CNTL-DFSISDB)     OFFSET TO CNT LIST'
                                 1776+         PUNCH '          DC    A(DMDL-DFSISDB)     OFFSET TO DMD LIST'
                                 1777+         PUNCH '          DC    A(PSBL-DFSISDB)     OFFSET TO PSB LIST'
                                 1778+         PUNCH '          DC    A(CVBL-DFSISDB)     OFFSET TO CVB LIST'
                                 1779+         PUNCH '          DC    A(CTBL-DFSISDB)     OFFSET TO CTB LIST'
                                 1780+         PUNCH '          DC    A(MASTER-DFSISDB)  OFFSET TO MASTER CNT'
                                 1781+         PUNCH '          DC    A((CTBL-DFSISDB)+((2-1)*5))     OFFSET TOX
                                     +               MASTER CTB'
                                 1782+         PUNCH 'SMBL     DC    AL2((SMBLE-SMBLL)/L''SMBLL,L''SMBLL) '
                                 1783+         PUNCH 'SMBLL    DS    0CL8'
                                 1784+         PUNCH '          DC    CL8''#'' '
                                 1785+         PUNCH '          DC    CL8''ADDI'' '
                                 1786+         PUNCH '          DC    CL8''ADDINV'' '
                                 1787+         PUNCH '          DC    CL8''ADDPART'' '
                                 1788+         PUNCH '          DC    CL8''ADDPN'' '
                                 1789+         PUNCH '          DC    CL8''CLOSE'' '
                                 1790+         PUNCH '          DC    CL8''CLSORD'' '
                                 1791+         PUNCH '          DC    CL8''DFS'' '
                                 1792+         PUNCH '          DC    CL8''DFSIBORS'' '
                                 1793+         PUNCH '          DC    CL8''DISB'' '
                                 1794+         PUNCH '          DC    CL8''DISBURSE'' '
                                 1795+         PUNCH '          DC    CL8''DLETI'' '
                                 1796+         PUNCH '          DC    CL8''DLETINV'' '
                                 1797+         PUNCH '          DC    CL8''DLETPART'' '
                                 1798+         PUNCH '          DC    CL8''DLETPN'' '
                                 1799+         PUNCH '          DC    CL8''DLI'' '
                                 1800+         PUNCH '          DC    CL8''DLN'' '
                                 1801+         PUNCH '          DC    CL8''DSPALLI'' '
                                 1802+         PUNCH '          DC    CL8''DSPINV'' '
                                 1803+         PUNCH '          DC    CL8''DSPPN'' '
                                 1804+         PUNCH '          DC    CL8''ENQ'' '
                                 1805+         PUNCH '          DC    CL8''ICS'' '
                                 1806+         PUNCH '          DC    CL8''IMS'' '
                                 1807+         PUNCH '          DC    CL8''INVTORY'' '
                                 1808+         PUNCH '          DC    CL8''NOP'' '
                                 1809+         PUNCH '          DC    CL8''PART'' '
                                 1810+         PUNCH '          DC    CL8''RJE'' '
                                 1811+         PUNCH '          DC    CL8''SKH1'' '
                                 1812+         PUNCH '          DC    CL8''SKI1'' '
                                 1813+         PUNCH '          DC    CL8''SKI2'' '
                                 1814+         PUNCH '          DC    CL8''SWI'' '
                                 1815+         PUNCH '          DC    CL8''SWIBR'' '
                                 1816+         PUNCH '          DC    CL8''SWIPASS'' '
```

```
LOC  OBJECT CODE    ADDR1 ADDR2  STMT   SOURCE STATEMENT                                            F30SEP69   2/12/70

                                 1817+         PUNCH '          DC    CL8''SWIPR'' '
                                 1818+         PUNCH '          DC    CL8''SWITS'' '
                                 1819+         PUNCH '          DC    CL8''SWN'' '
                                 1820+         PUNCH '          DC    CL8''SW1'' '
                                 1821+         PUNCH '          DC    CL8''SW2'' '
                                 1822+         PUNCH '          DC    CL8''TPPL1'' '
                                 1823+         PUNCH '          DC    CL8''TPPL2'' '
                                 1824+         PUNCH '          DC    CL8''TUBE'' '
                                 1825+         PUNCH 'SMBLE    EQU   *'
                                 1826+         PUNCH 'CNTL     DC    AL2((CNTLE-CNTLL)/L''CNTLL,L''CNTLL) '
                                 1827+         PUNCH 'CNTLL    DS    0CL8'
                                 1828+         PUNCH '          DC   CL8''BILL'' '
                                 1829+         PUNCH '          DC   CL8''BUD'' '
                                 1830+         PUNCH '          DC   CL8''CARDPNCH'' '
                                 1831+         PUNCH '          DC   CL8''CARL'' '
                                 1832+         PUNCH '          DC   CL8''CAROL'' '
                                 1833+         PUNCH '          DC   CL8''DAN'' '
                                 1834+         PUNCH '          DC   CL8''ELEANOR'' '
                                 1835+         PUNCH '          DC   CL8''ERNE'' '
                                 1836+         PUNCH '          DC   CL8''HOWARD'' '
                                 1837+         PUNCH '          DC   CL8''INQUIRY1'' '
                                 1838+         PUNCH '          DC   CL8''INQUIRY2'' '
                                 1839+         PUNCH '          DC   CL8''JOE'' '
                                 1840+         PUNCH '          DC   CL8''LEONARD'' '
                                 1841+         PUNCH '          DC   CL8''L2740SM1'' '
                                 1842+         PUNCH '          DC   CL8''L2740SM2'' '
                                 1843+         PUNCH '          DC   CL8''L2740S1'' '
                                 1844+         PUNCH '          DC   CL8''L2740S2'' '
                                 1845+         PUNCH 'MASTER   DC   CL8''MASTER'' '
                                 1846+         PUNCH '          DC   CL8''MODEL2'' '
                                 1847+         PUNCH '          DC   CL8''MODEL2M'' '
                                 1848+         PUNCH '          DC   CL8''PRINTER'' '
                                 1849+         PUNCH '          DC   CL8''RICHARD'' '
                                 1850+         PUNCH '          DC   CL8''SHARRON'' '
                                 1851+         PUNCH '          DC   CL8''TAPEPNCH'' '
                                 1852+         PUNCH '          DC   CL8''T2780'' '
                                 1853+         PUNCH '          DC   CL8''WTOR'' '
                                 1854+         PUNCH 'CNTLE    EQU    *'
                                 1855+         PUNCH 'DMDL     DC     AL2((DMDLE-DMDLL)/L''DMDLL,L''DMDLL) '
                                 1856+         PUNCH 'DMDLL    DS     0CL8'
                                 1857+         PUNCH '          DC    CL8''DFSIBDRT'' '
                                 1858+         PUNCH '          DC    CL8''DI21IRJE'' '
                                 1859+         PUNCH '          DC    CL8''DI21PART'' '
                                 1860+         PUNCH '          DC    CL8''DI31PH01'' '
                                 1861+         PUNCH '          DC    CL8''DI31PH02'' '
                                 1862+         PUNCH '          DC    CL8''DI31SK01'' '
                                 1863+         PUNCH '          DC    CL8''DI32SK01'' '
                                 1864+         PUNCH '          DC    CL8''DS31SK01'' '
                                 1865+         PUNCH '          DC    CL8''DS40JC01'' '
                                 1866+         PUNCH 'DMDLE    EQU    *'
```

```
LCC  OBJECT CODE    ADDR1 ADDR2  STMT   SOURCE STATEMENT                                            F30SEP69   2/12/70

                                 1867+         PUNCH 'PSBL     DC    AL2((PSBLE-PSBLL)/L''PSBLL,L''PSBLL)
                                 1868+         PUNCH 'PSBLL    DS    OCL8'
                                 1869+         PUNCH '         DC    CL8''DFSIBDRO'' '
                                 1870+         PUNCH '         DC    CL8''DFSLKMOO'' '
                                 1871+         PUNCH '         DC    CL8''DFSSAM02'' '
                                 1872+         PUNCH '         DC    CL8''DFSSAM03'' '
                                 1873+         PUNCH '         DC    CL8''DFSSAM04'' '
                                 1874+         PUNCH '         DC    CL8''DFSSAM05'' '
                                 1875+         PUNCH '         DC    CL8''DFSSAM06'' '
                                 1876+         PUNCH '         CC    CL8''DFSSAM07'' '
                                 1877+         PUNCH '         DC    CL8''ENQOSK01'' '
                                 1878+         PUNCH '         DC    CL8''HIBASK01'' '
                                 1879+         PUNCH '         CC    CL8''HIBLSK01'' '
                                 1880+         PUNCH '         DC    CL8''HIMAJC01'' '
                                 1881+         PUNCH '         DC    CL8''HIMAJC02'' '
                                 1882+         PUNCH '         DC    CL8''HIMAJC03'' '
                                 1883+         PUNCH '         DC    CL8''HIMARJ01'' '
                                 1884+         PUNCH '         CC    CL8''HIMASN01'' '
                                 1885+         PUNCH '         DC    CL8''HITASK01'' '
                                 1886+         PUNCH '         DC    CL8''HITASK02'' '
                                 1887+         PUNCH '         DC    CL8''HSBASK01'' '
                                 1888+         PUNCH '         DC    CL8''HSTASK01'' '
                                 1889+         PUNCH '         DC    CL8''NOPSB'' '
                                 1890+         PUNCH '         DC    CL8''SWITCH'' '
                                 1891+         PUNCH 'PSBLE    EQU   *'
                                 1892+         PUNCH 'CVBL     DC    AL2((CVBLE-CVBLL)/L''CVBLL,L''CVBLL) '
                                 1893+         PUNCH 'CVBLL    DS    OCL10'
                                 1894+         PUNCH '         DC    CL10''START'''
                                 1895+         PUNCH '         DC    CL10''STOP'''
                                 1896+         PUNCH '         DC    CL10''PSTOP'''
                                 1897+         PUNCH '         DC    CL10''PURGE'''
                                 1898+         PUNCH '         DC    CL10''DISPLAY'''
                                 1899+         PUNCH '         DC    CL10''RDISPLAY'''
                                 1900+         PUNCH '         DC    CL10''CHANGE'''
                                 1901+         PUNCH '         DC    CL10''ASSIGN'''
                                 1902+         PUNCH '         DC    CL10''DELETE'''
                                 1903+         PUNCH '         DC    CL10''BROADCAST'''
                                 1904+         PUNCH '         DC    CL10''CHECKPOINT'''
                                 1905+         PUNCH '         DC    CL10''DBDUMP'''
                                 1906+         PUNCH '         DC    CL10''LOCK'''
                                 1907+         PUNCH '         DC    CL10''UNLOCK'''
                                 1908+         PUNCH '         DC    CL10''TEST'''
                                 1909+         PUNCH '         DC    CL10''EXCLUSIVE'''
                                 1910+         PUNCH '         DC    CL10''END'''
                                 1911+         PUNCH '         DC    CL10''LOG'''
                                 1912+         PUNCH '         DC    CL10''CANCEL'''
                                 1913+         PUNCH '         DC    CL10''DBLOG'''
                                 1914+         PUNCH '         CC    CL10''DBNOLOG'''
                                 1915+         PUNCH '         DC    CL10''NRESTART'''
                                 1916+         PUNCH '         DC    CL10''ERESTART'''
```

```
LCC  OBJECT CCDE    ADDR1 ADDR2  STMT   SOURCE STATEMENT                                            F30SEP69   2/12/70

                                 1917+         PUNCH '         DC    CL10''DBRECOVERY'''
                                 1918+         PUNCH '         DC    CL10''IAM'''
                                 1919+         PUNCH '         DC    CL10''SET'''
                                 1920+         PUNCH '         DC    CL10''RESET'''
                                 1921+         PUNCH 'CVBLE    EQU   *'
                                 1922+         PUNCH '         DS    OF'
                                 1923+         PUNCH 'CTBL     DC    AL2(15,5)'
                                 1924+         PUNCH '         DC    C''001E2''   S/360 OPERATOR''S CONSOLE'
                                 1925+         PUNCH '         DC    C''002E2'' '
                                 1926+         PUNCH '         DC    C''003E2'' '
                                 1927+         PUNCH '         DC    C''004E2'' '
                                 1928+         PUNCH '         DC    C''004E4'' '
                                 1929+         PUNCH '         DC    C''005E2'' '
                                 1930+         PUNCH '         DC    C''006E2'' '
                                 1931+         PUNCH '         DC    C''00700'' '
                                 1932+         PUNCH '         DC    C''00700'' '
                                 1933+         PUNCH '         DC    C''00700'' '
                                 1934+         PUNCH '         DC    C''008E2'' '
                                 1935+         PUNCH '         DC    C''009A0'' '
                                 1936+         PUNCH '         DC    C''CC9A1'' '
                                 1937+         PUNCH '         DC    C''009A2'' '
                                 1938+         PUNCH '         DC    C''009A3'' '
                                 1939+         PUNCH '         END'
                                 1940+         PUNCH '/*'
                                 1941+         PUNCH '//STEP22 EXEC PGM=IEWL,PARM=''REUS,NCAL,XREF,LIST'',REGX
                                     +               ION=110K'
                                 1942+         PUNCH '//SYSPRINT DD  SYSOUT=A,DCB=(LRECL=121,BLKSIZE=605,RECFX
                                     +               M=FBA),        X'
                                 1943+         PUNCH '//             SPACE=(605,(10,10),RLSE,,ROUND)'
                                 1944+         PUNCH '//SYSLIN DD DSNAME=*.STEP21.SYSGO,DISP=(OLD,DELETE)'
                                 1945+         PUNCH '//SYSOBJ DD   DSNAME=ICS.CLOD,DISP=(OLD,PASS)'
                                 1946+         PUNCH '//SYSLMCD DD  DSNAME=ICS.CLOD(DFSISDBO),DISP=(OLD,PASS)X
                                     +               '
                                 1947+         PUNCH '//SYSUT1 DD    UNIT=( SYSDA,SEP=( SYSLIN,SYSLMOD)),DISP=(X
                                     +               ,DELETE),      X'
                                 1948+         PUNCH '//             SPACE=(1700,(100,50))'
                                 1949+         PUNCH '//STEP23  EXEC  PGM=IEUASM,PARM=''LOAD,NODECK'',REGION=X
                                     +               92K'
                                 1950+         PUNCH '//SYSLIB DD   DSNAME=IMS.GENLIB,DISP=(SHR,PASS)'
                                 1951+         PUNCH '//       DD    DSNAME=SYS1.MACLIB,DISP=(SHR,PASS)'
                                 1952+         PUNCH '//SYSGO  DD    UNIT=SYSDA,DISP=(,PASS),DCB=(LRECL=80,BLX
                                     +               KSIZE=400,     X'
                                 1953+         PUNCH '//             RECFM=FB),SPACE=(CYL,(1,1),RLSE)'
                                 1954+         PUNCH '//SYSPRINT DC  SYSOUT=A,DCB=(LRECL=121,BLKSIZE=605,RECFX
                                     +               M=FBA),        X'
                                 1955+         PUNCH '//             SPACE=(605,(100,50),RLSE,,ROUND)'
                                 1956+         PUNCH '//SYSUT1 DD    UNIT=SYSDA,DISP=(,DELETE),SPACE=(1700,(1X
                                     +               00,50))'
                                 1957+         PUNCH '//SYSUT2 DD    UNIT=SYSDA,DISP=(,DELETE),SPACE=(1700,(1X
                                     +               00,50))'
```

```
LCC  OBJECT CCDE     ADDR1 ADDR2  STMT   SOURCE STATEMENT                                      F30SEP69    2/12/70

                                  1958+           PUNCH '//SYSUT3 DD     UNIT=(SYSDA,SEP=(SYSLIB,SYSUT1,SYSUT2)),X
                                     +                  DISP=(,DELETE),  X'
                                  1959+           PUNCH '//              SPACE=(1700,(100,50),RLSE)'
                                  1960+           PUNCH '//SYSIN   DD    *'
                                  1961+           PUNCH '          CCPY  PCHSSI'
                                  1962+           PUNCH '          PRINT ON'
                                  1963+           PUNCH '          DFSAVARA 12,EVENTS=14,SECTYPF=CSECT'
                                  1964+           PUNCH '          END'
                                  1965+           PUNCH '/*'
                                  1966+           PUNCH '//STEP24 EXEC PGM=IEWL,PARM=''REUS,NCAL,XREF,LIST'',REGX
                                     +                  ION=110K'
                                  1967+           PUNCH '//SYSPRINT DC  SYSOLT=A,DCB=(LRECL=121,BLKSIZE=605,RECFX
                                     +                  M=FBA),          X'
                                  1968+           PUNCH '//              SPACE=(605,(10,10),RLSE,,RCUND)'
                                  1969+           PUNCH '//SYSLIN DD DSNAME=*.STEP23.SYSGO,DISP=(OLD,DELETE)'
                                  197C+           PUNCH '//SYSOBJ DC    CSNAME=ICS.CLCD,DISP=(OLD,PASS)'
                                  1971+           PUNCH '//SYSLMOD DD  DSNAME=ICS.CLCD(DFSISAVO),DISP=(OLD,PASS)X
                                     +                  '
                                  1972+           PUNCH '//SYSUT1 DC     UNIT=(SYSDA,SEP=(SYSLIN,SYSLMOD)),DISP=(X
                                     +                  ,DELETE),        X'
                                  1973+           PUNCH '//              SPACE=(17CC,(1C0,5C))'
                                  1974+           PUNCH '//STEP25  EXEC  PGM=IEUASM,PARM=''LCAD,NODECK'',REGION=X
                                     +                  92K'
                                  1975+           PUNCH '//SYSLIB DC    CSNAME=IMS.GENLIB,DISP=(SHR,PASS)'
                                  1976+           PUNCH '//       DD     DSNAME=SYS1.MACLIB,DISP=(SHR,PASS)'
                                  1977+           PUNCH '//SYSGO  DD     UNIT=SYSDA,DISP=(,PASS),DCB=(LRECL=80,BLX
                                     +                  KSIZE=400,       X'
                                  1978+           PUNCH '//              RECFM=FB),SPACE=(CYL,(1,1),RLSE)'
                                  1979+           PUNCH '//SYSPRINT DD  SYSOUT=A,DCB=(LRECL=121,BLKSIZE=605,RECFX
                                     +                  M=FBA),          X'
                                  198C+           PUNCH '//              SPACE=(605,(1C0,50),RLSE,,RCUND)'
                                  1981+           PUNCH '//SYSUT1 DD     UNIT=SYSDA,DISP=(,DELETE),SPACE=(17C0,(1X
                                     +                  00,50))'
                                  1982+           PUNCH '//SYSUT2 DD     UNIT=SYSDA,DISP=(,CELETE),SPACE=(17C0,(1X
                                     +                  00,50))'
                                  1983+           PUNCH '//SYSUT3 DD     UNIT=(SYSDA,SEP=(SYSLIB,SYSUT1,SYSUT2)),X
                                     +                  DISP=(,DELETE),  X'
                                  1984+           PUNCH '//              SPACE=(17CC,(100,50),RLSE)'
                                  1985+           PUNCH '//SYSIN   DD    *'
                                  1986+           PUNCH '          COPY  PCHSSI'
                                  1987+           PUNCH '          PRINT ON'
                                  1988+           PUNCH '          DFSIPST REGICNS=3'
                                  1989+           PUNCH '          END'
                                  199C+           PUNCH '/*'
                                  1991+           PUNCH '//STEP26 EXEC PGM=IEWL,PARM=''REUS,NCAL,XREF,LIST'',REGX
                                     +                  ION=110K'
                                  1992+           PUNCH '//SYSPRINT DC  SYSOUT=A,DCB=(LRECL=121,BLKSIZE=605,RECFX
                                     +                  M=FBA),          X'
                                  1993+           PUNCH '//              SPACE=(6C5,(10,10),RLSE,,ROUND)'
                                  1994+           PUNCH '//SYSLIN DD DSNAME=*.STEP25.SYSGO,DISP=(OLD,DELETE)'
```

```
LOC  OBJECT CODE     ADDR1 ADDR2  STMT   SOURCE STATEMENT                                      F30SEP69    2/12/70

                                  1995+           PUNCH '//SYSOBJ DD    DSNAME=ICS.CLOD,DISP=(OLD,PASS)'
                                  1996+           PUNCH '//SYSLMOD DD  DSNAME=ICS.CLOD(DFSIPSTO),DISP=(OLD,PASS)X
                                     +                  '
                                  1997+           PUNCH '//SYSUT1 DD     UNIT=(SYSDA,SEP=(SYSLIN,SYSLMOD)),DISP=(X
                                     +                  ,DELETE),        X'
                                  1998+           PUNCH '//              SPACE=(1700,(100,50))'
                                  1999+           PUNCH '//STEP27  EXEC  PGM=IEUASM,PARM=''LOAD,NODECK'',REGION=X
                                     +                  92K'
                                  2000+           PUNCH '//SYSLIB DD    DSNAME=IMS.GENLIB,DISP=(SHR,PASS)'
                                  2001+           PUNCH '//       DD     DSNAME=SYS1.MACLIB,DISP=(SHR,PASS)'
                                  2002+           PUNCH '//SYSGO  DD     UNIT=SYSDA,DISP=(,PASS),CCB=(LRECL=80,BLX
                                     +                  KSIZE=400,       X'
                                  2003+           PUNCH '//              RECFM=FB),SPACE=(CYL,(1,1),RLSE)'
                                  2004+           PUNCH '//SYSPRINT DD  SYSOUT=A,DCB=(LRECL=121,BLKSIZE=605,RECFX
                                     +                  M=FBA),          X'
                                  2005+           PUNCH '//              SPACE=(605,(100,50),RLSE,,ROUND)'
                                  2006+           PUNCH '//SYSUT1 DD     UNIT=SYSDA,DISP=(,DELETE),SPACE=(1700,(1X
                                     +                  00,50))'
                                  2007+           PUNCH '//SYSUT2 DD     UNIT=SYSDA,DISP=(,DELETE),SPACE=(17CC,(1X
                                     +                  00,50))'
                                  2008+           PUNCH '//SYSUT3 DD     UNIT=(SYSDA,SEP=(SYSLIB,SYSUT1,SYSUT2)),X
                                     +                  DISP=(,DELETE),  X'
                                  2009+           PUNCH '//              SPACE=(17C0,(100,50),RLSE)'
                                  2010+           PUNCH '//SYSIN   DD    *'
                                  2011+           PUNCH '          COPY  PCHSSI'
                                  2012+           PUNCH '          PRINT ON'
                                  2013+           PUNCH '          DFSCUEUE TASK=3,LINES=9'
                                  2014+           PUNCH '          ENC'
                                  2015+           PUNCH '/*'
                                  2016+           PUNCH '//STEP28 EXEC PGM=IEWL,PARM=''REUS,NCAL,XREF,LIST'',REGX
                                     +                  ION=110K'
                                  2017+           PUNCH '//SYSPRINT DD  SYSOLT=A,DCB=(LRECL=121,BLKSIZE=605,RECFX
                                     +                  M=FBA),          X'
                                  2018+           PUNCH '//              SPACE=(605,(10,10),RLSE,,RCUND)'
                                  2019+           PUNCH '//SYSLIN DD DSNAME=*.STEP27.SYSGO,DISP=(OLD,DELETE)'
                                  202C+           PUNCH '//SYSOBJ DC    DSNAME=ICS.CLOD,DISP=(OLD,PASS)'
                                  2021+           PUNCH '//SYSLMOD DD  DSNAME=ICS.CLCD(DFSIQUEO),DISP=(OLD,PASS)X
                                     +                  '
                                  2022+           PUNCH '//SYSUT1 DC     UNIT=(SYSDA,SEP=(SYSLIN,SYSLMOD)),DISP=(X
                                     +                  ,DELETE),        X'
                                  2023+           PUNCH '//              SPACE=(1700,(1C0,5C))'
                                  2024+           PUNCH '//STEP29  EXEC  PGM=IEUASM,PARM=''LOAD,NODECK'',REGION=X
                                     +                  92K'
                                  2025+           PUNCH '//SYSLIB DD    CSNAME=IMS.GENLIB,DISP=(SHR,PASS)'
                                  2026+           PUNCH '//       DD     DSNAME=SYS1.MACLIB,DISP=(SHR,PASS)'
                                  2027+           PUNCH '//SYSGO  DC     UNIT=SYSDA,DISP=(,PASS),DCB=(LRECL=80,BLX
                                     +                  KSIZE=400,       X'
                                  2028+           PUNCH '//              RECFM=FB),SPACE=(CYL,(1,1),RLSE)'
                                  2029+           PUNCH '//SYSPRINT DD  SYSOUT=A,DCB=(LRECL=121,BLKSIZE=605,RECFX
                                     +                  M=FBA),          X'
```

```
LOC  OBJECT CODE    ADDR1 ADDR2 STMT   SOURCE STATEMENT                                      F30SEP69  2/12/70

                               2030+       PUNCH '//             SPACE=(605,(100,50),RLSE,,ROUND)'
                               2031+       PUNCH '//SYSUT1 DD    UNIT=SYSCA,DISP=(,DELETE),SPACE=(17C0,(1X
                                   +             00,50))'
                               2032+       PUNCH '//SYSUT2 DD    UNIT=SYSCA,DISP=(,DELETE),SPACE=(1700,(1X
                                   +             00,50))'
                               2033+       PUNCH '//SYSUT3 DD    UNIT=(SYSDA,SEP=(SYSLIB,SYSUT1,SYSUT2)),X
                                   +             DISP=(,DELETE),  X'
                               2034+       PUNCH '//             SPACE=(1700,(100,50),RLSE)'
                               2035+       PUNCH '//SYSIN  DD    *'
                               2036+       PUNCH '         COPY  PCHSSI'
                               2037+       PUNCH '         PRINT CN'
                               2038+       PUNCH '         DFSICIOB NUMIOB=10'
                               2039+       PUNCH '         TMSGSIZE QCRBUFN=14,MSGBUFN=10,DEVTYPE=(2314,2X
                                   +             314,2314,2314)'
                               2040+       PUNCH '        MSGDCB IQCRDNM=INQCR,                          X
                                   +                 CCNTINUE'
                               2041+       PUNCH '               COCRDNM=OUTQCR,                         X
                                   +                  CONTINUE'
                               2042+       PUNCH '               IMSGCNM=INMSG,                          X
                                   +                 CCNTINUE'
                               2043+       PUNCH '               OMSGDNM=OUTMSG'
                               2044+       PUNCH '         END'
                               2045+       PUNCH '/*'
                               2046+       PUNCH '//STEP30 EXEC PGM=IEWL,PARM=''REUS,NCAL,XREF,LIST'',REGX
                                   +             ION=110K'
                               2047+       PUNCH '//SYSPRINT DD  SYSOUT=A,DCB=(LRECL=121,BLKSIZE=605,RECFX
                                   +             M=FBA),         X'
                               2048+       PUNCH '//             SPACE=(605,(10,10),RLSE,,ROUND)'
                               2049+       PUNCH '//SYSLIN DD DSNAME=*.STEP29.SYSGO,DISP=(OLD,DELETE)'
                               2050+       PUNCH '//SYSCBJ CD   DSNAME=ICS.CLOD,DISP=(OLD,PASS)'
                               2051+       PUNCH '//SYSLMOD DD  DSNAME=ICS.CLCD(DFSICS40),DISP=(OLD,PASS)X
                                   +             '
                               2052+       PUNCH '//SYSUT1 CD    UNIT=(SYSDA,SEP=(SYSLIN,SYSLMOD)),DISP=(X
                                   +             ,DELETE),       X'
                               2053+       PUNCH '//             SPACE=(1700,(100,50))'
                               2054+       PUNCH '//STEP31  EXEC  PGM=IEUASM,PARM=''LOAD,NODECK'',REGION=X
                                   +             92K'
                               2055+       PUNCH '//SYSLIB DC   DSNAME=IMS.GENLIB,DISP=(SHR,PASS)'
                               2056+       PUNCH '//       DD    DSNAME=SYS1.MACLIB,DISP=(SHR,PASS)'
                               2057+       PUNCH '//SYSGO   DD   UNIT=SYSDA,DISP=(,PASS),DCB=(LRECL=80,BLX
                                   +             KSIZE=400,      X'
                               2058+       PUNCH '//             RECFM=FB),SPACE=(CYL,(1,1),RLSE)'
                               2059+       PUNCH '//SYSPRINT DD  SYSOUT=A,DCB=(LRECL=121,BLKSIZE=605,RECFX
                                   +             M=FBA),         X'
                               2060+       PUNCH '//             SPACE=(605,(100,50),RLSE,,ROUND)'
                               2061+       PUNCH '//SYSUT1 DD    UNIT=SYSDA,DISP=(,DELETE),SPACE=(1700,(1X
                                   +             00,50))'
                               2062+       PUNCH '//SYSUT2 DD    UNIT=SYSDA,DISP=(,CELETE),SPACE=(1700,(1X
                                   +             00,50))'
                               2063+       PUNCH '//SYSUT3 DD    UNIT=(SYSDA,SEP=(SYSLIB,SYSUT1,SYSUT2)),X
```

```
LOC  OBJECT CODE    ADDR1 ADDR2 STMT   SOURCE STATEMENT                                      F30SEP69  2/12/70

                                   +             DISP=(,DELETE),  X'
                               2064+       PUNCH '//             SPACE=(17C0,(100,50),RLSE)'
                               2065+       PUNCH '//SYSIN  DD    *'
                               2066+       PUNCH '         COPY  PCHSSI'
                               2067+       PUNCH '         PRINT ON'
                               2068+       PUNCH '       DFSGLBS PSB=22,DMB=9,SMB=41,                    X
                                   +                 CONTINUE'
                               2069+       PUNCH '               CLB=9,CTB=15,CNT=26,CDB=5,              X
                                   +               CONTINUE'
                               2070+       PUNCH '               PST=3,SAV=12,WAT=14,RQE=2,QUE=(1,150),  X
                                   +                CCNTINUE'
                               2071+       PUNCH '               SVC=(244,245),OSAM=(243,Z8),CVB=27,CTM=2X
                                   +             ,CTML=4'
                               2072+       PUNCH '       ISCD    SECTYPE=CSECT,CPOPT=500,                X
                                   +               CONTINUE'
                               2073+       PUNCH '               PUNIT=2314,PSER=STORGE,PLIB=ICS.PROCLIB X
                                   +             '
                               2074+       PUNCH '      TMSGSIZE QCRBUFN=14,MSGBUFN=10,DEVTYPE=(2314,2314X
                                   +             ,2314,2314)'
                               2075+       PUNCH '         DFSINT QCRS=14,MSGS=10'
                               2076+       PUNCH '         END'
                               2077+       PUNCH '/*'
                               2078+       PUNCH '//STEP32 EXEC PGM=IEWL,PARM=''REUS,NCAL,XREF,LIST'',REGX
                                   +             ION=110K'
                               2079+       PUNCH '//SYSPRINT DC  SYSOUT=A,DCB=(LRECL=121,BLKSIZE=605,RECFX
                                   +             M=FBA),         X'
                               2080+       PUNCH '//             SPACE=(605,(10,10),RLSE,,ROUND)'
                               2081+       PUNCH '//SYSLIN DD CSNAME=*.STEP31.SYSGO,DISP=(OLD,DELETE)'
                               2082+       PUNCH '//SYSOBJ DD   DSNAME=ICS.CLOD,DISP=(OLD,PASS)'
                               2083+       PUNCH '//SYSLMOD CD  DSNAME=ICS.CLOD(DFSISCDO),DISP=(OLD,PASS)X
                                   +             '
                               2084+       PUNCH '//SYSUT1 DD    UNIT=(SYSDA,SEP=(SYSLIN,SYSLMOD)),DISP=(X
                                   +             ,DELETE),       X'
                               2085+       PUNCH '//             SPACE=(1700,(100,50))'
                               2086+       PUNCH '//STEP33  EXEC  PGM=IEUASM,PARM=''LOAD,NODECK'',REGION=X
                                   +             92K'
                               2087+       PUNCH '//SYSLIB DD   DSNAME=IMS.GENLIB,DISP=(SHR,PASS)'
                               2088+       PUNCH '//       DD    DSNAME=SYS1.MACLIB,DISP=(SHR,PASS)'
                               2089+       PUNCH '//SYSGC   DD   UNIT=SYSCA,DISP=(,PASS),CCB=(LRECL=80,BLX
                                   +             KSIZE=400,      X'
                               2090+       PUNCH '//             RECFM=FB),SPACE=(CYL,(1,1),RLSE)'
                               2091+       PUNCH '//SYSPRINT DC  SYSOUT=A,DCB=(LRECL=121,BLKSIZE=605,RECFX
                                   +             M=FBA),         X'
                               2092+       PUNCH '//             SPACE=(605,(100,50),RLSE,,ROUND)'
                               2093+       PUNCH '//SYSUT1 DD    UNIT=SYSCA,DISP=(,DELETE),SPACE=(1700,(1X
                                   +             00,50))'
                               2094+       PUNCH '//SYSUT2 DD    UNIT=SYSCA,DISP=(,CELETE),SPACE=(17C0,(1X
                                   +             00,50))'
                               2095+       PUNCH '//SYSUT3 DD    UNIT=(SYSDA,SEP=(SYSLIB,SYSUT1,SYSUT2)),X
                                   +             DISP=(,CELETE),  X'
```

```
LOC  OBJECT CODE     ADDR1 ADDR2  STMT    SOURCE STATEMENT                                          F30SEP69    2/12/70

                                  2096+          PUNCH '//                 SPACE=(1700,(100,50),RLSE)'
                                  2097+          PUNCH '//SYSIN   DD       *'
                                  2098+          PUNCH '          COPY     PCHSSI'
                                  2099+          PUNCH '          PRINT    ON'
                                  2100+          PUNCH 'DFSISVA0 CSECT'
                                  2101+          PUNCH '          SVC      244          ASK SVC NUMBER'
                                  2102+          PUNCH '          BR       14'
                                  2103+          PUNCH '          ENTRY    DFSISVR0'
                                  2104+          PUNCH 'DFSISVR0 SVC      245          REPLY SVC NUMBER'
                                  2105+          PUNCH '          BR       14'
                                  2106+          PUNCH '          END'
                                  2107+          PUNCH '/*'
                                  2108+          PUNCH '//STEP34 EXEC  PGM=IEWL,PARM=''RENT,REFR,NCAL,XREF,LISTX
                                     +                '',REGION=110K'
                                  2109+          PUNCH '//SYSPRINT DD  SYSOUT=A,DCB=(LRECL=121,BLKSIZE=605,RECFX
                                     +                M=FBA),         X'
                                  2110+          PUNCH '//              SPACE=(605,(10,10),RLSE,,ROUND)'
                                  2111+          PUNCH '//SYSLIN DD     DSNAME=*.STEP33.SYSGO,DISP=(OLD,DELETE)'
                                  2112+          PUNCH '//       DD     DDNAME=SYSIN'
                                  2113+          PUNCH '//SYSOBJ DD    DSNAME=ICS.CLOD,DISP=(OLD,PASS)'
                                  2114+          PUNCH '//SYSLMOD DD    DSNAME=ICS.CLOD(DFSISVA0),DISP=(OLD,PASSX
                                     +                )'
                                  2115+          PUNCH '//SYSUT1 DD     UNIT=(SYSDA,SEP=(SYSLIN,SYSLMOD)),DISP=(X
                                     +                ,DELETE),        X'
                                  2116+          PUNCH '//              SPACE=(1700,(100,50),RLSE)'
                                  2117+          PUNCH '//STEP35 EXEC   PGM=IEWL,PARM=''RENT,REFR,NCAL,XREF,LISTX
                                     +                '',REGION=110K'
                                  2118+          PUNCH '//SYSPRINT DD   SYSOUT=A'
                                  2119+          PUNCH '//SYSLIN DD     DDNAME=SYSIN'
                                  2120+          PUNCH '//SYSLMOD DD    VOLUME=SER=IMSLIB,DISP=(OLD,PASS),        X
                                     +                             CONTINUE'
                                  2121+          PUNCH '//              DSNAME=ICS.CLOD,UNIT=2314'
                                  2122+          PUNCH '//SYSOBJ DD     VOLUME=SER=IMSLIB,DISP=(OLD,PASS),        X
                                     +                             CONTINUE'
                                  2123+          PUNCH '//              DSNAME=ICS.CLOD,UNIT=2314'
                                  2124+          PUNCH '//SYSUT1 DD     UNIT=(SYSDA,SEP=(SYSLIN,SYSLMOD)),DISP=(X
                                     +                ,DELETE),        X'
                                  2125+          PUNCH '//              SPACE=(1700,(100,50),RLSE)'
                                  2126+          PUNCH '//SYSIN  DD     *'
                                  2127+          PUNCH '          SETSSI 05012090'
                                  2128+          PUNCH '  CHANGE DFSIASK0(IGC244)'
                                  2129+          PUNCH '  CHANGE DFSIREP0(IGC245)'
                                  2130+          PUNCH '  INCLUDE SYSOBJ(DFSISVV0)    INTER-REGION SVC RTNES'
                                  2131+          PUNCH '  NAME DFSISVV0(R)'
                                  2132+          PUNCH '          SETSSI 05012090'
                                  2133+          PUNCH '  INCLUDE SYSOBJ(DFSISVA0)    SVC BUMPS'
                                  2134+          PUNCH '  INCLUDE SYSOBJ(DFSIRCC0)'
                                  2135+          PUNCH '  ENTRY DFSIRC00'
                                  2136+          PUNCH '  NAME DFSIRC00(R)            REGION CONTROLLER MODULE'
                                  2137+          PUNCH '          SETSSI 05012090'
```

```
LOC  OBJECT CODE     ADDR1 ADDR2  STMT    SOURCE STATEMENT                                          F30SEP69    2/12/70

                                  2138+          PUNCH '  INCLUDE SYSOBJ(DFSISVA0)    SVC BUMPS'
                                  2139+          PUNCH '  INCLUDE SYSOBJ(DFSIPR00)'
                                  2140+          PUNCH '  INCLUDE SYSOBJ(DFSIPCC0)'
                                  2141+          PUNCH '  ENTRY DFSIPC00'
                                  2142+          PUNCH '  NAME DFSIPC00(R)            PROG. CONTROLLER MODULE'
                                  2143+          PUNCH '          SETSSI 05012090'
                                  2144+          PUNCH '  INCLUDE SYSOBJ(DFSISVA0)    SVC BUMPS'
                                  2145+          PUNCH '  INCLUDE SYSOBJ(DFSIDLK0)'
                                  2146+          PUNCH '  ENTRY DFSIDLL0'
                                  2147+          PUNCH '  NAME DFSIDLL0(R)            DL/I BLOCK LOADER MODULE'
                                  2148+          PUNCH '          SETSSI 05012090'
                                  2149+          PUNCH '  INCLUDE SYSOBJ(DFSIBDD0)    DATABASE RECOVERY MODULE'
                                  2150+          PUNCH '  INCLUDE SYSOBJ(DFSILI00)    DL/I LANGUAGE INTERFACE'
                                  2151+          PUNCH '  ENTRY DLITCEL'
                                  2152+          PUNCH '  NAME DFSIBDR0(R)'
                                  2153+          PUNCH '/*'
                                  2154+          PUNCH '//STEP36 EXEC PGM=IEWL,REGION=110K,                       X
                                     +                           CONTINUE'
                                  2155+          PUNCH '//              PARM=''REUS,LET,NCAL,XREF,LIST'' '
                                  2156+          PUNCH '//SYSPRINT DD  SYSOUT=A,DCB=(LRECL=121,BLKSIZE=605,RECFX
                                     +                M=FBA),         X'
                                  2157+          PUNCH '//              SPACE=(605,(10,10),RLSE,,ROUND)'
                                  2158+          PUNCH '//SYSOBJ DD    DSNAME=ICS.CLOD,DISP=(OLD,PASS)'
                                  2159+          PUNCH '//       DD    DSNAME=ICS.CLOD,DISP=(OLD,PASS)'
                                  2160+          PUNCH '//SYSLMOD DD   DSNAME=ICS.CLOD(DFSIBLK0),DISP=(OLD,PASS)X
                                     +                '
                                  2161+          PUNCH '//SYSUT1 DD     UNIT=(SYSDA,SEP=(SYSOBJ,SYSLMOD)),DISP=(X
                                     +                ,DELETE),        X'
                                  2162+          PUNCH '//              SPACE=(1700,(100,50))'
                                  2163+          PUNCH '//SYSLIN DD * '
                                  2164+          PUNCH '          SETSSI 05012090'
                                  2165+          PUNCH '  INCLUDE SYSOBJ(DFSIDIR0)    PSB DMBL AND DMD BLOCKS'
                                  2166+          PUNCH '  INCLUDE SYSOBJ(DFSISMB0)    SCHEDULER MSGE BLOCKS'
                                  2167+          PUNCH '  INCLUDE SYSOBJ(DFSICLL0)    COMM LINE BLOCKS'
                                  2168+          PUNCH '  INCLUDE SYSOBJ(DFSICTB0)    COMM TERM BLOCKS'
                                  2169+          PUNCH '  INCLUDE SYSOBJ(DFSICNT0)    COMM NAME TABLE'
                                  2170+          PUNCH '  INCLUDE SYSOBJ(DFSICTT0)    COMM TRANS TABLE'
                                  2171+          PUNCH '  INCLUDE SYSOBJ(DFSICVB0)    COMM VERB TABLE'
                                  2172+          PUNCH '  INCLUDE SYSOBJ(DFSIOS40)    OSAM CONTROL BLOCKS'
                                  2173+          PUNCH '  INCLUDE SYSOBJ(DFSIPST0)    PARTITION SPEC TABLE'
                                  2174+          PUNCH '  INCLUDE SYSOBJ(DFSISAV0)    IMS/360 SAVE AREA SETS'
                                  2175+          PUNCH '  INCLUDE SYSOBJ(DFSIQUE0)    QUEUE CONTROL BLOCKS'
                                  2176+          PUNCH '  INCLUDE SYSOBJ(DFSISCD0)    SYSTEM CONTENTS DIRECTORY'
                                  2177+          PUNCH '  NAME DFSIBLK0(R)            IMS/360 CONTROL BLOCKS MODX
                                     +                ULE'
                                  2178+          PUNCH '/*'
                                  2179+          PUNCH '//STEP37 EXEC PGM=IEWL,REGION=110K,                       X
                                     +                           CONTINUE'
                                  2180+          PUNCH '//              PARM=''OVLY,LET,NCAL,XREF,LIST'' '
                                  2181+          PUNCH '//SYSPRINT DD  SYSOUT=A,DCB=(LRECL=121,BLKSIZE=605,RECFX
                                   +          M=FBA),          X'
                               2182+      PUNCH '//            SPACE=(605,(10,10),RLSE,,RCUND)'
                               2183+      PUNCH '//SYSORJ DD   DSNAME=ICS.CLCD,DISP=(OLD,PASS)'
                               2184+      PUNCH '//       DD   DSNAMC=ICS.CLCD,DISP=(OLD,PASS)'
                               2185+      PUNCH '//SYSLMCD CC  DSNAME=ICS.CLCD(DFSINUCO),DISP=(OLD,PASS)X
                                   +          '
                               2186+      PUNCH '//SYSUT1 CC   UNII=(SYSDA,SEP=(SYSOBJ,SYSLMOD)),DISP=(X
                                   +          ,DELETE),        X'
                               2187+      PUNCH '//            SPACE=(17C0,(1C0,50))'
                               2188+      PUNCH '//TELLIB CD   CSNAME=SYS1.TELCMLIB,DISP=SHR'
                               2189+      PUNCH '//SYSLIN DD *  '
                               2190+      PUNCH '          SETSSI 05012C90'
                               2191+      PUNCH '  INCLUDE SYSCBJ(DFSIXXXO)   RESIDENT MCDULE MAP'
                               2192+      PUNCH '  INCLUDF SYSCBJ(DFSIBLKO)   IMS CCNTROL BLOCKS'
                               2193+      PUNCH '  INCLUDE SYSOBJ(DFSIDSPO)   IMS SUBTASK DISPATCHER'
                               2194+      PUNCH '  INCLUDE SYSCBJ(CFSIRSTO)   IMS RESTART'
                               2195+      PUNCH '  INCLUDE SYSCBJ(DFSIRSIO)   RESTART INITIALIZATION'
                               2196+      PUNCH '  INCLUCE SYSCBJ(DFSICPCO)   IMS CHECKPOINT'
                               2197+      PUNCH '  INCLUDE SYSCBJ(DFSIASIO)   SCHECULER - INITIATION'
                               2198+      PUNCH '  INCLUDE SYSOBJ(DFSIASTO)   SCHEDULER - TERMINATICN'
                               2199+      PUNCH '  INCLUDE SYSCBJ(DFSIMBEO)   SCHEDULER - SMB ENQUEUE'
                               2200+      PUNCH '  INCLUDE SYSCBJ(CFSIMBDO)   SCHEDULER - SMB DEQUEUE'
                               2201+      PUNCH '  INCLUDE SYSCBJ(DFSICLIO)   COMM INPUT PROCESSOR'
                               2202+      PUNCH '  INCLUCE SYSCBJ(DFSICLOO)   CCMM CUTPUT PROCESSOR'
                               2203+      PUNCH '  INCLUDE SYSCBJ(DFSICLPO)   COMMANC MSGE PROCESSOR'
                               2204+      PUNCH '  INCLUDE SYSCBJ(DFSICLRO)   MESSAGE ROUTER'
                               2205+      PUNCH '  INCLUDE SYSCBJ(DFSICLMO)   MESSAGE GENERATOR'
                               2206+      PUNCH '  INCLUDE SYSCBJ(DFSICLTO)   COMM TRANSLATION MODULE'
                               2207+      PUNCH '  INCLUDE SYSCBJ(DFSICLBO)   COMM BACKSPACE EDIT'
                               2208+      PUNCH '  INCLUDE SYSCBJ(DFSICLFO)   SYMBCLIC DEST FINDER'
                               2209+      PUNCH '  INCLUDE SYSOBJ(DFSICLSO)   SECURITY PROCESSCR'
                               2210+      PUNCH '  INCLUDE SYSCBJ(CFSICLXO)   CCMM RESET POLL'
                               2211+      PUNCH '  INCLUDE SYSCBJ(DFSICL1O)   /BROACCAST CCMMAND'
                               2212+      PUNCH '  INCLUDE SYSOBJ(DFSICL2O)   /CHE /RES COMMAND'
                               2213+      PUNCH '  INCLUDE SYSCBJ(DFSICLAO)   /IAM CCMMAND'
                               2214+      PUNCH '  INCLUDE SYSOBJ(DFSICL3O)   EDIT COMMAND MSGE'
                               2215+      PUNCH '  INCLUDE SYSCBJ(CFSICL4O)   /STA /STO /PST COMMAND'
                               2216+      PUNCH '  INCLUDE SYSCBJ(DFSICL5O)   /TEST /END /EXC COMMANDS'
                               2217+      PUNCH '  INCLUDE SYSOBJ(DFSICL6O)   /CHA COMMAND'
                               2218+      PUNCH '  INCLUDE SYSCBJ(DFSICL7O)   /ASSIGN COMMAND'
                               2219+      PUNCH '  INCLUDE SYSCBJ(DFSICL8O)   /DEL COMMAND'
                               2220+      PUNCH '  INCLUDE SYSOBJ(DFSICL9O)   /LOCK / UNLOCK COMMAND'
                               2221+      PUNCH '  INCLUDE SYSCBJ(CFSICLDO)   /DISPLAY CONTROL MODULE'
                               2222+      PUNCH '  INCLUDE SYSCBJ(DFSIDP1O)        "     STATUS'
                               2223+      PUNCH '  INCLUDE SYSCBJ(DFSIDP2O)        "     ACTIVE'
                               2224+      PUNCH '  INCLUDE SYSCBJ(CFSICP3O)        "     QUEUES'
                               2225+      PUNCH '  INCLUDE SYSCBJ(DFSIDP4O)        "     TRAN && LTERM'
                               2226+      PUNCH '  INCLUDE SYSCBJ(CFSICP5O)        "     PGM && DATABASE'
                               2227+      PUNCH '  INCLUDE SYSCBJ(CFSIDP6O)        "     LINE && PTERM'
                               2228+      PUNCH '  INCLUDE SYSOBJ(DFSIDP7O)        "     ASSIGNMENT'
```

```
                               2229+      PUNCH '  INCLUDE SYSCBJ(DFSIRD1O)        "     MASTER'
                               2230+      PUNCH '  INCLUDE SYSOBJ(DFSIDLAO)   DL/I CALL ANALYZER'
                               2231+      PUNCH '  INCLUDE SYSOBJ(DFSIDLMO)   DL/I BLOCK MOVER'
                               2232+      PUNCH '  INCLUDE SYSCBJ(CFSICLGO)   DL/I OPEN MODULE'
                               2233+      PUNCH '  INCLUDE SYSCBJ(DFSIOS1O)   CSAM CPEN'
                               2234+      PUNCH '  INCLUDE SYSCBJ(CFSIIDEO)   BLOCK DEQUEUE MODULE'
                               2235+      PUNCH '  INCLUDE SYSCBJ(DFSIIENO)   BLOCK ENQUEUE MODULE'
                               2236+      PUNCH '  INCLUDE SYSCBJ(DFSIDBLO)   DB SEG'LCG FCR BACKCUT'
                               2237+      PUNCH '  INCLUDE SYSCBJ(DFSIPREO)   MSGE AND LOG PREFIX BLDR'
                               2238+      PUNCH '  INCLUDE SYSCBJ(DFSILCOO)   WRITE LCG ROUTINE'
                               2239+      PUNCH '  INCLUDE SYSCBJ(DFSISTPO)   START REGION'
                               2240+      PUNCH '  INCLUDE SYSCBJ(CFSIPTPO)   STCP REGION'
                               2241+      PUNCH '  INCLUDE SYSCBJ(DFSIASEO)   SIM REGION TERMINATION'
                               2242+      PUNCH '  INCLUDE SYSCBJ(DFSIRWQO)   READ/WRITE MSGE QUEUE'
                               2243+      PUNCH '  INCLUDE SYSCBJ(DFSIQMSO)   REUSE CUEUE MODULE'
                               2244+      PUNCH '  INCLUDE SYSCBJ(OFSISMNO)   STORAGE POCL MGMT'
                               2245+      PUNCH '  INCLUDE TELLIB(IECTLOPN)   BTAM SAD/ENABLE'
                               2246+      PUNCH '  INCLUDE TELLIB(IECTCHGN)  '
                               2247+      PUNCH '  INCLUDE SYSOBJ(DFSICM1O)   COMM MESSAGF TABLE'
                               2248+      PUNCH '  INCLUDE SYSCBJ(DFSICLEO)   /SET /RESET COMMANDS'
                               2249+      PUNCH '  INCLUDE SYSOBJ(DFSISMIO)   SECURITY MAINT INIT'
                               2250+      PUNCH '  INCLUDE SYSCBJ(DFSIINTO)   INIT - CONTROL && MISC'
                               2251+      PUNCH '  INCLUDE SYSCBJ(DFSIINLO)   INIT - MODULE LOADER'
                               2252+      PUNCH '  INCLUDE SYSOBJ(DFSIIN1O)   INIT - JOBLIB MODULE TABLEX
                                   +          '
                               2253+      PUNCH '  INCLUDE SYSCBJ(DFSIIN2O)   INIT - SVCLIB MODULE TABLEX
                                   +          '
                               2254+      PUNCH '  INCLUDE SYSCBJ(DFSIINDO)   INIT - DMB DIRECTORY'
                               2255+      PUNCH '  INCLUDE SYSCBJ(DFSIINSO)   INIT - STORAGE POOL MGMT'
                               2256+      PUNCH '  INCLUDE SYSOBJ(DFSIINQO)   INIT - QUEUE MANAGEMENT'
                               2257+      PUNCH '  INCLUDE SYSCBJ(DFSIINBO)   INIT - COMMUNICATIONS'
                               2258+      PUNCH '  INCLUDE SYSOBJ(DFSIINXO)   INIT - RESIDENT XFR CTL'
                               2259+      PUNCH '  INCLUDE SYSCBJ(DFSIOS6O)   OSAM CLOSE RCUTINE'
                               2260+      PUNCH '  CHANGE DFSICS6O(DFSIOS7O)  CHG EP TO OSAM CLOSE RTNE'
                               2261+      PUNCH '  INCLUDE SYSCBJ(DFSIOS6O)   OSAM CLCSE RTNE(2ND COPY)'
                               2262+      PUNCH '  CHANGE CFSIOS6O(DFSIOS7O)  CHG DLCO REFERENCE'
                               2263+      PUNCH '  INCLUDE SYSCBJ(DFSIDLCO)   DL/I CLOSE MODULE'
                               2264+      PUNCH '         ENTRY DFSSTART'
                               2265+      PUNCH '                OVERLAY IMSA'
                               2266+      PUNCH '   INSERT DFSICL1O'
                               2267+      PUNCH '                OVERLAY IMSA'
                               2268+      PUNCH '   INSERT DFSICL2O'
                               2269+      PUNCH '                OVERLAY IMSA'
                               2270+      PUNCH '   INSERT DFSICL3O'
                               2271+      PUNCH '                OVERLAY IMSA'
                               2272+      PUNCH '   INSERT DFSICL4O'
                               2273+      PUNCH '                OVERLAY IMSA'
                               2274+      PUNCH '   INSERT DFSICL5O'
                               2275+      PUNCH '                OVERLAY IMSA'
                               2276+      PUNCH '   INSERT DFSICL6O'
```

```
LOC  OBJECT CODE    ADDR1 ADDR2  STMT   SOURCE STATEMENT                                   F30SEP69   2/12/70

                                 2277+         PUNCH '              OVERLAY IMSA'
                                 2278+         PUNCH '   INSERT DFSICL7C'
                                 2279+         PUNCH '              OVERLAY IMSA'
                                 2280+         PUNCH '   INSERT DFSICL80'
                                 2281+         PUNCH '              OVERLAY IMSA'
                                 2282+         PUNCH '   INSERT DFSICL90'
                                 2283+         PUNCH '              OVERLAY IMSA'
                                 2284+         PUNCH '   INSERT DFSIDP10'
                                 2285+         PUNCH '              OVERLAY IMSA'
                                 2286+         PUNCH '   INSERT DFSIDP20'
                                 2287+         PUNCH '              OVERLAY IMSA'
                                 2288+         PUNCH '   INSERT DFSIDP30'
                                 2289+         PUNCH '              OVERLAY IMSA'
                                 2290+         PUNCH '   INSERT CFSICP40'
                                 2291+         PUNCH '              OVERLAY IMSA'
                                 2292+         PUNCH '   INSERT DFSIDP50'
                                 2293+         PUNCH '              OVERLAY IMSA'
                                 2294+         PUNCH '   INSERT DFSIDP60'
                                 2295+         PUNCH '              OVERLAY IMSA'
                                 2296+         PUNCH '   INSERT CFSICP70'
                                 2297+         PUNCH '              OVERLAY IMSA'
                                 2298+         PUNCH '   INSERT DFSIRD10'
                                 2299+         PUNCH '              OVERLAY IMSA'
                                 2300+         PUNCH '   INSERT DFSISMI0'
                                 2301+         PUNCH '              OVERLAY IMSA'
                                 2302+         PUNCH '   INSERT DFSIINT0'
                                 2303+         PUNCH '   INSERT DFSIINX0'
                                 2304+         PUNCH '              OVERLAY IMSB'
                                 2305+         PUNCH '   INSERT DFSIINL0'
                                 2306+         PUNCH '   INSERT DFSIIN10'
                                 2307+         PUNCH '   INSERT DFSIIN20'
                                 2308+         PUNCH '              OVERLAY IMSB'
                                 2309+         PUNCH '   INSERT DFSIIND0'
                                 2310+         PUNCH '              OVERLAY IMSB'
                                 2311+         PUNCH '   INSERT DFSIINS0'
                                 2312+         PUNCH '   INSERT DFSINTB0'
                                 2313+         PUNCH '              OVERLAY IMSB'
                                 2314+         PUNCH '   INSERT DFSIINQ0'
                                 2315+         PUNCH '              OVERLAY IMSB'
                                 2316+         PUNCH '   INSERT DFSIINB0'
                                 2317+         PUNCH '              OVERLAY IMSA'
                                 2318+         PUNCH '   INSERT DFSISTP0'
                                 2319+         PUNCH '              OVERLAY IMSC'
                                 2320+         PUNCH '   INSERT DFSIPTP0'
                                 2321+         PUNCH '   INSERT CFSIASE0 '
                                 2322+         PUNCH '              OVERLAY IMSA'
                                 2323+         PUNCH '   INSERT DFSIDL00'
                                 2324+         PUNCH '              CVERLAY IMSD'
                                 2325+         PUNCH '   INSERT DFSIOS60'
                                 2326+         PUNCH '              OVERLAY IMSE'
```

```
LOC  OBJECT CODE    ADDR1 ADDR2  STMT   SOURCE STATEMENT                                   F30SEP69   2/12/70

                                 2327+         PUNCH '   INSERT DFSIOS10'
                                 2328+         PUNCH '              CVERLAY IMSA'
                                 2329+         PUNCH '   INSERT CFSIDLC0'
                                 2330+         PUNCH '              CVERLAY IMSF'
                                 2331+         PUNCH '   INSERT DFSIOS7C'
                                 2332+         PUNCH '              OVERLAY IMSA'
                                 2333+         PUNCH '   INSERT DFSICLE0'
                                 2334+         PUNCH '              OVERLAY IMSA'
                                 2335+         PUNCH '   INSERT DFSICM10'
                                 2336+         PUNCH '               OVERLAY IMSA'
                                 2337+         PUNCH '   INSERT DFSIRSI0'
                                 2338+         PUNCH '  NAME DFSINUCO(R)            IMS/360 ONLINE NUCLEUS'
                                 2339+         PUNCH '/*'
                                 2340+         PUNCH '//STEP38 EXEC PGM=IEHLIST,REGION=100K'
                                 2341+         PUNCH '//SYSPRINT DD  SYSOUT=A'
                                 2342+         PUNCH '//RESLIB DD     VOLUME=SER=IMSLIB,DISP=(OLD,PASS),       X
                                     +                        CONTINUE'
                                 2343+         PUNCH '//              DSNAME=ICS.CLOD,UNIT=2314'
                                 2344+         PUNCH '//SYSIN  DD     *'
                                 2345+         PUNCH '  LISTVTOC DSNAME=ICS.CLOD,VOL=2314=IMSLIB'
                                 2346+         PUNCH '  LISTVTOC DUMP,DSNAME=ICS.CLCD,VOL=2314=IMSLIB'
                                 2347+         PUNCH '  LISTPDS DSNAME=ICS.CLOD,VOL=2314=IMSLIB'
                                 2348+         PUNCH '/*'


                                 2350                *,*** SUCCESSFUL IMS/360 SYSTEM DEFINITION
                                 2351                *,    GENERATION WILL BE FOR ALL IMS/360 FUNCTIONS.
```

```
LCC  OBJECT CODE     ADDR1 ADDR2  STMT    SOURCE STATEMENT                                          F30SEP69   2/12/70

                                  2353+** WARNING **
                                  2354                 *,****************************************************
                                  2355                 *,****************************************************
                                  2356                 *,                                                   *
                                  2357                 *,   IGG019Z8 MUST BE MOVED TO SYS1.SVCLIB AND IGC243
                                  2358                 *,MUST BE LINK EDITED WITH THE OS/360 NUCLEUS FOR
                                  2359                 *,SUCCESSFUL IMS/360 SYSTEM EXECUTION.
                                  2360                 *,
                                  2361                 *,   THE CN LINE(TP) FUNCTICNS OF THE IMS/360 SYSTEM
                                  2362                 *,REQUIRE IGC244 AND IGC245 BE LINK EDITED WITH THE
                                  2363                 *,OS/360 NUCLELS FCR SUCCESSFUL EXECUTION OF THESE
                                  2364                 *,FEATURES.  THE LOAD MEMBER NAME IS DFSISVVO AND IT
                                  2365                 *,WILL BE PLACED IN ICS.CLOD BY STAGE II OF
                                  2366                 *,IMS/360 SYSTEM GENERATION.
                                  2367                 *,
                                  2368                 *,   DATABASE BACKOUT AND DUMP FUNCTIONS OF IMS/360
                                  2369                 *,REQUIRE DFSIPDPO BE MOVED TO ICS.PSBLIB AND
                                  2370                 *,RENAMED DFSIBDRO FCR SUCCESSFUL EXECUTION OF THESE
                                  2371                 *,FEATURES.
                                  2372                 *,
                                  2373                 *,   STEP 37 OF STAGE II OF IMS/360 SYSTEM GENERATICN
                                  2374                 *,REQUIRES SYS1.TELCMLIB BE A CATALOGED DATA SET ON
                                  2375                 *,THE GENERATING SYSTEM AND CONTAIN THE INDICATED LOAD
                                  2376                 *,MODULES TO BE INCLUDED IN THE IMS/360 NUCLEUS.
                                  2377                 *,
                                  2378                 *,   PRCCEDURE 'IMS' MUST BE MOVED TO SYS1.PROCLIB
                                  2379                 *,FOR SUCCESSFUL EXECUTICN OF THIS PROCEDURE. STAGE II
                                  2380                 *,CF IMS/360 SYSTEM GENERATICN PLACES ALL PROCEDURES
                                  2381                 *,IN ICS.PROCLIB.
                                  2382                 *,
                                  2383                 *,   PROCEDURES 'IMSO' AND 'IMS1' MUST BE UPDATED TO
                                  2384                 *,INCLUDE DD CARDS FOR THE DATABASES SPECIFIED DURING
                                  2385                 *,IMS/360 SYSTEM DEFINITION BEFORE THESE PROCEDURES
                                  2386                 *,CAN BE SUCCESSFULLY EXECUTED.
                                  2387                 *,
                                  2388                 *,   DFSILNKO AND DFSIRCCO SHOULD BE IN SYS1.LINKLIB
                                  2389                 *,FOR EFFICIENT IMS/360 SYSTEM CPERATION.
                                  2390                 *,
                                  2391                 *,   SEE IMS/360 AND OS/360 SYSTEM OPERATION MANUALS
                                  2392                 *,FOR MODULES TO BE PLACED IN LINK PACK AREA FOR
                                  2393                 *,EFFICIENT SYSTEM OPERATION.
                                  2394                 *,
                                  2395                 *,   APPROXIMATE SIZE OF DFSIBLKO WILL BE 19500 BYTES.
                                  2396                 *,IF CALCULATEC AND DEFAULT BUFFER AND POOL SIZES ARE
                                  2397                 *,USED;  THE TCTAL SIZE WILL BE 52000 BYTES.
                                  2398                 *,
                                  2399                 *,   STAGE II OF IMS/360 SYSTEM GENERATICN WILL PLACE
                                  2400                 *,ALL SYSTEM LCAC MODULES IN ICS.CLOD.
                                  2401                 *,                                                   *
                                  2402                 *,****************************************************
```

```
LCC  CBJECT CCDE     ADDR1 ADDR2  STMT    SOURCE STATEMENT                                          F30SEP69   2/12/70

                                  2403                 *,****************************************************
                                  2404           END
```

Batch Stand-Alone Example

This example illustrates the output from Stage 1 of IMS/360 system definition. The input to Stage 1 (that is, the control cards) is provided in the output listing, as is a summary of the Data Set Specifications, followed by the punch statements and warning comments at the end.

```
                                                                                                    PAGE    1

LOC  OBJECT CODE    ADDR1 ADDR2  STMT   SOURCE STATEMENT                                    F01JAN68  10/16/68
                                    1        IMSCTRL SYSTEM=(MVT,BATCH),OSAMSVC=245,OCENDA=WA

                                    3                *,     BATCH IMS/360 FUNCTIONS ARE SELECTED
                                    4                *,     MVT PROGRAMMING SYSTEM WILL BE USED
                                    5                *,     OSAM CHANNEL END APPENDAGE - IGG019WA
                                    6                *,     SUPERVISOR STATE SVC NUMBER - 245

                                    8        RESLIB UNIT=2311,VOLNO=ILIB01
                                    9        MACLIB UNIT=2311,VOLNO=ILIB02
                                   10        PGMLIB
                                   11        PSBLIB
                                   12        DBDLIB
                                   13        PROCLIB
                                   14        IMSGEN UT1SOS=GENSET,LEPRT=(LIST,XREF),ASMPRT=ON
```

```
     IMS/360 SYSTEM DEFINITION SPECIFICATIONS                                                PAGE    2

LOC  OBJECT CODE    ADDR1 ADDR2  STMT   SOURCE STATEMENT                                    F01JAN68  10/16/68
                                   16                *,IMS/360 DATA SET SPECIFICATIONS (BATCH)

                                   18                *,  RESLIB SPECIFICATION:
                                   19                *,     DSNAME-IMS.RESLIB VOLUME-ILIB01 UNIT-2311

                                   21                *,  MACLIB SPECIFICATION:
                                   22                *,     DSNAME-IMS.MACLIB VOLUME-ILIB02
                                   23                *,       UNIT-2311         COPY-UTILITY

                                   25                *,  PROCLIB SPECIFICATION:
                                   26                *,     DSNAME-IMS.PROCLIB VOLUME-N/A UNIT-N/A

                                   28                *,  PGMLIB SPECIFICATION:
                                   29                *,     DSNAME-IMS.PGMLIB VOLUME-N/A UNIT-N/A

                                   31                *,  PSBLIB SPECIFICATION:
                                   32                *,     DSNAME-IMS.PSBLIB VOLUME-N/A UNIT-N/A

                                   34                *,  DBDLIB SPECIFICATION:
                                   35                *,     DSNAME-IMS.DBDLIB VOLUME-N/A UNIT-N/A
```

```
    IMS/360 SYSTEM DEFINITION SPECIFICATIONS                                                        PAGE    3

LCC  OBJECT CODE     ADDR1 ADDR2  STMT    SOURCE STATEMENT                                      F01JAN68  10/16/68

                                   38+         PUNCH '//IMSGEN JOB 1,''IMSGEN STAGE II'',MSGCLASS=A,MSGLEVEL=X
                                    +                1'
                                   39+         PUNCH '//STEP1 EXEC PGM=IEHMOVE,REGION=100K'
                                   40+         PUNCH '//SYSPRINT DD SYSOUT=A'
                                   41+         PUNCH '//SYSUT1 DD DSNAME=GENSET,DISP=(OLD,PASS)'
                                   42+         PUNCH '//DD2 DD DSNAME=IMS.LOAD,DISP=(OLD,PASS)'
                                   43+         PUNCH '//DD3     DD     VOLUME=SER=ILIB01,DISP=(OLD,PASS),       X
                                    +                              CONTINUE'
                                   44+         PUNCH '//                DSNAME=IMS.RESLIB,UNIT=2311'
                                   45+         PUNCH '//SYSIN DD *'
                                   46+         PUNCH '  COPY PDS=IMS.LOAD,TO=2311=ILIB01,RENAME=IMS.RESLIB'
                                   47+         PUNCH '  SELECT MEMBER=DFSIRA00   REGION ANALYZER MODULE'
                                   48+         PUNCH '  SELECT MEMBER=DFSIRCC0   REGION CONTROLLER MODULE'
                                   49+         PUNCH '  SELECT MEMBER=DFSIPCC0   PROG. CONTROLLER MODULE'
                                   50+         PUNCH '  SELECT MEMBER=DFSIPR00   PROG. REQUEST HANDLER'
                                   51+         PUNCH '  SELECT MEMBER=DFSILNK0   IMS/360 LINKAGE EDITOR'
                                   52+         PUNCH '  SELECT MEMBER=DFSILI00   DL/I LANGUAGE INTERFACE'
                                   53+         PUNCH '  SELECT MEMBER=DFSIDLR0   DL/I RETRIEVE MODULE'
                                   54+         PUNCH '  SELECT MEMBER=DFSIDLI0   DL/I INSERT MODULE'
                                   55+         PUNCH '  SELECT MEMBER=DFSIDLD0   DL/I DELETE/REPLACE MODULE'
                                   56+         PUNCH '  SELECT MEMBER=DFSIDLE0   DL/I DATA BASE LOAD MODULE'
                                   57+         PUNCH '  SELECT MEMBER=DFSIDLN0   DL/I BATCH INITIALIZATION'
                                   58+         PUNCH '  SELECT MEMBER=DFSIDLH0   DL/I HSAM MODULE'
                                   59+         PUNCH '  SELECT MEMBER=DFSIDLT0   DL/I PROGRAM TEST MODULE'
                                   60+         PUNCH '  SELECT MEMBER=DFSISNAP   DL/I BLOCK SNAP ROUTINE'
                                   61+         PUNCH '  SELECT MEMBER=DFSIISM0   DL/I ISAM SIMULATOR'
                                   62+         PUNCH '  SELECT MEMBER=DFSIWKN0   DL/I WRITE KEY NEW MODULE'
                                   63+         PUNCH '  SELECT MEMBER=DFSIDLK0   DL/I BLOCK LOADER MODULE'
                                   64+         PUNCH '  SELECT MEMBER=DFSIOS20   OSAM READ/WRITE MODULE'
                                   65+         PUNCH '  SELECT MEMBER=DFSIOS30   OSAM CHECK ROUTINE'
                                   66+         PUNCH '  SELECT MEMBER=DFSIOS60   OSAM OPEN/CLOSE(OVFW)'
                                   67+         PUNCH '  SELECT MEMBER=DFSIOS10   OSAM OPEN ROUTINE'
                                   68+         PUNCH '  SELECT MEMBER=DFSISMM0   STORAGE MANAGEMENT MODULE'
                                   69+         PUNCH '  SELECT MEMBER=DFSIOL00   DL/I OPEN MODULE'
                                   70+         PUNCH '  SELECT MEMBER=DFSIDLC0   DL/I CLOSE MODULE'
                                   71+         PUNCH '  SELECT MEMBER=DFSIDBA0   DL/I BATCH ANALYZER'
                                   72+         PUNCH '  SELECT MEMBER=DFSIBKB0   DL/I BATCH BLOCK MODULE'
                                   73+         PUNCH '  SELECT MEMBER=((DFSIOCE0,IGG019WA))   OSAM CH. END APX
                                    +                PENDAGE'
                                   74+         PUNCH '/*'
                                   75+         PUNCH '//STEP2 EXEC PGM=IEWL,PARM=''RENT,NCAL,LIST,XREF'' '
                                   76+         PUNCH '//SYSPRINT DD SYSOUT=A'
                                   77+         PUNCH '//SYSLIN DD DDNAME=SYSIN'
                                   78+         PUNCH '//SYSLMOD DD   VOLUME=SER=ILIB01,DISP=(OLD,PASS),       X
                                    +                              CONTINUE'
                                   79+         PUNCH '//               DSNAME=IMS.RESLIB,UNIT=2311'
                                   80+         PUNCH '//SYSOBJ DD DSNAME=IMS.LOAD,DISP=(OLD,PASS)'
                                   81+         PUNCH '//SYSUT1 DD      UNIT=(SYSDA,SEP=(SYSLIN,SYSLMOD)),DISP=(X
                                    +                ,DELETE),         X'
                                   82+         PUNCH '//               SPACE=(1700,(100,50),RLSE)'
                                   83+         PUNCH '//SYSIN DD *'
                                   84+         PUNCH '  CHANGE IGC255(IGC245)'
                                   85+         PUNCH '  INCLUDE SYSOBJ(DFSIOSV0)    OSAM SVC ROUTINE'
                                   86+         PUNCH '  NAME IGC245(R)'
                                   87+         PUNCH '  INCLUDE SYSOBJ(DFSISVN0)'
```

```
    IMS/360 SYSTEM DEFINITION SPECIFICATIONS                                                        PAGE    4

LCC  OBJECT CODE     ADDR1 ADDR2  STMT    SOURCE STATEMENT                                      F01JAN68  10/16/68

                                   88+         PUNCH '  INCLUDE SYSOBJ(DFSIRCC0)'
                                   89+         PUNCH '  ENTRY DFSIRC00'
                                   90+         PUNCH '  NAME DFSIRC00(R)            REGION CONTROLLER MODULE'
                                   91+         PUNCH '  INCLUDE SYSOBJ(DFSISVN0)'
                                   92+         PUNCH '  INCLUDE SYSOBJ(DFSIPR00)'
                                   93+         PUNCH '  INCLUDE SYSOBJ(DFSIPCC0)'
                                   94+         PUNCH '  ENTRY DFSIPC00     '
                                   95+         PUNCH '  NAME DFSIPC00(R)            PROG. CONTROLLER MODULE'
                                   96+         PUNCH '  INCLUDE SYSOBJ(DFSISVN0)'
                                   97+         PUNCH '  INCLUDE SYSOBJ(DFSIDLK0)'
                                   98+         PUNCH '  ENTRY DFSIDLL0'
                                   99+         PUNCH '  NAME DFSIDLL0(R)            DL/I BLOCK LOADER MODULE'
                                  100+         PUNCH '/*'
                                  101+         PUNCH '//STEP3 EXEC PGM=IEHMOVE,REGION=100K'
                                  102+         PUNCH '//SYSPRINT DD SYSOUT=A'
                                  103+         PUNCH '//SYSUT1 DD DSNAME=GENSET,DISP=(OLD,PASS)'
                                  104+         PUNCH '//DD2 DD DSNAME=IMS.GENLIB,DISP=(OLD,PASS)'
                                  105+         PUNCH '//DD3 DD   VOLUME=SER=ILIB02,DISP=OLD,                    X
                                    +                             CONTINUE'
                                  106+         PUNCH '//               DSNAME=IMS.MACLIB,UNIT=2311'
                                  107+         PUNCH '//SYSIN DD *'
                                  108+         PUNCH '  COPY PDS=IMS.GENLIB,TO=2311=ILIB02,RENAME=IMS.MACLIB'
                                  109+         PUNCH '      SELECT MEMBER=DBD'
                                  110+         PUNCH '      SELECT MEMBER=DBDFP'
                                  111+         PUNCH '      SELECT MEMBER=DBDFP1'
                                  112+         PUNCH '      SELECT MEMBER=DBDGEN'
                                  113+         PUNCH '      SELECT MEMBER=DMAN'
                                  114+         PUNCH '      SELECT MEMBER=SEGM'
                                  115+         PUNCH '      SELECT MEMBER=GLOBALS'
                                  116+         PUNCH '      SELECT MEMBER=IDCBOS'
                                  117+         PUNCH '      SELECT MEMBER=CONVERT'
                                  118+         PUNCH '      SELECT MEMBER=FINISH'
                                  119+         PUNCH '      SELECT MEMBER=FLD'
                                  120+         PUNCH '      SELECT MEMBER=FLDK'
                                  121+         PUNCH '      SELECT MEMBER=PCB'
                                  122+         PUNCH '      SELECT MEMBER=PSBGEN'
                                  123+         PUNCH '      SELECT MEMBER=SENSEG'
                                  124+         PUNCH '/*'
                                  125+         PUNCH '//STEP4 EXEC PGM=IEBUPDTE,PARM=NEW,REGION=90K'
                                  126+         PUNCH '//SYSPRINT DD SYSOUT=A'
                                  127+         PUNCH '//SYSUT2 DD DSNAME=IMS.PROCLIB,DISP=OLD'
                                  128+         PUNCH '//SYSIN DD DATA'
                                  129+         PUNCH './        ADD   NAME=DLITCBL'
                                  130+         PUNCH '  INCLUDE SYSOBJ(DFSILI00)'
                                  131+         PUNCH '  ENTRY DLITCBL'
                                  132+         PUNCH './        ADD   NAME=DLITPLI'
                                  133+         PUNCH '  INCLUDE SYSOBJ(DFSILI00)'
                                  134+         PUNCH '  ENTRY IHESAPB'
                                  135+         PUNCH './        ADD   NAME=DLI'
                                  136+         PUNCH './        NUMBER NEW1=00000010,INCR=00000010'
                                  137+         PUNCH '//        PROC  PSB=TEMPNAME'
                                  138+         PUNCH '//G     EXEC PGM=DFSIRC00,PARM=''3,&&PSB'',REGION=120K'
                                  139+         PUNCH '//IMS DD   DSNAME=IMS.PSBLIB,DISP=SHR'
                                  140+         PUNCH '//    DD   DSNAME=IMS.DBDLIB,DISP=SHR'
                                  141+         PUNCH '//SYSUDUMP DD SYSOUT=A,SPACE=(605,(500,500),RLSE,,ROUNDX
IMS/360 SYSTEM DEFINITION SPECIFICATIONS                                                   PAGE    5

LOC  OBJECT CODE    ADDR1 ADDR2  STMT   SOURCE STATEMENT                                 F01JAN68  10/16/68

                                    +          ),             X'
                                 142+    PUNCH '//            DCB=(RECFM=FBA,LRECL=121,BLKSIZE=605)'
                                 143+    PUNCH './      ADD  NAME=IMSCOBOL'
                                 144+    PUNCH './      NUMBER NEW1=00000010,INCR=00000010'
                                 145+    PUNCH '//      PROC MBR=,PAGES=60'
                                 146+    PUNCH '//C     EXEC PGM=IEQCBL00,PARM=''SIZE=110000,LINECNT=5X
                                    +          0'',REGION=126K'
                                 147+    PUNCH '//SYSLIN DD   DSNAME=&&&&LIN,DISP=(MOD,PASS),UNIT=SYSDAX
                                    +          ,DCB=(LRECL=80,   X'
                                 148+    PUNCH '//            RECFM=FB,BLKSIZE=400),SPACE=(CYL,(4,1),RX
                                    +          LSE)'
                                 149+    PUNCH '//SYSPRINT DD SYSOUT=A,DCB=(RECFM=FBA,LRECL=121,BLKSIZEX
                                    +          =605),         X'
                                 150+    PUNCH '//            SPACE=(605,(&&PAGES.0,&&PAGES),RLSE,,ROUX
                                    +          ND)'
                                 151+    PUNCH '//SYSUT1 DD   UNIT=SYSDA,DISP=(NEW,DELETE),SPACE=(CYL,(X
                                    +          10,1),RLSE)'
                                 152+    PUNCH '//SYSUT2 DD   UNIT=SYSDA,DISP=(NEW,DELETE),SPACE=(CYL,(X
                                    +          10,1),RLSE)'
                                 153+    PUNCH '//SYSUT3 DD   UNIT=SYSDA,DISP=(NEW,DELETE),SPACE=(CYL,(X
                                    +          10,1),RLSE)'
                                 154+    PUNCH '//SYSUT4 DD   UNIT=SYSDA,DISP=(NEW,DELETE),SPACE=(CYL,(X
                                    +          10,1),RLSE)'
                                 155+    PUNCH '//L     EXEC PGM=DFSILNK0,PARM=''XREF,LIST,LET'',REGIOX
                                    +          N=100K,          X'
                                 156+    PUNCH '//            COND=(4,LT,C)'
                                 157+    PUNCH '//SYSLIB DD   DSNAME=SYS1.COBLIB,DISP=SHR'
                                 158+    PUNCH '//       DD   DSNAME=SYS1.PL1LIB,DISP=SHR'
                                 159+    PUNCH '//SYSOBJ DD   DSNAME=IMS.RESLIB,DISP=SHR'
                                 160+    PUNCH '//SYSLIN DD   DSNAME=&&&&LIN,DISP=(OLD,DELETE)'
                                 161+    PUNCH '//       DD   DSNAME=IMS.PROCLIB(DLITCBL),DISP=SHR'
                                 162+    PUNCH '//       DD   DDNAME=SYSIN'
                                 163+    PUNCH '//SYSLMOD DD   DSNAME=IMS.PGMLIB(&&MBR),DISP=SHR'
                                 164+    PUNCH '//SYSPRINT DD SYSOUT=A,DCB=(RECFM=FBA,LRECL=121,BLKSIZEX
                                    +          =605),         X'
                                 165+    PUNCH '//            SPACE=(605,&&PAGES.0,RLSE,,ROUND)'
                                 166+    PUNCH '//SYSUT1 DD   UNIT=SYSDA,DISP=(NEW,DELETE),SPACE=(CYL,(X
                                    +          10,1),RLSE)'
                                 167+    PUNCH './      ADD   NAME=IMSPLI'
                                 168+    PUNCH './      NUMBER NEW1=10,INCR=10'
                                 169+    PUNCH '//      PROC  MBR=,PAGES=50'
                                 170+    PUNCH '//C     EXEC   PGM=IEMAA,PARM=''XREF,ATR,LOAD,NODECK,NX
                                    +          OMACRO,OPT=1'',   X'
                                 171+    PUNCH '//            REGION=114K'
                                 172+    PUNCH '//SYSUT1 DD    UNIT=SYSDA,SPACE=(1024,(60,60),RLSE,,ROUX
                                    +          ND),           X'
                                 173+    PUNCH '//            DCB=(BLKSIZE=1024),DISP=(NEW,PASS)'
                                 174+    PUNCH '//SYSPRINT DD  SYSOUT=A,DCB=(LRECL=121,BLKSIZE=605,RECFX
                                    +          M=FBA),        X'
                                 175+    PUNCH '//            SPACE=(605,(&&PAGES.0,&&PAGES),RLSE)'
                                 176+    PUNCH '//SYSLIN DD    UNIT=SYSDA,SPACE=(80,(250,80),RLSE),DCB=X
                                    +          BLKSIZE=80,    X'
                                 177+    PUNCH '//            DISP=(NEW,PASS)'
                                 178+    PUNCH '//L     EXEC  PGM=DFSILNK0,PARM=''XREF,LIST,LET'',CONDX
                                    +          =(4,LT,C),       X'


IMS/360 SYSTEM DEFINITION SPECIFICATIONS                                                   PAGE    6

LOC  OBJECT CODE    ADDR1 ADDR2  STMT   SOURCE STATEMENT                                 F01JAN68  10/16/68

                                 179+    PUNCH '//             REGION=100K'
                                 180+    PUNCH '//SYSLIB DD    DSNAME=SYS1.PL1LIB,DISP=SHR'
                                 181+    PUNCH '//       DD    DSNAME=SYS1.COBLIB,DISP=SHR'
                                 182+    PUNCH '//SYSLIN DD    DSNAME=*.C.SYSLIN,DISP=(OLD,DELETE)'
                                 183+    PUNCH '//       DD    DSNAME=IMS.PROCLIB(DLITPLI),DISP=SHR'
                                 184+    PUNCH '//       DD    DDNAME=SYSIN'
                                 185+    PUNCH '//SYSLMOD DD   DSNAME=IMS.PGMLIB(&&MBR),DISP=SHR'
                                 186+    PUNCH '//SYSPRINT DD  SYSOUT=A,DCB=(LRECL=121,BLKSIZE=605,RECFX
                                    +          M=FBA),        X'
                                 187+    PUNCH '//             SPACE=(605,(&&PAGES.0,&&PAGES),RLSE)'
                                 188+    PUNCH '//SYSOBJ DD    DSNAME=IMS.RESLIB,DISP=SHR'
                                 189+    PUNCH '//SYSUT1 DD    UNIT=SYSDA,DISP=(NEW,DELETE),SPACE=(CYL,X
                                    +          (5,1),RLSE)'
                                 190+    PUNCH './      ADD   NAME=IMSCOBGO'
                                 191+    PUNCH './      NUMBER NEW1=00000010,INCR=00000010'
                                 192+    PUNCH '//      PROC MBR=,PAGES=60'
                                 193+    PUNCH '//C     EXEC PGM=IEQCBL00,PARM=''SIZE=110000,LINECNT=5X
                                    +          0'',REGION=126K'
                                 194+    PUNCH '//SYSLIN DD   DSNAME=&&&&LIN,DISP=(MOD,PASS),UNIT=SYSDAX
                                    +          ,DCB=(LRECL=80,   X'
                                 195+    PUNCH '//             RECFM=FB,BLKSIZE=400),SPACE=(CYL,(4,1),RX
                                    +          LSE)'
                                 196+    PUNCH '//SYSPRINT DD SYSOUT=A,DCB=(RECFM=FBA,LRECL=121,BLKSIZEX
                                    +          =605),         X'
                                 197+    PUNCH '//             SPACE=(605,(&&PAGES.0,&&PAGES),RLSE,,ROUX
                                    +          ND)'
                                 198+    PUNCH '//SYSUT1 DD   UNIT=SYSDA,DISP=(NEW,DELETE),SPACE=(CYL,(X
                                    +          10,1),RLSE)'
                                 199+    PUNCH '//SYSUT2 DD   UNIT=SYSDA,DISP=(NEW,DELETE),SPACE=(CYL,(X
                                    +          10,1),RLSE)'
                                 200+    PUNCH '//SYSUT3 DD   UNIT=SYSDA,DISP=(NEW,DELETE),SPACE=(CYL,(X
                                    +          10,1),RLSE)'
                                 201+    PUNCH '//SYSUT4 DD   UNIT=SYSDA,DISP=(NEW,DELETE),SPACE=(CYL,(X
                                    +          10,1),RLSE)'
                                 202+    PUNCH '//L     EXEC PGM=DFSILNK0,PARM=''XREF,LIST,LET'',REGIOX
                                    +          N=100K,          X'
                                 203+    PUNCH '//             COND=(4,LT,C)'
                                 204+    PUNCH '//SYSLIB DD   DSNAME=SYS1.COBLIB,DISP=SHR'
                                 205+    PUNCH '//       DD   DSNAME=SYS1.PL1LIB,DISP=SHR'
                                 206+    PUNCH '//SYSOBJ DD   DSNAME=IMS.RESLIB,DISP=SHR'
                                 207+    PUNCH '//SYSLIN DD   DSNAME=&&&&LIN,DISP=(OLD,DELETE)'
                                 208+    PUNCH '//       DD   DSNAME=IMS.PROCLIB(DLITCBL),DISP=SHR'
                                 209+    PUNCH '//       DD   DDNAME=SYSIN'
                                 210+    PUNCH '//SYSLMOD DD   DSNAME=IMS.PGMLIB(&&MBR),DISP=SHR'
                                 211+    PUNCH '//SYSPRINT DD SYSOUT=A,DCB=(RECFM=FBA,LRECL=121,BLKSIZEX
                                    +          =605),         X'
                                 212+    PUNCH '//             SPACE=(605,&&PAGES.0,RLSE,,ROUND)'
                                 213+    PUNCH '//SYSUT1 DD   UNIT=SYSDA,DISP=(NEW,DELETE),SPACE=(CYL,(X
                                    +          10,1),RLSE)'
                                 214+    PUNCH '//G     EXEC   PGM=DFSIRC00,PARM=''3,&&MBR'',REGION=150X
                                    +          K,COND=(0,LT),    X'
                                 215+    PUNCH '//             TIME=2'
                                 216+    PUNCH '//IMS    DD    DSNAME=IMS.PSBLIB,DISP=SHR'
                                 217+    PUNCH '//       DD    DSNAME=IMS.DBDLIB,DISP=SHR'
                                 218+    PUNCH '//SYSOUT DD    SYSOUT=A,SPACE=(CYL,(1,1)),DCB=(LRECL=13X
     +                 3,RECFM=FA)'
   219+         PUNCH '//SYSUDUMP DD  SYSOUT=A,DCB=(LRECL=121,RECFM=FBA,BLKSIZX
     +                 E=3025),       X'
   220+         PUNCH '//             SPACE=(3025,(200,100),RLSE,,ROUND)'
   221+         PUNCH './       ADD   NAME=IMSPLIGO'
   222+         PUNCH './       NUMBER NEW1=10,INCR=10'
   223+         PUNCH '//       PROC  MBR=,PAGES=50'
   224+         PUNCH '//C      EXEC   PGM=IEMAA,PARM=''XREF,ATR,LOAD,NODECK,NX
     +                 OMACRO,OPT=1'',  X'
   225+         PUNCH '//             REGION=114K'
   226+         PUNCH '//SYSUT1 DD    UNIT=SYSDA,SPACE=(1024,(60,60),RLSE,,ROUX
     +                 ND),           X'
   227+         PUNCH '//             DCB=(BLKSIZE=1024),DISP=(NEW,PASS)'
   228+         PUNCH '//SYSPRINT DD  SYSOUT=A,DCB=(LRECL=121,BLKSIZE=605,RECFX
     +                 M=FBA),        X'
   229+         PUNCH '//             SPACE=(605,(&&PAGES.0,&&PAGES),RLSE)'
   230+         PUNCH '//SYSLIN DD    UNIT=SYSDA,SPACE=(80,(250,80),RLSE),DCB=X
     +                 BLKSIZE=80,    X'
   231+         PUNCH '//             DISP=(NEW,PASS)'
   232+         PUNCH '//L      EXEC  PGM=DFSILNKO,PARM=''XREF,LIST,LET'',CONDX
     +                 =(4,LT,C),       X'
   233+         PUNCH '//             REGION=100K'
   234+         PUNCH '//SYSLIB DD    DSNAME=SYS1.PL1LIB,DISP=SHR'
   235+         PUNCH '//       DD    DSNAME=SYS1.COBLIB,DISP=SHR'
   236+         PUNCH '//SYSLIN DD    DSNAME=*.C.SYSLIN,DISP=(OLD,DELETE)'
   237+         PUNCH '//       DD    DSNAME=IMS.PROCLIB(DLITPLI),DISP=SHR'
   238+         PUNCH '//       DD    DDNAME=SYSIN'
   239+         PUNCH '//SYSLMOD DD   DSNAME=IMS.PGMLIB(&&MBR),DISP=SHR'
   240+         PUNCH '//SYSPRINT DD  SYSOUT=A,DCB=(LRECL=121,BLKSIZE=605,RECFX
     +                 M=FBA),        X'
   241+         PUNCH '//             SPACE=(605,(&&PAGES.0,&&PAGES),RLSE)'
   242+         PUNCH '//SYSOBJ DD    DSNAME=IMS.RESLIB,DISP=SHR'
   243+         PUNCH '//SYSUT1 DD    UNIT=SYSDA,DISP=(NEW,DELETE),SPACE=(CYL,X
     +                 (5,1),RLSE)'
   244+         PUNCH '//G      EXEC  PGM=DFSIRC00,PARM=''3,&&MBR'',COND=(4,LTX
     +                 ),REGION=150K,   X'
   245+         PUNCH '//             TIME=5'
   246+         PUNCH '//IMS    DD    DSNAME=IMS.PSBLIB,DISP=SHR'
   247+         PUNCH '//       DD    DSNAME=IMS.DBDLIB,DISP=SHR'
   248+         PUNCH '//SYSPRINT DD  SYSOUT=A,DCB=(LRECL=121,BLKSIZE=605,RECFX
     +                 M=FBA),        X'
   249+         PUNCH '//             SPACE=(605,(500,500),RLSE,,ROUND)'
   250+         PUNCH '//SYSUDUMP DD  SYSOUT=A,DCB=(LRECL=121,BLKSIZE=605,RECFX
     +                 M=FBA),        X'
   251+         PUNCH '//             SPACE=(605,(500,500),RLSE,,ROUND)'
   252+         PUNCH './       ADD   NAME=PSBGEN'
   253+         PUNCH './       NUMBER NEW1=10,INCR=10'
   254+         PUNCH '//       PROC  MBR=TEMPNAME'
   255+         PUNCH '//C      EXEC  PGM=IEUASM,PARM=''LOAD,NODECK'',REGION=9X
     +                 2K'
   256+         PUNCH '//SYSLIB DD    VOLUME=SER=ILI802,DISP=SHR,             X
     +                              CONTINUE'
   257+         PUNCH '//             DSNAME=IMS.MACLIB,UNIT=2311'
   258+         PUNCH '//       DD    DSNAME=SYS1.MACLIB,DISP=SHR'
   259+         PUNCH '//SYSGO  DD    UNIT=SYSDA,DISP=(,PASS),DCB=(BLKSIZE=400X
```

```
     +                 ,              X'
   260+         PUNCH '//             RECFM=FB,LRECL=80),SPACE=(80,(100,100),RX
     +                 LSE)'
   261+         PUNCH '//SYSPRINT DD  SYSOUT=A,DCB=(LRECL=121,RECFM=FBA,BLKSIZX
     +                 E=605),        X'
   262+         PUNCH '//             SPACE=(121,(500,500),RLSE,,ROUND)'
   263+         PUNCH '//SYSUT1 DD    UNIT=SYSDA,DISP=(,DELETE),SPACE=(1700,(1X
     +                 00,50))'
   264+         PUNCH '//SYSUT2 DD    UNIT=SYSDA,DISP=(,DELETE),SPACE=(1700,(1X
     +                 00,50))'
   265+         PUNCH '//SYSUT3 DD    UNIT=(SYSDA,SEP=(SYSLIB,SYSUT1,SYSUT2)),X
     +                                X'
   266+         PUNCH '//             SPACE=(1700,(100,50))'
   267+         PUNCH '//L      EXEC  PGM=DFSILNKO,PARM=''XREF,LIST'',COND=(0,X
     +                 LT,C),           X'
   268+         PUNCH '//             REGION=100K'
   269+         PUNCH '//SYSLIN DD    DSNAME=*.C.SYSGO,DISP=(OLD,DELETE)'
   270+         PUNCH '//       DD    DDNAME=SYSIN'
   271+         PUNCH '//SYSPRINT DD  SYSOUT=A,DCB=(LRECL=121,RECFM=FBA,BLKSIZX
     +                 E=605),        X'
   272+         PUNCH '//             SPACE=(121,(100,100),RLSE)'
   273+         PUNCH '//SYSLMOD DD   DSNAME=IMS.PSBLIB(&&MBR),DISP=SHR'
   274+         PUNCH '//SYSUT1 DD    UNIT=(SYSDA,SEP=(SYSLMOD,SYSLIN)),DISP=(X
     +                 ,DELETE),      X'
   275+         PUNCH '//             SPACE=(1024,(100,10),RLSE)'
   276+         PUNCH './       ADD   NAME=DBDGEN'
   277+         PUNCH './       NUMBER NEW1=10,INCR=10'
   278+         PUNCH '//       PROC  MBR=TEMPNAME'
   279+         PUNCH '//C      EXEC  PGM=IEUASM,PARM=''LOAD,NODECK'',REGION=9X
     +                 2K'
   280+         PUNCH '//SYSLIB DD    VOLUME=SER=ILI802,DISP=SHR,,            X
     +                              CONTINUE'
   281+         PUNCH '//             DSNAME=IMS.MACLIB,UNIT=2311'
   282+         PUNCH '//       DD    DSNAME=SYS1.MACLIB,DISP=SHR'
   283+         PUNCH '//SYSGO  DD    UNIT=SYSDA,DISP=(,PASS),DCB=(BLKSIZE=400X
     +                 ,              X'
   284+         PUNCH '//             RECFM=FB,LRECL=80),SPACE=(80,(100,100),RX
     +                 LSE)'
   285+         PUNCH '//SYSPRINT DD  SYSOUT=A,DCB=(LRECL=121,RECFM=FBA,BLKSIZX
     +                 E=605),        X'
   286+         PUNCH '//             SPACE=(121,(500,500),RLSE,,ROUND)'
   287+         PUNCH '//SYSUT1 DD    UNIT=SYSDA,DISP=(,DELETE),SPACE=(1700,(1X
     +                 00,50))'
   288+         PUNCH '//SYSUT2 DD    UNIT=SYSDA,DISP=(,DELETE),SPACE=(1700,(1X
     +                 00,50))'
   289+         PUNCH '//SYSUT3 DD    UNIT=(SYSDA,SEP=(SYSLIB,SYSUT1,SYSUT2)),X
     +                                X'
   290+         PUNCH '//             SPACE=(1700,(100,50))'
   291+         PUNCH '//L      EXEC  PGM=DFSILNKO,PARM=''XREF,LIST'',COND=(0,X
     +                 LT,C),           X'
   292+         PUNCH '//             REGION=100K'
   293+         PUNCH '//SYSLIN DD    DSNAME=*.C.SYSGO,DISP=(OLD,DELETE)'
   294+         PUNCH '//       DD    DDNAME=SYSIN'
   295+         PUNCH '//SYSPRINT DD  SYSOUT=A,DCB=(LRECL=121,RECFM=FBA,BLKSIZX
     +                 E=605),        X'
     IMS/360 SYSTEM DEFINITION SPECIFICATIONS                                                    PAGE    9

LOC  OBJECT CODE     ADDR1 ADDR2  STMT   SOURCE STATEMENT                                       F01JAN68  10/16/68

                                  296+          PUNCH '//                  SPACE=(121,(100,100),RLSE)'
                                  297+          PUNCH '//SYSLMOD DD   DSNAME=IMS.DBDLIB(&&MBR),DISP=SHR'
                                  298+          PUNCH '//SYSUT1 DD    UNIT=(SYSDA,SEP=(SYSLMOD,SYSLIN)),DISP=(X
                                     +                ,DELETE),       X'
                                  299+          PUNCH '//                  SPACE=(1024,(100,10),RLSE)'
                                  300+          PUNCH './         ENDUP'
                                  301+          PUNCH '/*'
                                  302+          PUNCH '//STEP5  EXEC  PGM=IEUASM,PARM=''LOAD,NODECK'',REGION=9X
                                     +                2K'
                                  303+          PUNCH '//SYSLIB DD    DSNAME=IMS.GENLIB,DISP=(OLD,PASS)'
                                  304+          PUNCH '//        DD    DSNAME=SYS1.MACLIB,DISP=SHR'
                                  305+          PUNCH '//SYSGO  DD    UNIT=SYSDA,DISP=(,PASS),DCB=(LRECL=80,BLX
                                     +                KSIZE=400,      X'
                                  306+          PUNCH '//                  RECFM=FB),SPACE=(TRK,(10,10),RLSE)'
                                  307+          PUNCH '//SYSPRINT DD  SYSOUT=A,DCB=(LRECL=121,BLKSIZE=605,RECFX
                                     +                M=FBA),         X'
                                  308+          PUNCH '//                  SPACE=(605,(100,50),RLSE,,ROUND)'
                                  309+          PUNCH '//SYSUT1 DD    UNIT=SYSDA,DISP=(,DELETE),SPACE=(1700,(1X
                                     +                00,50))'
                                  310+          PUNCH '//SYSUT2 DD    UNIT=SYSDA,DISP=(,DELETE),SPACE=(1700,(1X
                                     +                00,50))'
                                  311+          PUNCH '//SYSUT3 DD    UNIT=(SYSDA,SEP=(SYSLIB,SYSUT1,SYSUT2)),X
                                     +                DISP=(,DELFTE), X'
                                  312+          PUNCH '//                  SPACE=(1700,(100,50))'
                                  313+          PUNCH '//SYSIN  DD    *'
                                  314+          PUNCH 'DFSISCD  CSECT'
                                  315+          PUNCH '         PRINT ON'
                                  316+          PUNCH '         IMSBATCH CENDA=WA,SPVSVC=245'
                                  317+          PUNCH '         ISCD     SECTYPE=CSECT'
                                  318+          PUNCH '         END'
                                  319+          PUNCH '/*'
                                  320+          PUNCH '//STEP6  EXEC  PGM=IEWL,PARM=''RENT,NCAL,LIST,XREF'',REX
                                     +                GION=110K'
                                  321+          PUNCH '//SYSPRINT DD  SYSOUT=A,DCB=(LRECL=121,BLKSIZE=605,RECFX
                                     +                M=FBA),         X'
                                  322+          PUNCH '//                  SPACE=(605,(10,10),RLSE,,ROUND)'
                                  323+          PUNCH '//SYSLIN DD    DSNAME=*.STEP5.SYSGO,DISP=(OLD,DELETE)'
                                  324+          PUNCH '//        DD    DDNAME=SYSIN'
                                  325+          PUNCH '//SYSOBJ DD    VOLUME=SER=ILIB01,DISP=(OLD,PASS),       X
                                     +                          CONTINUE'
                                  326+          PUNCH '//                  DSNAME=IMS.RESLIB,UNIT=2311'
                                  327+          PUNCH '//SYSLMOD DD   VOLUME=SER=ILIB01,DISP=(OLD,PASS),       X
                                     +                          CONTINUE'
                                  328+          PUNCH '//                  DSNAME=IMS.RESLIB,UNIT=2311'
                                  329+          PUNCH '//SYSUT1 DD    UNIT=(SYSDA,SEP=(SYSLIN,SYSLMOD)),DISP=(X
                                     +                ,DELETE),       X'
                                  330+          PUNCH '//                  SPACE=(1700,(100,50))'
                                  331+          PUNCH '//SYSIN  DD    *'
                                  332+          PUNCH '  INCLUDE SYSOBJ(DFSIOS10)   OSAM OPEN ROUTINE'
                                  333+          PUNCH '  INCLUDE SYSOBJ(DFSISMM0)   STORAGE MANAGEMENT MODULE'
                                  334+          PUNCH '  INCLUDE SYSOBJ(DFSIDL00)   DL/I OPEN MODULE'
                                  335+          PUNCH '  INCLUDE SYSOBJ(DFSIDLC0)   DL/I CLOSE MODULE'
                                  336+          PUNCH '  INCLUDE SYSOBJ(DFSIDBA0)   DL/I BATCH ANALYZER'
                                  337+          PUNCH '  INCLUDE SYSOBJ(DFSIBKB0)   DL/I BATCH BLOCK MODULE'
                                  338+          PUNCH '  ENTRY DFSSTART'


     IMS/360 SYSTEM DEFINITION SPECIFICATIONS                                                    PAGE   10

LOC  OBJECT CODE     ADDR1 ADDR2  STMT   SOURCE STATEMENT                                       F01JAN68  10/16/68

                                  339+          PUNCH '  NAME DFSIDLB0(R)          DL/I BATCH NUCLEUS'
                                  340+    -     PUNCH '/*'


                                  342                 *,*** SUCCESSFUL IMS/360 SYSTEM DEFINITION
                                  343                 *,    DEFINITION IS FOR BATCH IMS/360 FUNCTIONS.
```

```
LOC  OBJECT CODE    ADDR1 ADDR2  STMT   SOURCE STATEMENT                                          F01JAN68  10/16/68
                                 345*** WARNING **
                                 346                  *,****************************************************
                                 347                  *,****************************************************
                                 348                  *,                                                   *
                                 349                  *,    IGG019WA MUST BE MOVED TO SYS1.SVCLIB AND IGC245
                                 350                  *,MUST BE LINK EDITED WITH THE OS/360 NUCLEUS FOR
                                 351                  *,SUCCESSFUL IMS/360 SYSTEM EXECUTION.
                                 352                  *,
                                 353                  *,    DFSILNK0 AND DFSIRC00 SHOULD BE IN SYS1.LINKLIB
                                 354                  *,FOR EFFICIENT IMS/360 SYSTEM OPERATION.
                                 355                  *,
                                 356                  *,    SEE IMS/360 AND OS/360 SYSTEM OPERATION MANUALS
                                 357                  *,FOR MODULES TO BE PLACED IN LINK PACK AREA FOR
                                 358                  *,EFFICIENT SYSTEM OPERATION.
                                 359                  *,
                                 360                  *,    USER SHOULD OBTAIN A PDS DIRECTORY LISTING OF
                                 361                  *,THE LIBRARIES CREATED BY STAGE II OF IMS/360 SYSTEM
                                 362                  *,GENERATION.
                                 363                  *,
                                 364                  *,    STAGE II OF IMS/360 SYSTEM GENERATION WILL PLACE
                                 365                  *,ALL SYSTEM LOAD MODULES IN IMS.RESLIB.
                                 366                  *,                                                   *
                                 367                  *,****************************************************
                                 368                  *,****************************************************
                                 369          END

NO STATEMENTS FLAGGED IN THIS ASSEMBLY
  466 PRINTED LINES
```

SECURITY MAINTENANCE

Although IMS/360 system definition creates the majority of resident control blocks for the IMS/360 control program, it does not supply security capabilities. These capabilities are supplied in IMS/360 through a security maintenance program which allows the IMS/360 user the flexibility of changing security information without redefining his entire system. Security is provided by terminal and by password.

The reader should be familiar with IMS/360 system definition to obtain the best use of the following information.

The function of the security maintenance program (SMP) is to create or alter password or terminal protection of an online IMS/360 system. The generated IMS/360 system has only a minimum subset of terminal security to protect DISPLAY, NRESTART, CHECKPOINT, ERESTART, START, CHANGE, STOP, PURGE, DBRECOVERY, DBLOG, DBNOLOG, DBDUMP, ASSIGN, DELETE, and PSTOP commands. The security maintenance program creates password and terminal security for transactions and additional commands entered from terminals; it also creates password security on data bases and programs. The control of the security maintenance program is such that the user may view his system in terms of resources to which passwords may have access, or he may view the system as a security profile system, that is, by defining a password which has access to a set of resources. The detailed explanation covers the use of the various control cards to describe either a "profile-oriented" system or a "resource-oriented" system of security maintenance. There is no restriction on the use of both types of description within the same security maintenance program execution.

Password Maintenance

If password maintenance control cards are presented in the input stream for the SMP, the password maintenance function is performed. Using the SMP password control cards, the following functions are available:

- Add passwords to or delete passwords from the IMS/360 communication password table (CPT).

- Change the password security requirements for transaction codes, terminal command verbs, program status changes, data base status changes, and logical or physical terminal status changes.

IMS/360 password table and password matrix changes become effective the next time IMS/360 is restarted. If the next restart is a "cold start", the master terminal operator may specify that the system-defined status be used or that the new table and matrix be used. If the next restart is a "warm start", the master terminal operator may specify that the current status of the password table and matrix is to be restored using the system checkpoint records, or that the new password table and matrix are to be used.

Terminal Security Maintenance

If terminal security maintenance control cards are presented in the input stream for the SMP, maintenance functions are performed upon the IMS/360 communications terminal matrix. Using the SMP terminal security control cards, the following function is available:

- Add to or delete from terminal security requirements for command verbs and application program transaction codes.

Terminal security changes become effective the next time IMS/360 is restarted. If the next restart is a cold start, the master terminal operator may specify that the system-defined status be restored or that the new terminal matrix is to be used. If the next start is a warm start, the master terminal operator may specify that the current status of terminal security be restored using system checkpoint records, or that the new terminal security matrix is to be used.

The security maintenance program will not execute until an IMS/360 system definition has been performed. Input requirements for the SMP include an IMS/360 system description block (SDB), which is created at system definition time and which must reside in the same library with the IMS/360 control program nucleus. If multiple IMS/360 systems exist, the SMP maintains as many as nine sets of security control blocks in the same library. If errors are encountered in processing SMP control cards, no security block update functions are performed. Diagnostic error messages are produced for the entire input stream. At user option, the SMP performs a no-update run, producing a printed analysis of IMS/360 security requirements. In addition, each execution of the SMP produces a printed analysis of the IMS/360 configuration being maintained.

Control and Data Statements

The security maintenance program control and data statements available are PASSWORD, TERMINAL, TRANSACT, COMMAND, DATABASE, PROGRAM, and PTERM. In general, each of these cards may be used as required. The specifications to be considered in designing a password security system must be tailored to the particular environment in which IMS/360 is to run. The control cards above are used to describe the security environment that the IMS/360 system is to use in processing messages and commands.

Control statements are identified by)(characters (close and open parentheses in combination) in positions 1 and 2, followed by a blank in column 3. Data statements are identified by a blank in position 1. A control statement remains in effect until another control statement or end of input data is encountered. Each statement, control or data, has only one allowable operand. Valid combinations of control and data statements are shown in Figure 24.

NAME	OPERATION	OPERAND
)(	PASSWORD	password
	TERMINAL	logical terminal name
	TRANSACT	transaction code
	COMMAND	command language verb
	DATABASE	name
	PROGRAM	name
	PTERM	name
)(	TERMINAL	logical terminal name
	PASSWORD	password
	TRANSACT	transaction code
	COMMAND	command language verb
)(	TRANSACT	transaction code
	PASSWORD	password
	TERMINAL	logical terminal name
)(	COMMAND	command language verb
	PASSWORD	password
	TERMINAL	logical terminal name
)(	DATABASE	name
	or	
)(	PROGRAM	name
	or	
)(	PTERM	name
	PASSWORD	password

where:

password

A password must contain only alphameric characters and may be one through eight characters in length. The longest password statement encountered in the input stream governs the maximum length of the input password that will be accepted by the system.

Data statements are terminal transact command, data base, program, and PTERM.

logical terminal name

A valid logical terminal name may be one through eight characters in length. Terminal names that are not defined in the system being maintained are invalid and will be rejected by the security maintenance program.

transaction code

A valid transaction code may be one through eight characters in length and must be defined in the IMS/360 online system being maintained. If it is not, it is treated as invalid by the security maintenance program.

name

A valid data base name, program name, or physical terminal number is available from Stage 2 output of IMS/360 system definition.

command language verb

Valid command language verbs may be obtained from the Stage 2 output of IMS/360 system definition. The command verb, less leading slash, may be abbreviated to the first three characters.

Notes: Only the first three characters of the operation code are required to identify control or data statements. Physical terminal numbers may be found in the terminal map printed in the assembly of DFSISDB0 in Stage 2 of IMS/360 system definition.

To define additional passwords, a PASSWORD control statement may be used with no following data statements:

```
)(  PASSWORD  ABCD
)(  PASSWORD  EFGH
```

DATA CARD TYPE	CONTROL CARD TYPE						
	PASSWORD	TERMINAL	TRANSACT	COMMAND	DATABASE	PROGRAM	PTERM
PASSWORD	NO	YES	YES	YES	YES	YES	YES
TERMINAL	YES	NO	YES	YES	NO	NO	NO
TRANSACT	YES	YES	NO	NO	NO	NO	NO
COMMAND	YES	YES	NO	NO	NO	NO	NO
DATABASE	YES	NO	NO	NO	NO	NO	NO
PROGRAM	YES	NO	NO	NO	NO	NO	NO
PTERM	YES	NO	NO	NO	NO	NO	NO

Figure 24. Security maintenance control and data card types

Control and Data Statement Combinations

The following outlines the use of various control and data statement combinations:

	Control Statement	Data Statement	Explanation
1.	PASSWORD TERMINAL	TERMINAL PASSWORD	To require a password to be used with the logical terminal name when modifying the status of a logical terminal via a /LOCK, /UNLOCK, or /IAM command
2.	PASSWORD TRANSACT	TRANSACT PASSWORD	To require a password to be entered from the input terminal following the transaction code for each message
3.	PASSWORD COMMAND	COMMAND PASSWORD	To require a password to be entered following the command verb when using the terminal command language
4.	PASSWORD DATABASE	DATABASE PASSWORD	To require a password to be entered following the data base name when modifying the status of a data base via a /LOCK or /UNLOCK command
5.	PASSWORD PROGRAM	PROGRAM PASSWORD	To require a password to be entered following the program name when modifying the status of a program (PSB) via a /LOCK or /UNLOCK command
6.	PASSWORD PTERM	PTERM PASSWORD	To require a password to be entered following the keyword PTERM when modifying the status of a physical terminal via a /LOCK, /UNLOCK, or /IAM command
7.	TERMINAL TRANSACT	TRANSACT TERMINAL	To restrict use of a transaction code to a specific logical terminal. Note: Entry of the named transaction codes will be only permitted from the terminals specified.
8.	COMMAND TERMINAL	TERMINAL COMMAND	To restrict use of a command verb to specific logical terminals

Input statements may be used as control cards or data cards. Using the input statements, security requirements may be expressed as either profile-oriented or resource-oriented. A profile security system describes the resources to be secured in terms of the securing element. For example, the following describes a profile for password SAMSMITH.

```
)( PASSWORD SAMSMITH
        TRANSACT    PAYROLL
        TRANSACT    PERS
        COMMAND     LOCK
        COMMAND     UNLOCK
        DATABASE    PAYREC
        PROGRAM     PAYPROG
```

To describe these same security requirements by resource, the following statements are required.

```
)(      TRANSACT    PAYROLL
        PASSWORD    SAMSMITH
)(      TRANSACT    PERS
        PASSWORD    SAMSMITH
)(      COMMAND     LOCK
        PASSWORD    SAMSMITH
)(      COMMAND     UNLOCK
        PASSWORD    SAMSMITH
)(      DATABASE    PAYREC
        PASSWORD    SAMSMITH
)(      PROGRAM     PAYPROG
        PASSWORD    SAMSMITH
```

As the preceding example illustrates, passwords may be more easily described by using the securing elements as data. Terminal security, however, is more easily described by using the secured element, the transaction, as a control statement, followed by the security elements, the terminals, as data.

```
)( TRANSACT PAYROLL
        TERMINAL    DEPT40
        TERMINAL    DEPT65
        TERMINAL    VPPERS
)( TRANSACT PERS
        TERMINAL    DEPT40
```

The reverse or profile example would be:

```
)( TERMINAL         DEPT40
        TRANSACT    PAYROLL
        TRANSACT    PERS
)( TERMINAL         DEPT65
        TRANSACT    PAYROLL
)( TERMINAL         VPPERS
        TRANSACT    PAYROLL
```

The basic online system provides terminal security only for a subset of the command language. The following example would secure a more typical set of commands against entry from any terminal except the master terminal:

```
)( TERMINAL  master terminal name
   COMMAND     START
   COMMAND     STOP
   COMMAND     NRESTART
   COMMAND     CHECKPOINT
   COMMAND     PSTOP
   COMMAND     ERESTART
   COMMAND     DBRECOVERY
   COMMAND     ASSIGN
   COMMAND     BROADCAST
   COMMAND     CHANGE
   COMMAND     DBDUMP
   COMMAND     DUMPQ
   COMMAND     PURGE
   COMMAND     LOG
```

Description of SMP Output

The security maintenance program produces three printed reports. The first report is the logical configuration of system being maintained, the second is the password table generated, and the third is the matrix for the security of a particular nucleus.

Security Maintenance Program Execution

The security maintenance run is a three-step job. The first step accepts the input control and data cards for the security maintenance program and edits them for correct format and validity against the IMS/360 system being maintained. If there are no errors in the first step, the second step, an Operating System/360 assembly, will be performed. Step three is a link-edit which takes the assembly output from step two and creates the communication password table, communications password matrix, and communication terminal matrix load modules used by the IMS/360 control program. Depending upon the input presented, a variable number of output load modules will be created.

The maximum bounds of the generated matrices, terminal or password are expressed as:

$$(I/8) * R = M = < 32767$$

where:

M is the total main storage requirement in bytes.

I is the number of securing resources (passwords or logical terminals).

R is the number of unique combinations of secured resources.

The maximum number of entries in the password table is expressed as:

$$I/8 = < 32768$$

where I is the total number of passwords.

To perform a security maintenance run, the user must have previously defined an IMS/360 control program using the value ALL as the second sublist entry of the SYSTEM operand of the IMSCTRL macro-instruction. One of the modules created during Stage 2 of IMS/360 system definition is a directory of resources of the defined system, which is placed in the IMS.RESLIB data set. This directory and the security maintenance control cards comprise the input requirements for the security maintenance program (SMP). Output from the SMP consists of four sequential members in IMS.RESLIB. These members may not be reprocessed using the linkage editor. The four members contain:

1. Communication Password Table (CPT)
2. Communication Terminal Matrix (CTM)
3. Terminal Offset List (CTL)
4. Password Offset List (CPL)

In addition, the SMP provides a listing of the created maintenance tables. Each run of the SMP replaces previously created members. Figure 25 depicts the security maintenance flow.

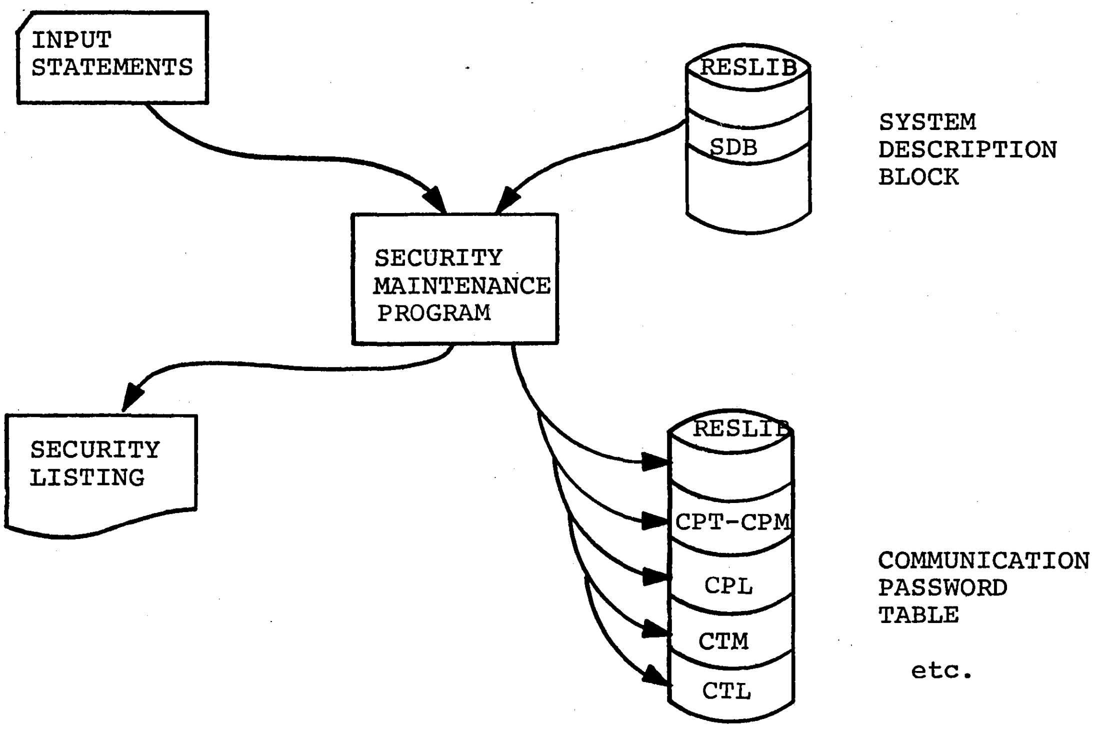


Figure 25. Security maintenance flow

The table below shows the Job Control statements by step necessary to execute the security maintenance utility.

STATEMENT	USAGE
JOB statement	Initiates security maintenance job
JOBLIB statement	Defines the partitioned data set named in RESLIB macro-statement during IMS system definition. Contains the members DFSINUCn and DFSISMP0.
step S	
EXEC statement	Specifies the program name (PGM=DFSISMP0) and may contain a PARM keyword value of the form PARM = <u>'UPDATE,0'</u> 'option,number' option LIST - validity check and list new security tables UPDATE - validity check, list, and update security tables in RESLIB number a value ranging from 0-9 which is the last character of the IMS/360 nucleus member name to be maintained

SYSPRINT DD statement	Defines a sequential message data set. The data set can be written to system output devices, magnetic tape, or direct access volumes. The following DCB parameters must be specified:
	RECFM=VBA
	BLKSIZE=125 or greater
	BUFL=value of BLKSIZE + 4

SYSPUNCH DD statement	Defines a sequential output data set which contains Assembler statements produced by step S. The data set may be passed to step C. The following DCB parameters must be specified:
	RECFM=F or FB
	LRECL = 80
	BLKSIZE = 80 or multiple of 80

SYSLIN DD statement	Defines a sequential output data set which contains linkage editor control statements produced by step S. The data set may be passed to step L. The following DCB parameters must be specified:
	RECFM=F or FB
	LRECL = 80
	BLKSIZE = 80 or multiple of 80

SYSUT1 DD statement	Defines a sequential work data set used only during step S. The following DCB parameters must be specified:
	RECFM=F or FB
	BLKSIZE = 100 or multiple of 100

SYSUT2 DD statement	Defines a sequential work data set used only during step S. The following DCB parameters must be specified:
	RECFM=F or FB
	BLKSIZE = 100 or multiple of 100

SYSIN DD statement	Defines a sequential data set or a member of a partitioned data set which contains security maintenance input statements. The following DCB parameters must be specified:
	RECFM=F or FB
	BLKSIZE = 80 or multiple of 80

step C	

EXEC statement	Specifies the program name (PGM=IEUASM) of the assembler. Following parameters must be present:
	PARM='LOAD,NODECK'
	COND=(12,LT,S)

SYSPRINT DD statement	Defines a sequential message data set. The data set can be written to system output devices, magnetic tape, or direct access volumes. The following DCB parameters must be specified:
	RECFM=FM or FBM
	LRECL = 121
	BLKSIZE=121 or multiple of 121

SYSGO DD statement	Defines a sequential temporary data set for object output from the assembler. The data set may be passed to step L.
SYSUT1 SYSUT2 SYSUT3 DD statements	Defines sequential data sets used for work space by the assembler only during step C.
SYSIN DD statement	Defines passed sequential input data set created in step S using DD name SYSPUNCH.

step L	

EXEC statement	Specifies the program name (PGM=IEWL) of the linkage editor. Following parameters must be present:
	PARM='LIST,NE,OL'
	COND=(4,LT,S)

SYSPRINT DD statement	Defines a sequential message data set for the linkage editor. The data set can be written to system output devices, magnetic tape, or direct access volumes. The following DCB parameters must be specified:
	RECFM=FA or FBA
	LRECL=121
	BLKSIZE=121 or multiple of 121

SYSLMOD DD statement	Defines output partitioned data set for the linkage editor. Normally the same data set specified for DD name JOBLIB.
INPUT DD statement	Defines passed sequential temporary data set created using DD name SYSGO in step C.
SYSUT1 DD statement	Defines sequential temporary data set used in step L by the linkage editor.
SYSLIN DD statement	Defines passed sequential temporary data set created using DD name SYSLIN in step S.

Once created, these new matrices and the password table are not made available to the online system until a restart is performed. At normal restart time, the operator has the option of incorporating or not incorporating the newly created security tables. At either cold start (that is, NRESTART CHECKPOINT 0) or warm restart (NRESTART any checkpoint number), the new security tables are not included unless specifically requested by the system operator. The two keyword operands of the NRESTART command, which are used to request new security, are PASSWORD, for password security, and TERMINAL, for terminal security. Once these two keywords are used in a normal restart, the system checkpoint facility causes the new security maintenance to continue through subsequent warm starts. If the user desires, once a normal successful restart using the normal keywords has been accomplished, he may change his system security configuration. Again, these changes will not become effective until the user specifically requests them at normal restart time.

SECURITY MAINTENANCE EXAMPLE

The following is an example of the input cards for the security maintenance program that reflects the system definition example in this chapter. This example assumes:

- A password exists for each program.
- A password exists for each data base.
- A password exists for each transaction code except INQUIRY.
- The list of terminals can use each transaction code, along with the required password.
- Some IMS/360 terminal commands are limited to the master terminal.
- The master terminal can enter all IMS/360 terminal commands and transaction codes defined by the system definition example in this manual.

```
)( PROGRAM      ACCT
    PASSWORD    DOLLAR

)( PROGRAM      ENG560
    PASSWORD    PARTNO

)( PROGRAM      LOGREC
    PASSWORD    NONE

)( PROGRAM      AGC0568
    PASSWORD    MONEY

)( DATABASE     ACCTLOG
    PASSWORD    LOG

)( DATABASE     ACCTREC
    PASSWORD    REC

)( DATABASE     ACTIVITY
    PASSWORD    ACTIVE

)( DATABASE     ENGREC
    PASSWORD    PIERSQ

)( DATABASE     PARTSREC
    PASSWORD    PIERSQ

)( DATABASE     PARTSREC
    PASSWORD    ASSY

)( TRANSACT     ACCTCHG
    PASSWORD    CHARGE
    TERMINAL    A875111
    TERMINAL    C8751112
    TERMINAL    D8751113
    TERMINAL    A8751114
    TERMINAL    A8751115

)( TRANSACT     ACTY
    PASSWORD    GO
```

```
            TERMINAL    A8751111
            TERMINAL    C8751112
            TERMINAL    D8751113
            TERMINAL    A8751114
            TERMINAL    A8751115

        )( TRANSACT     TNL
            PASSWORD    QTY
            TERMINAL    DEPT 650
            TERMINAL    DEPT 610
            TERMINAL    DEPT 620
            TERMINAL    DEPT 631
            TERMINAL    DEPT 632
            TERMINAL    DEPT 630
            TERMINAL    DEPT 640
            TERMINAL    DEPT 641
            TERMINAL    DEPT 642

        )( TRANSACT     ING
            PASSWORD    QUESTION
            TERMINAL    DEPT310
            TERMINAL    DEPT311
            TERMINAL    DEPT312
            TERMINAL    DEPT410
            TERMINAL    DEPT411
            TERMINAL    DEPT412
            TERMINAL    DEPT510
            TERMINAL    DEPT511
            TERMINAL    DEPT512
            TERMINAL    DEPT100
            TERMINAL    DEPT200
            TERMINAL    DEPT686
            TERMINAL    MASTER
            TERMINAL    ALTMAST
            TERMINAL    MAINT
            TERMINAL    DEPT710
            TERMINAL    DEPT720
            TERMINAL    DEPT848
            TERMINAL    DEPT850
            TERMINAL    DEPT900
            TERMINAL    TEST1
            TERMINAL    TEST2

  |     )( TRANSACT     INVNTRY
            PASSWORD    SUBASSY
            TERMINAL    DEPT310
            TERMINAL    DEPT311
            TERMINAL    DEPT312
            TERMINAL    DEPT410
            TERMINAL    DEPT411
            TERMINAL    DEPT412
            TERMINAL    DEPT510
            TERMINAL    DEPT511
            TERMINAL    DEPT512
            TERMINAL    DEPT100
            TERMINAL    DEPT200
            TERMINAL    DEPT686
            TERMINAL    MASTER
            TERMINAL    ALTMAST
            TERMINAL    MAINT
            TERMINAL    DEPT710
            TERMINAL    DEPT720
            TERMINAL    DEPT848
            TERMINAL    DEPT850
            TERMINAL    DEPT900
```

```
    TERMINAL    TEST1
    TERMINAL    TEST2

)( TRANSACT     ACCT
    PASSWORD    LEDGER
    TERMINAL    DEPT310
    TERMINAL    DEPT311
    TERMINAL    DEPT312
    TERMINAL    DEPT410
    TERMINAL    DEPT411
    TERMINAL    DEPT412
    TERMINAL    DEPT510
    TERMINAL    DEPT511
    TERMINAL    DEPT512
    TERMINAL    DEPT100
    TERMINAL    DEPT200
    TERMINAL    DEPT686
    TERMINAL    MASTER
    TERMINAL    ALTMAST
    TERMINAL    MAINT
    TERMINAL    DEPT710
    TERMINAL    DEPT720
    TERMINAL    DEPT848
    TERMINAL    DEPT850
    TERMINAL    DEPT900
    TERMINAL    TEST1
    TERMINAL    TEST2

)( TERMINAL     MASTER
    TRANSACT    ACCTCHG
    TRANSACT    ACTY
    TRANSACT    TNL
    TRANSACT    INQUIRY
    TRANSACT    INQ
    TRANSACT    ENG
    TRANSACT    ACCT
    COMMAND     BROADCAST
    COMMAND     START
    COMMAND     STOP
    COMMAND     PSTOP
    COMMAND     PURGE
    COMMAND     CHANGE
    COMMAND     DELETE
    COMMAND     ASSIGN
    COMMAND     CHECKPOINT
    COMMAND     CHECKPOINT PURGE
    COMMAND     CHECKPOINT FREEZE
    COMMAND     DBDUMP
    COMMAND     NRESTART
    COMMAND     ERESTART
    COMMAND     DBRECOVERY
    COMMAND     DBLOG
    COMMAND     DBNOLOG
```

CHAPTER 6. STATISTICS AND ACCOUNTING

One of the basic components of the IMS/360 control program is the IMS/360 system log.

The information placed on the system log is used for many purposes, including statistics, accounting, restart, and data base recovery. All input messages received and all output messages sent are logged. All messages processed, the processing time, and the number and type of data base references made are recorded. This information is used to supply statistics about message volume by communication line and terminal. Error message counts as well as other data can be obtained. Accounting information about computer usage by application program can be derived.

An IMS/360 utility program is placed by IMS/360 system definition in IMS.RESLIB and may be used for analyzing the information on the IMS/360 system log tapes. The name of this program is DFSIST01.

IMS/360 SYSTEM LOG UTILITY PROGRAM

General Description

The IMS/360 control program includes a common service routine, the system recorder, designed to facilitate the placing of data on the system log. This information is used primarily for restart and offline statistical analysis (accounting etc.). The following information is written:

Data	When Written
1. For restart:	
a. Message queue control blocks	When they change
b. Checkpoint data	When checkpoint is taken
c. Record indicating an Operating System/360 data set open or close	When an IMS/360 data set used for message processing is opened or closed
d. Record indicating changes to a data base	When a data base insert, delete, or replace is made
2. For both restart and statistics:	
a. Message received from terminal	When a complete message is received or when disk block is full
b. Message sent to a terminal or another program	When a complete message is received or when disk block is full

3. For statistics only:

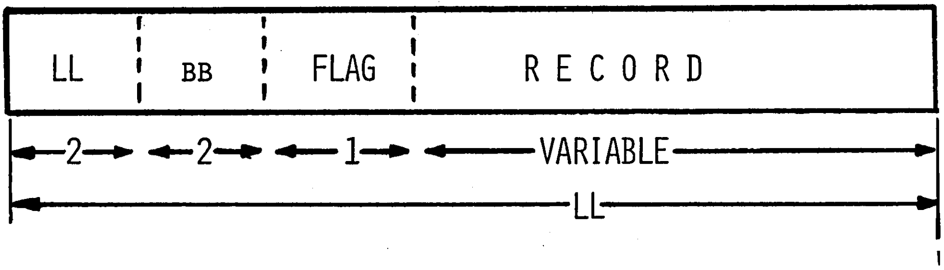
a. Error segments	When hardware error is detected receiving or sending to a terminal
b. Completion of send record	At completion of sending a message to a terminal
c. Application accounting record	When an application program terminates
d. IMS/360 accounting record	When system is started or stopped

Log Format

Records are written on the log using QSAM variable-length blocked records. Since different types of records are written on the log for different purposes, some method must be used to identify each logical record.

The first byte of each logical record is called the log flag and can be used to identify that logical record. The user can then look at the first byte of each logical record, process those records with which he is concerned, and bypass any record (first byte of log flag) with which he is not concerned.

Each logical record written on the log must be of the following format:



where LL is a halfword binary number representing the total length of the logical record, bb is a halfword used by OS, flag is a one-byte log flag, and the record is of variable length.

Each message received or sent carries control information in the form of the message prefix. In this prefix are message destination or source, date and time, and an input or output sequence number.

When the log routine receives a request to log a message, it first requests a prefix builder routine. On return from the prefix builder, the log routine logs the message. The majority of other log records are completely edited by the calling program; no processing is performed by the logging routine.

Log Data Set Allocation

The IMS0 procedure includes DD cards for old and new log data set allocations. The old log DD card name is IMSLOGR. The new log DD card name is IMSLOG.

Statistics Reports

Statistics reports provide a means of evaluating line and terminal loading, traffic volumes, response times, and accounting (billing) information. Samples of statistics reports are shown at the end of this chapter.

The flow of the system log utility program is shown in Figure 26.

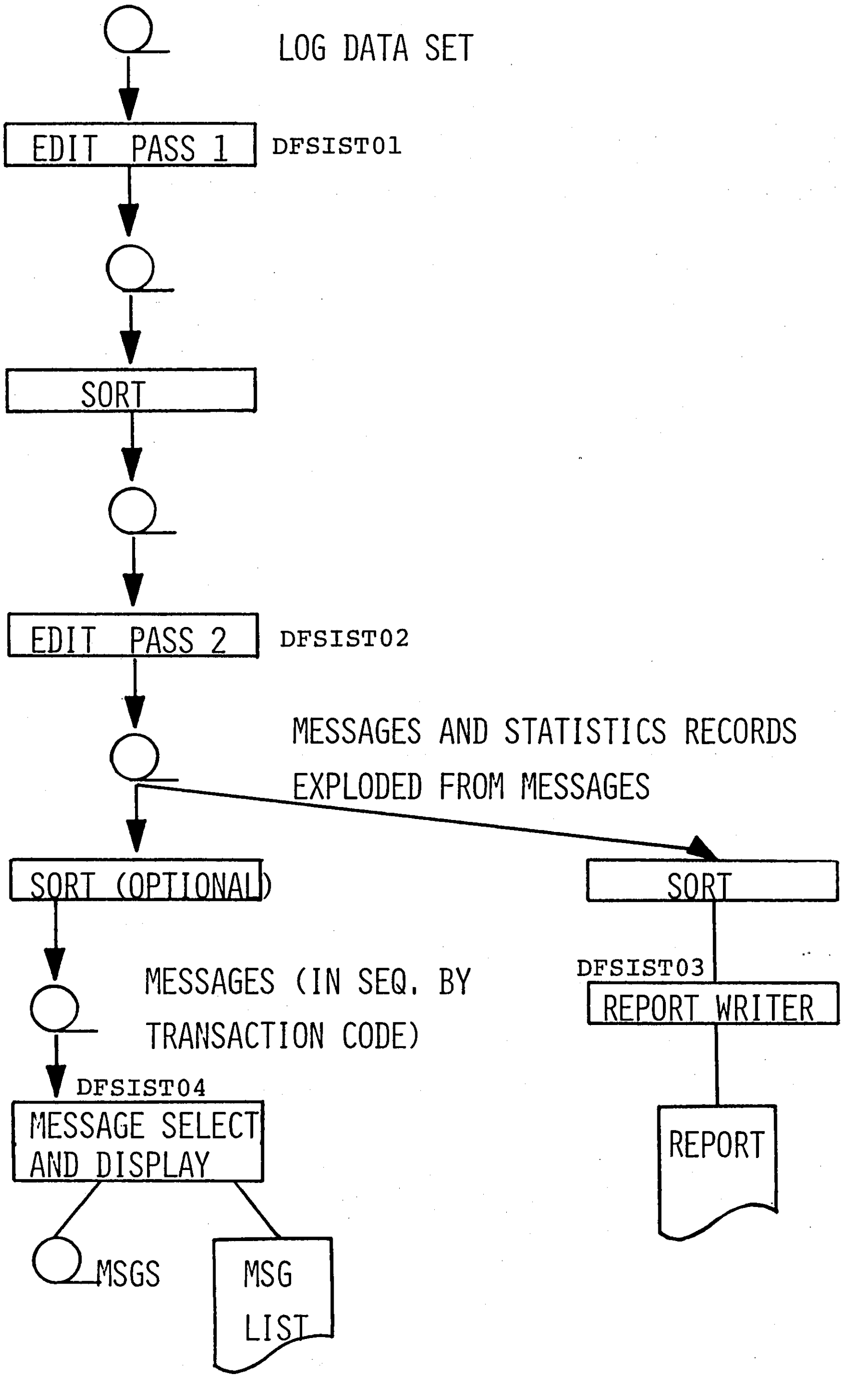


Figure 26. System log utility program flow

The functions of Edit Pass 1 are to select from the log those records used by statistics and to edit the prefix of the message so that, when sorted, computer input message and all outputs sent as a result of that input are contiguous.

The function of Edit Pass 2 is to explode from system messages the records to be used to produce statistics reports.

Types of Statistics Reports

The types of statistics reports are outlined below:

1. Messages Queued but Not Sent - by Terminal

    - Generated message appears on log, but no record appears to indicate the message was sent to the terminal

2. Line and Terminal Report

    - Shows line and terminal loading by time of day (could be used to determine line and terminal utilization, peak traffic periods, etc.)

3. Error Report

    - Same format as 2, above

    - Input is those segments on which hardware errors were detected

    - Could be used to pinpoint lines or terminals having excessive error routes

4. Messages Queued but Not Sent - by Terminal Code

    - Similar to 1, above, input

    - Sorted by transaction code rather than by terminal address

5. Transaction Report

    - Purpose: to show loading by transaction code and by time of day

    - Same format as 2, above

    - Input sorted by transaction code

6. Transaction Response Report

    - Measures time from complete receipt of input message until response to that message starts back to terminal

    - Percentile report shows shortest response, longest response, and 25th, 50th, 75th, and 95th percentile response.

7. Application Accounting Report

    - Purpose: to provide sufficient data to allow machine charges to be distributed back to terminal users

    - Following information contained in this report:

      Counts of all requests to Data Language/I

Amount of CPU task time

Task timer is set when request for scheduling is made. (Value is maximum time per transaction multipled by maximum number of transactions.) Remaining time is requested first prior to next request for scheduling. (This time is actual time program executed, not including any wait time for data accesses.)

All requests for services from Data Language/I, for access to either messages or data bases, are counted. These counts are accumulated by program, by transaction code within program, and by priority within transaction code.

Counts of messages processed and of "get uniques" are included (will be different because of "get unique" issued on which end-of-file is returned).

Average CPU time is total message CPU time divided by number of messages. Number of move calls reflects number of times block mover was requested to get the DBD and PSB blocks and move them to IMS/360 region.

Number of bad completion codes reflects number of times program terminated abnormally.

8. IMS/360 Accounting Report

- Shows amount of CPU time used by IMS/360 region. (This is task time, not including wait time.)
- Can be used in conjunction with Application Accounting Report to distribute IMS/360 time to users on the basis of services performed.

Operating Information

- Reports are produced either with or without date control.
- The program determines whether input was sorted on date.
- A control break occurs whenever the date changes; totals are printed, and a new report is started.
- If not sorted on date, should allow merging activity for a consecutive period (for example, one week) to produce one summary report.
- To sort by date, the sort control card is:

```
SORT FIELD=(5,1,CH,A,9,4,PD,A,13,36,CH,A)
```

- To sort disregarding date, the sort control card is:

```
SORT FIELD=(5,1,CH,A,13,36,CH,A),SIZE=XXXX
```

- The other control is a LINCNT=XX parameter included in the execute card. This is the only parameter expected and is optional. If not included, the default line count is 36.
- Printing of the different statistics reports is not optional; they are all generated.

Message Select and Copy or List

The execution of the message select and copy or list is optional; it may be executed as a separate step in the same job with the statistics reports or may be run independent of the statistics reports.

This utility takes output of the second edit program before it is sorted (when in line and terminal sequence), or after sorting (in transaction code sequence), and selects messages on the basis of control cards read from SYSIN. Messages selected are printed and/or copied onto an output data set. If a DD card named IMSLOGO is included, an output data set will be created. If a DD card named IMSLOGP is included, messages selected will be printed.

Control Cards

All control cards begin in column 1, with a keyword identifying that control card. Following the keyword is a series of parameters, enclosed within parentheses and separated by commas. Control cards cannot be continued beyond column 71. Multiple control cards with the same keyword starting in column 1 are permitted. Within parentheses, all parameters are positional; missing parameters must be indicated by commas.

A group of names may be indicated by terminating the parameter with an *. For example, INV* would cause name of INV, INVENTORY, INVA, or INVB to be selected.

The name parameter "all" may be used to select all names rather than a specific name.

Transaction Code Control Card

The format of the transaction code control card is:

```
TRANS CODE=(TRANSCOD,I,O),(TRANSA,I),(INV*,,O),(ALL,I,O)
```

- The first parameter is a transaction code of from one to eight bytes.
- The second is I to indicate that input messages with this code are to be selected.
- The third is O to indicate that output messages resulting from this code are to be selected.
- The transaction code of ALL indicates selection of all transaction codes.
- An asterisk within the transaction code causes only characters preceding the asterisk to be compared with the corresponding number of characters from the input transaction code to determine selection. This may also be used to select groups of transaction codes.

Symbolic Terminal Name Control Card

An example of the symbolic terminal name control card is:

```
SYM NAME=(TERMA,I,O),(TERM*,I),(TERMINV,,O,ALL)
SYM NAME=(TERMPAY,I,O,TERM)
```

- The first parameter is a symbolic terminal name of from one to eight bytes.

- The second and third parameters are I and O respectively, to select input from and output to this symbolic terminal.

- The O may be further qualified with another symbolic name to cause only output to that symbolic name which resulted from inputs from preceding name to be selected. If ALL is specified, all output resulting from the preceding name will be selected.

Hardware Terminal Address Control Card

The format of the hardware terminal address control card is:

```
TERM ADDR=(3,A,I,O),(42,C,,O,21,A), (I,ALL,I,O)
```

- Selection by hardware terminal name is similar to selection by terminal symbolic name, except that, instead of symbolic name, line number and terminal address are specified.

- The first parameter is the line number.

- The second parameter is the terminal address.

- The third and fourth parameters are I and O for selection of input to and output from this terminal.

- Output may be further qualified (similar to symbolic terminal output).

- ALL may be specified instead of terminal address or line number.

Time Control Card

The format of the time control card is:

```
TIME=(68014,1620,68015,1900)
```

- The first parameter is the starting date - year and day of year.

- The second parameter is the starting time - hours and minutes.

- The third parameter is the ending date.

- The fourth parameter is the ending time.

- If this card is included, only messages falling within the time slot are selected.

Nonprintable Character Control Card

The format of the nonprintable character control card is:

```
NON PRINT=HEX
```

- If this control card is included, nonprintable characters will be printed in hexadecimal on two lines, with one hexadecimal character above the other.

- By default, if this card is not included, nonprintable characters will appear as blanks.

System Log Utility Program JCL

The JCL for the execution of the IMS/360 system log utility program is shown in Figure 27.

```
//STATS JOB 848,NAME,MSGCLASS=I,MSGLEVEL=1,PRTY=8
//JOBLIB DD DSNAME=IMS.RESLIB,DISP=SHR
//STAT    EXEC  PGM=DFSIST01
//LOGDCB DD DSNAME=IMS.LOG,DISP=(OLD,DELETE)  INCLUDE VOL AND UNIT<-- NOTE 1
//EDITDCB1 DD   DSNAME=&&EDIT1,DISP=(NEW,PASS),UNIT=SYSDA,              X
//              SPACE=(CYL,(5,5)),DCB=(RECFM=VB,BLKSIZE=1404,LRECL=1400,X<-- NOTE 2
//              BUFNO=3)
//SYSOUT    DD    SYSOUT=I
//SORT    EXEC  SORTD,REGION=72K
//SYSOUT    DD    SYSOUT=A
//SORTIN DD     DSNAME=&&EDIT1,DISP=(OLD,DELETE)
//SORTOUT DD    DSNAME=&&EDIT1S,DISP=(NEW,PASS),UNIT=SYSDA.             X
//              SPACE=(CYL,(5,5)),DCB=(RECFM=VB,BLKSIZE=1404,LRECL=1400,X
//              BUFNO=3)
//SORTWK01 DD   UNIT=SYSDA,SPACE=(CYL,(05),,CONTIG)
//SORTWK02 DD   UNIT=SYSDA,SPACE=(CYL,(05),,CONTIG)
//SORTWK03 DD   UNIT=SYSDA,SPACE=(CYL,(05),,CONTIG)
//SORTWK04 DD   UNIT=SYSDA,SPACE=(CYL,(05),,CONTIG)
//SORTWK05 DD   UNIT=SYSDA,SPACE=(CYL,(05),,CONTIG)
//SORTWK06 DD   UNIT=SYSDA,SPACE=(CYL,(05),,CONTIG)
//SYSIN  DD     *
 SORT FIELDS=(5,1,CH,A,9,4,PD,A,13,24,CH,A),SIZE=E200
/*
//STAT02 EXEC   PGM=DFSIST02.
//EDITDCB1 DD   DSNAME=&&EDIT1S,DISP=(OLD,DELETE)
//FDITDCB2 DD   DSNAME=&&EDIT2,DISP=(NEW,PASS),UNIT=SYSDA.              X
//              SPACE=(CYL,(5,5)),DCB=(RECFM=VB,BLKSIZE=1404,LRECL=1400,X
//              BUFNO=3)
//SYSOUT   DD   SYSOUT=I
//SORT    EXEC  SORTD,REGION=72K
//SYSOUT    DD  SYSOUT=I
//SORTIN DD     DSNAME=&&EDIT2,DISP=(OLD,DELETE)
//SORTOUT   DD  DSNAME=IMS.EDIT,DISP=(NEW,KEEP),                        X
//              VOL=SER=222222,UNIT=2311,                               X
//              SPACE=(CYL,(1,1)),DCB=(RECFM=VB,BLKSIZE=1404,LRECL=1400)
//SORTWK01 DD   UNIT=SYSDA,SPACE=(CYL,(05),,CONTIG)
//SORTWK02 DD   UNIT=SYSDA,SPACE=(CYL,(05),,CONTIG)
//SORTWK03 DD   UNIT=SYSDA,SPACE=(CYL,(05),,CONTIG)
//SORTWK04 DD   UNIT=SYSDA,SPACE=(CYL,(05),,CONTIG)
//SORTWK05 DD   UNIT=SYSDA,SPACE=(CYL,(05),,CONTIG)
//SORTWK06 DD   UNIT=SYSDA,SPACE=(CYL,(05),,CONTIG)
//SYSIN  DD     *
 SORT FIELDS=(5,1,CH,A,9,4,PD,A,13,36,CH,A),SIZE=E200<---------------  NOTE 3
//RPTWRT EXEC   PGM=DFSIST03
//EDITDCB2 DD DSNAME=IMS.EDIT,DISP=(OLD,KEEP),UNIT=2311.VOL=SER=222222
//PRINTDCB DD   SYSOUT=I,DCB=(BLKSIZE=0133,LRECL=133,RECFM=FA)<-------  NOTE 4
//SIDEX1  EXEC PGM=DFSIS104 <-----------------------------------------  NOTE 5
//IMSLOGI DD DSNAME=IMS.EDIT,DISP=(OLD,DELETE),UNIT=2311,VOL=SER=222222
//IMSLOGP DD SYSOUT=I,DCB=(BLKSIZE=0133,LRECL=133,RECFM=FBA)<---------  NOTE 4
//SYSIN DD *
TRANS CODE=(ALL,I,O) <------------------------------------------------  NOTE 6
NON PRINT=HEX
```

Figure 27. JCL for the system log utility program

Notes:

1. Concatenate if necessary other volumes and units under DD cards if multiple data sets are to be processed.

2. BLKSIZE and LRECL may be changed here and in subsequent steps. LRECL must be at least as large as the largest buffers used for message queues.

3. Sort control card shown is for sorting by data and therefore producing reports under date control. To sort disregarding date and subsequently not control on date when producing reports, the sort control card is:

   ```
   SORT FIELDS = (5,1,CH,A,13,36,CH,A),SIZE=XXXX
   ```

4. Output may be blocked or unblocked; all I/O for statistics program is done using QSAM, with QSAM acquiring the buffers.

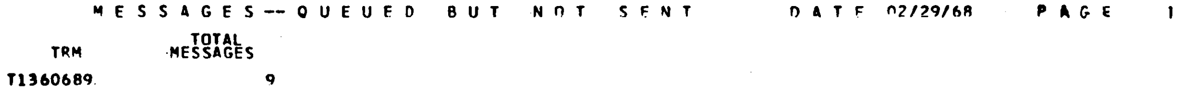
5. See preceding section, titled "Message Select and Copy or List", as this is a variable portion of the JCL where the user has different options.

6. See preceding section titled "Transaction Code Control Card", as this is a variable portion of the JCL.

STATISTICS REPORTS EXAMPLES

Following is a list of types of statistics reports available to the user of IMS/360. Examples follow on subsequent pages.

- Messages queued but not sent (by terminal)
- Line and terminal
- Error
- Messages queued but not sent (by transaction code)
- Transaction
- Transaction response
- Application accounting
- IMS accounting
- Messages



```
        MESSAGES--QUEUED BUT NOT SENT          DATE 02/29/68     PAGE     1
                 TOTAL
  TRM          MESSAGES
T1360689              9
```

```
         L I N E   A N D   T E R M I N A L   R E P C R T              D A T E  05/27/68                        P A G E    1

                    TOTAL          TOTAL        AVG                    HOURLY          DISTRIBUTION
LINE TRM   R/S    MESSAGES      CHARACTERS     SIZE  00-07 C7-08 08-09 09-10 10-11 11-12 12-13 13-14 14-15 15-16 16-17 17-18 18-19 19-24

 OC2  A

   *MASTER  S          67           2,229        33     0     0     0     0     0     0    31     7    14     9     6     0     0     0
            R          50           1,617        32     0     0     0     0     0     0    18     8    12    10     2     0     0     0

   *P057A   S           1              17        17     0     0     0     0     0     0     0     0     0     0     1     0     0     0

   *P682A   S           1              17        17     0     0     0     0     0     0     0     0     0     0     1     0     0     0

   *P682C   S           1              17        17     0     0     0     0     0     0     0     0     0     0     1     0     0     0

     TRM    S          70           2,280        32     0     0     0     0     0     0    31     7    14     9     9     0     0     0
    TOTALS  R          50           1,617        32     0     0     0     0     0     0    18     8    12    10     2     0     0     0

 003  A

   *2740A2  S          73           2,657        36     0     0     0     0     0     0    19    34     5    14     1     0     0     0
            R         104           2,487        23     0     0     0     0     0     0    27    39    16    22     0     0     0     0

 005  A

   *P663    S         190           7,522        39     0     0     0     0     0     0    38    64    58    30     0     0     0     0
            R         249           5,508        22     0     0     0     0     0     0    49    82    77    41     0     0     0     0

 009  A

   *P057A   S          46           1,785        38     0     0     0     0     0     0    13     7     7    19     0     0     0     0
            R          59           1,397        23     0     0     0     0     0     0    16    12    10    21     0     0     0     0



    SYSTEM  S         669          27,619        41     0     0     0     0     0     0   147   167   198   147    10     0     0     0
    TOTALS  R         846          23,323        27     0     0     0     0     0     0   168   230   261   185     2     0     0     0
```

```
                          ERROR REPORT                                       DATE 03/31/67                              PAGE  1

                            TOTAL      TOTAL        AVG                      HOURLY          DISTRIBUTION
LINE TRM      R/S        MESSAGES   CHARACTERS     SIZE  00-07 07-08 08-09 09-10 10-11 11-12 12-13 13-14 14-15 15-16 16-17 17-18 18-19 19-24

001  A0

   T15CAS1A  S               14          700        50     1     1     1     1     1     1     1     1     1     1     1     1     1     1
             R               14          700        50     1     1     1     1     1     1     1     1     1     1     1     1     1     1
002  A0

   T15CAS2A  S               14          700        50     1     1     1     1     1     1     1     1     1     1     1     1     1     1
             R               14          700        50     1     1     1     1     1     1     1     1     1     1     1     1     1     1

   T15CAS2B  S               14          700        50     1     1     1     1     1     1     1     1     1     1     1     1     1     1
             R               14          700        50     1     1     1     1     1     1     1     1     1     1     1     1     1     1

      TRM    S               28        1,400        50     2     2     2     2     2     2     2     2     2     2     2     2     2     2
    TOTALS   R               28        1,400        50     2     2     2     2     2     2     2     2     2     2     2     2     2     2
003  A0

   T15CAS3A  S               14          700        50     1     1     1     1     1     1     1     1     1     1     1     1     1     1
             R               14          700        50     1     1     1     1     1     1     1     1     1     1     1     1     1     1

   T15CAS3B  S               14          700        50     1     1     1     1     1     1     1     1     1     1     1     1     1     1
             R               14          700        50     1     1     1     1     1     1     1     1     1     1     1     1     1     1

   T15CAS3C  S               14          700        50     1     1     1     1     1     1     1     1     1     1     1     1     1     1
             R               14          700        50     1     1     1     1     1     1     1     1     1     1     1     1     1     1

      TRM    S               42        2,100        50     3     3     3     3     3     3     3     3     3     3     3     3     3     3
    TOTALS   R               42        2,100        50     3     3     3     3     3     3     3     3     3     3     3     3     3     3
     B0

   T15CAS3D  S               14          700        50     1     1     1     1     1     1     1     1     1     1     1     1     1     1
             R               14          700        50     1     1     1     1     1     1     1     1     1     1     1     1     1     1

   T15CAS3E  S               14          700        50     1     1     1     1     1     1     1     1     1     1     1     1     1     1
             R               14          700        50     1     1     1     1     1     1     1     1     1     1     1     1     1     1


   T15CAS3F  S               14          700        50     1     1     1     1     1     1     1     1     1     1     1     1     1     1
             R               14          700        50     1     1     1     1     1     1     1     1     1     1     1     1     1     1

      TRM    S               42        2,100        50     3     3     3     3     3     3     3     3     3     3     3     3     3     3
    TOTALS   R               42        2,100        50     3     3     3     3     3     3     3     3     3     3     3     3     3     3

     LINE    S               84        4,200        50     6     6     6     6     6     6     6     6     6     6     6     6     6     6
    TOTALS   R               84        4,200        50     6     6     6     6     6     6     6     6     6     6     6     6     6     6

    SYSTEM   S              126        6,300        50     9     9     9     9     9     9     9     9     9     9     9     9     9     9
    TOTALS   R              126        6,300        50     9     9     9     9     9     9     9     9     9     9     9     9     9     9
```

```
        MESSAGES--QUEUED BUT NOT SENT          DATE 05/31/67   PAGE    1
TRANSACTION      TOTAL
   CODE        MESSAGES

 T19QCA2A           9
 T19QCA2B           9
```

```
                    T R A N S A C T I O N   R E P O R T                    D A T E  05/27/68                                  P A G E   1
 TRANSACTICN        TOTAL         TOTAL        AVG                  HOURLY           DISTRIBUTION
    CODE    R/S   MESSAGES     CHARACTERS     SIZE  00-07 C7-08 08-09 09-10 10-11 11-12 12-13 13-14 14-15 15-16 16-17 17-18 18-19 19-24

 /CHECKPO    R           1             23       23     0     0     0     0     0     0     0     0     0     0     1     0     0     0

 /ENC.       R           1             10       10     0     0     0     0     0     0     0     1     0     0     0     0     0     0

 /START      S          54          1,396       25     0     0     0     0     0     0    35     5     8     6     0     0     0     0
             R          30            619       20     0     0     0     0     0     0    17     3     6     4     0     0     0     0

 /STOP       S           7            182       26     0     0     0     0     0     0     0     3     2     2     0     0     0     0
             R           4             72       18     0     0     0     0     0     0     0     2     1     1     0     0     0     0

 /TEST.      R           1             11       11     0     0     0     0     0     0     0     1     0     0     0     0     0     0

 PA          R           1              8        8     0     0     0     0     0     0     0     0     1     0     0     0     0     0

 PA /CZNC    R           1             15       15     0     0     0     0     0     0     0     1     0     0     0     0     0     0

 PA          S          23          1,121       48     0     0     0     0     0     0     4     7     3     9     0     0     0     0
             R          41          4,110      100     0     0     0     0     0     0    10    11     8    12     0     0     0     0

 PB          S           4            276       69     0     0     0     0     0     0     0     2     2     0     0     0     0     0
             R           7            919      131     0     0     0     0     0     0     0     4     3     0     0     0     0     0

 PC          S          88          2,156       24     0     0     0     0     0     0    21    32    17    18     0     0     0     0
             R          89          2,421       27     0     0     0     0     0     0    22    32    17    18     0     0     0     0

 PD          S           5            219       43     0     0     0     0     0     0     0     5     0     0     0     0     0     0
             R           5            116       23     0     0     0     0     0     0     0     5     0     0     0     0     0     0

 PI          S          11            395       35     0     0     0     0     0     0     2     2     4     3     0     0     0     0
             R         114          1,508       13     0     0     0     0     0     0    31    34    28    21     0     0     0     0

 PI123456    R           1             13       13     0     0     0     0     0     0     0     1     0     0     0     0     0     0

 PK          S          15            408       27     0     0     0     0     0     0     4     1     6     4     0     0     0     0
             R          16            336       21     0     0     0     0     0     0     4     1     6     5     0     0     0     0


    SYSTEM   S         669         27,619       41     0     0     0     0     0     0   147   167   198   147    10     0     0     0
    TOTALS   R         846         23,323       27     0     0     0     0     0     0   168   230   261   185     2     0     0     0
```

```
             TRANSACTION RESPONSE REPORT          DATE 05/27/68                       PAGE   1

    TYPE         TOTAL        LONGEST       95%        75%        50%        25%      SHORTEST
TRANSACTION    RESPONSES     RESPONSE    RESPONSE   RESPONSE   RESPONSE   RESPONSE   RESPONSE

   PA              23          51.6S       09.3S      01.9S      01.8S      01.8S      01.1S
   PB               4          01.9S       01.9S      01.8S      01.7S      01.4S      01.4S
   PC              68      12M 43.7S       02.0S      01.2S      01.1S      01.1S      01.0S
   PD               5          01.2S       01.2S      01.2S      01.1S      01.1S      01.1S
   PI              11          22.7S       22.7S      01.2S      01.2S      01.1S      01.1S
   PK              16          02.2S       02.2S      01.2S      01.1S      01.1S      01.1S

   PM               3          01.2S       01.2S      01.1S      01.1S      01.1S      01.1S
   PN               1          01.1S       01.1S      01.1S      01.1S      01.1S      01.1S
   PO               1          01.1S       01.1S      01.1S      01.1S      01.1S      01.1S
   PP               7          01.3S       01.3S      01.1S      01.1S      01.1S      01.1S
   PQ              33          02.0S       01.9S      01.7S      01.1S      01.1S      01.0S
   PR               3          48.4S       48.4S      01.6S      01.6S      01.6S      01.6S
   PS             271          24.7S       02.5S      01.2S      01.1S      01.1S      01.0S
   PV              39          02.9S       02.7S      01.2S      01.1S      01.1S      01.0S
   PW               7          01.4S       01.4S      01.2S      01.1S      01.1S      01.1S
   PX              13          02.7S       02.7S      01.8S      01.7S      01.7S      01.2S
```

```
          A P P L I C A T I O N   A C C O U N T I N G   R E P O R T        D A T E  05/27/68                       P A G E     1
PROGRAM TRANSACTION   MESSAGE- - - - COUNTS    DATA - - - - - - - - - - BASE - - - - - - - - COUNTS  MOVE    BAD          TOT MESS    AVR
  NAME      NAME     PRI QTY     GU     GN  ISRT    GU     GN    GNP    GHU    GHN   GHNP  ISRT  DLET  REPL  CALL    CC          CPU TIME   TIME

U818M001 PK          C2   16     32      0    18    16      0      0     14      0      0    14     0    14     0     0             00.4S  0.024S
         PS          C1  272    534      0   430     0      0      0    298      0      0   112     1   155     1     0             08.7S  0.030S
         PX          C1   15     30      0    25     0      0      0      0      0      0    37     0     0     0     0             00.4S  0.029S
         ******** **     303    596      0   473    16      0      0    312      0      0   163     1   169     1     0             09.6S  0.030S
U818M002 PA          C5   38     75    174   199     0      0      0     15      0      0    84     0     3     1     0             13.2S  0.333S
U818M003 PB          C4    7     14     63    84     0      0      0     57      0      0     0     0    48     0     0             01.0S  0.145S
         PC          C4   89    178      0   131     0      0      0     91      0      0    22     4    82     1     0             03.3S  0.036S
         ******** **      96    192     63   215     0      0      0    148      0      0    22     4   130     1     0             04.4S  0.044S
U818M004 PI          C1  114    227      0   303   111    103      0      0      0      0     0     0     0     0     0             03.0S  0.025S
         PN          C1    1      2      0     2     0      0      0      0      0      0     0     0     0     0     0             00.0S  0.048S
         PO          C1    1      2      0     2     0      0      0      0      0      0     0     0     0     0     0             00.0S  0.032S
         PP          C3   49     98      0   140   121    138      0      0      0      0     0     0     0     1     0             01.8S  0.035S
         PT          C1   51    101      0   537    51      0    407      0      0      0     0     0     0     0     0             07.8S  0.147S
         PV          C1   41     82      0    86    41      0      0      0      0      0     0     0     0     0     0             01.1S  0.026S
         ******** **     257    512      0  1070   324    241    407      0      0      0     0     0     0     1     0             13.8S  0.051S
U818M005 PM          C2    3      6      0     6     0      0      0      3      0      0     0     0     0     0     0             00.0S  0.014S
         PC          C1   33     65      0    58     0      0      0     32      0      0    11     0    16     1     0             00.9S  0.027S
         ******** **      36     72      0    64     0      0      0     35      0      0    11     0    16     1     0             00.9S  0.026S
U818M006 PC          C1    5     10      0     8     0      0      0      2      0      0     4     0     2     0     0             00.1S  0.032S

         PR          C1    3      4      0     2     0      0      0      0      0      0     0     0     0     0     0             10.0S  3.216S
         PW          C1    7     14      0    13     2      0      0      0      0      0     2     0     0     1     0             00.2S  0.028S
         ******** **      15     28      0    23     2      0      0      2      0      0     6     0     2     1     0             10.4S  0.667S
SYSTEM TOTALS            745   1475    237  2044   342    241    407    512      0      0   286     5   320     6     0             52.5S  0.067S
* INDICATES TOTAL SHOWN IN 100'S
@ INDICATES TOTAL SHOWN IN 10,000'S
```

```
     I M S   ACCOUNTING   REPORT                     D A T E  05/27/68    P A G E    I
-----------------------------------------------------------------------------------

I M S  CPU TIME FOR DAY 05/27/68  IS      01M 11.4S OR       71.4S
I M S  CPU TOTAL TIME             IS      01M 11.4S OR       71.4S
```

```
                   M E S S A G E S
INPUT   SEG=001 LEN=117 *A1 RF=01 CF=0C TYPE=1050 LINE=001 TERM=A0 SEQ=001 NAME=TEST 01 DATE=67.240 TIME=01.01.01CS-IN,  -OUT*
                        *,STAT,00100,00100*
OUTPUT SEG=001 LEN=117 *A1 RF=03 CF=0C TYPE=1050 LINE=001 TERM=A0 SEQ=001 NAME=TEST 01 DATE=67.240 TIME=01.01.01  -IN,CS-OUT*
                        *,STAT,00200,00200*
INPUT   TRANSACTION   LINE   TERM   SEQ   SYMBOLIC                          OUTPUT   LINE   TERM   SEQ   SYYBOLIC
REFIX      CODE        NO    ADDR   NO    ADDRESS    DATE      TIME         PREFIX    NO    ADDR   NO    ADDRESS    DATE      TIME

        A1             001    A0   00001  TEST 01   67.240  01.01.01                  001    A0   81984  TEST 01   67.241  01.00.01

INPUT   SEG=001 LEN=117 */CANCEL  CF=0C TYPE=2740 LINE=002 TERM=B0 SEQ=002 NAME=TEST 03 DATE=67.241 TIME=01.01.41CANCEL 00705*
                        *,     ,00706,00706*
INPUT   TRANSACTION   LINE   TERM   SEQ   SYMBOLIC
PREFIX     CODE        NO    ADDR   NO    ADDRESS    DATE      TIME

        /CANCEL        002    B0   00002  TEST 03   67.241  01.01.41

INPUT   SEG=001 LEN=117 *B1 RF=01 CF=08 TYPE=2740 LINE=002 TERM=B0 SEQ=001 NAME=TEST 02 DATE=67.240 TIME=01.01.21FS-IN,  -OUT*
                        *,NOST,00300,00300*
INPUT   SEG=002 LEN=117 *B1 RF=01 CF=00 TYPE=2740 LINE=002 TERM=B0 SEQ=001 NAME=TEST 02 DATE=67.240 TIME=01.01.21IS-IN,  -OUT*
                        *,NOST,00400,00400*
INPUT   SEG=003 LEN=117 *B1 RF=01 CF=04 TYPE=2740 LINE=002 TERM=B0 SEQ=001 NAME=TEST 02 DATE=67.241 TIME=01.01.21LS-IN,DAY-CH*
                        *,NOST,00500,00500*
OUTPUT SEG=001 LEN=117 *B1 RF=03 CF=08 TYPE=2740 LINE=002 TERM=B0 SEQ=001 NAME=TEST 02 DATE=67.241 TIME=01.01.21  -IN,FS-OUT*
                        *,NOST,00600,00600*
OUTPUT SEG=002 LEN=117 *B1 RF=03 CF=04 TYPE=2740 LINE=002 TERM=B0 SEQ=001 NAME=TEST 02 DATE=67.241 TIME=01.01.21  -IN,LS-OUT*
                        *,NOST,00700,00700*
INPUT   TRANSACTION   LINE   TERM   SEQ   SYMBOLIC                          OUTPUT   LINE   TERM   SEQ   SYYBOLIC
REFIX      CODE        NO    ADDR   NO    ADDRESS    DATE      TIME         PREFIX    NO    ADDR   NO    ADDRESS    DATE      TIME

        B1             002    B0   00001  TEST 02   67.241  01.01.21                  THIS OUTPUT MESSAGE WAS NOT SENT

INPUT   SEG=001 LEN=117 *C1 RF=01 CF=0C TYPE=1030 LINE=003 TERM=C0 SEQ=001 NAME=TEST 04 DATE=67.241 TIME=01.01.41INPUT ONLY  *
                        *     ,00800,00800*
INPUT   TRANSACTION   LINE   TERM   SEQ   SYMBOLIC
PREFIX     CODE        NO    ADDR   NO    ADDRESS    DATE      TIME

        C1             003    C0   00001  TEST 04   67.241  01.01.41

OUTPUT SEG=001 LEN=124 *C2 RF=03 CF=0C TYPE=1050 LINE=003 TERM=D0 SEQ=001 NAME=TEST 05 DATE=67.241 TIME=01.01.51OUTPUT ONLY *
                        *     ,00900,0090NEXT LIN*
```

PROGRAM SPECIFICATION BLOCK GENERATION - PSBGEN ERROR CONDITIONS

Erroneous Control Card	Error Message
PCB	---PCB010---PCB type parameter missing or invalid
PCB	---PCB020---PCB LTERM parameter not specified for TP PCB
PCB	---PCB030---DBDNAME parameter not specified for DB PCB
PCB	---PCB040---KEYLEN parameter not specified for DB PCB
PCB	---PCB050---PROCOPT parameter not specified for DB PCB
PCB	---PCB060---DBDNAME specified for TP PCB
PCB	---PCB070---PROCOPT specified for TP PCB
PCB	---PCB080---KEYLEN operand for TP PCB
PCB	---PCB090---LTERM operand specified for DB PCB
PCB	---PCB100---Invalid processing option in PCB
PCB	---PCB110---TP PCB must occur before any DB PCB's
SENSEG	---SEG010---Segment name parameter invalid
SENSEG	---SEG020---Too many SENSEG cards; 255 maximum
SENSEG	---SEG030---SENSEG invalid for TP PCB's
SENSEG	---SEG040---Parent name parameter invalid
SENSEG	---SEG050---Parent segment not predefined
SENSEG	---SEG060---Parent name parameter omitted or invalid
SENSEG	---SEG070---Duplicate segment name

PSBGEN	---PSB010---PCB in error, generation terminated
PSBGEN	---PSB020---PSBNAME not specified
PSBGEN	---PSB030---Invalid language operand
PSBGEN	---PSB040---No sensitive segments for DB PCB
PSBGEN	---PSB050---PSB name must begin with alpha character
PSBGEN	---PSB099---System error, generation terminated

DATA BASE DESCRIPTION GENERATION - DBDGEN ERROR CONDITIONS

Erroneous Control Card	Error Messages
DBD	---DBD010---Incorrect or missing access method
DBD	---DBD020---DBD name parameter not specified
DBD	---DBD030---Too many DBD cards
DBD	---DBD040---DBD name must begin with alphabetic characters
DMAN	---DMAN010---Incorrect device specification
DMAN	---DMAN020---Incorrect access specification
DMAN	---DMAN030---DD2 parameter invalid with ACCESS equal to ISAM
DMAN	---DMAN040---Too many DMAN cards
DMAN	---DMAN050---BLKFACT specified but no LRECL
DMAN	---DMAN060---LRECL specified but no BLKFACT operand
DMAN	---DMAN070---LRECL BLKFACT greater than track length
DMAN	---DMAN080---Missing DLIOF operand with access equal to ISAM
DMAN	---DMAN090---DLIOF is present or DD2 is missing with access equal to SAM
DMAN	---DMAN100---DD1 operand omitted
DMAN	---DMAN110---DD1 and DD2 have same DD names for HSAM

DMAN	---DMAN120---DD1/DLIOF duplicate DD names for HISAM
SEGM	---SEGM10---Segment name not specified
SEGM	---SEGM20---Segment bytes parameter not specified
SEGM	---SEGM30---Segment frequency parameter not specified
SEGM	---SEGM40---Root segment parent must equal zero
SEGM	---SEGM50---Parent operand not specified for dependent segment
SEGM	---SEGM60---Too many SEGM cards; 255 maximum
SEGM	---SEGM70---Segment length greater than DASD track
SEGM	---SEGM80---Segment length specified as zero
SEGM	---SEGM90---Segment frequency of zero invalid
SEGM	---SEGM100---Duplicate segment names
SEGM	---SEGM110---Segment length greater than specified LRECL
FLD	---FLD010---Field name parameter not specified or invalid (that is, more than 8 characters)
FLD	---FLD040---Type parameter not specified or invalid
FLD	---FLD050---FLDK card not first after SEGM card
FLD	---FLD060---Too many FLD or FLDK cards specified
FLD	---FLD070---Field length extends beyond segment end
FLD	---FLD080---First byte of segment is 1
FLD	---FLD100---Duplicate field name in segment
FLD	---FLD110---Bytes parameter invalid (that is, a nonnumeric field, 0 or less, or greater than 256)
FLD	---FLD120---Start parameter is invalid. (1 - if the size of the field is greater than the size of the segment that it is in 2 - size of the start parameter is a nonnumeric field)

FLD	---FLD130---Specified fields in segment exceed 255
FLDK	---FLDK010---Key field specified inappropriately
DBDGEN	---DGEN010---Segment X parent Y not found
DBDGEN	---DGEN020---Invalid number of DMAN cards for access method specified
DBDGEN	---DGEN030---DAM not supported
DBDGEN	---DGEN040---No segments for DMAN X
DBDGEN	---DGEN050---DAM not supported
DBDGEN	---DGEN060---Errors in this DBD
DBDGEN	---DGEN070---Too many levels in data base segment hierarchy
DBDGEN	---DGEN080---First segment in secondary data set group lower than level two
FINISH	---FINI10---No successful DBD's in this run

SYSTEM DEFINITION ERROR CONDITIONS

Stage 1 Error Messages:

G000 IMSCTRL MUST BE 1ST MACRO, IMSGEN MUST BE LAST

**** ENTERED DDNAME IS A DUPLICATE - name

**** ENTERED DDNAME IS RESERVED - name

The following ddnames are reserved ddnames: IMSLOG, IMS, IMSCSP, IMSLOGR, SYSUDUMP, SYSABEND, JOBLIB, STEPLIB. Note that this list may be modified by OS/360 system changes.

<u>IMSCTRL</u>

G001 MORE THAN ONE IMSCTRL MACRO SPECIFIED

G002 SYSTEM OPERAND OMITTED OR INVALID

The generation type must be BATCH or ALL.

G003 MAXREGN OPERAND OMITTED OR INVALID

Range is 1 - 255.

G004 MAXIO OPERAND INVALID

Range is 1 - 255.

G005 MSGBUFF OPERAND OMITTED OR INVALID

Range is 1 - 255.

G006 OCENDA OPERAND OMITTED OR INVALID

Value range is WA to Z9.

G007 CKPT LOG FREQ OPERAND IS INVALID

Range is 500 - 36863.

G008 ONE OR MORE OF THE SVC OPERANDS ARE INVALID ALL SVC OPERANDS MUST BE MUTUALLY EXCLUSIVE SVC OPERAND RANGE IS 128 - 255

APPLCTN

G101 APPLCTN SPECIFICATION IS NOT COMPATIBLE WITH GENTYPE SPECIFICATION (BATCH) IN IMSCTRL MACRO

G102 PSB OPERAND OMITTED OR INVALID

Cannot exceed 8 characters.

G103 PGMTYPE OPERAND OMITTED OR INVALID

G104 APPLCTN SPECIFICATION LIMIT EXCEEDED

No more than 255 applications can be specified.

G105 PSB - name - PREVIOUSLY SPECIFIED

G106 PSB OPERAND MUST BEGIN WITH ALPHA - name

DATABASE

G201 DATABASE SPECIFICATION IS NOT COMPATIBLE WITH GENTYPE SPECIFICATION (BATCH) IN IMSCTRL

G202 DATABASE SPECIFICATION NOT IN APPLCTN GROUP

Data base specification must be preceded by an application specification.

G203 DBD OPERAND OMITTED OR INVALID

G204 INTENT OPERAND OMITTED OR INVALID

G205 DBD OPERAND MUST BEGIN WITH ALPHA - name

G206 DATABASE SPECIFICATION LIMIT EXCEEDED

Maximum number of data bases is 255.

G207 LOG OPERAND IS INVALID - name

TRANSACT

G301 TRANSACT SPECIFICATION IS NOT COMPATIBLE WITH GENTYPE SPECIFICATION (BATCH) IN IMSCTRL MACRO

G302 TRANSACT SPECIFICATION NOT IN APPLCTN GROUP

Transact must be preceded by an application.

G303 CODE OPERAND OMITTED OR INVALID

Cannot exceed 8 characters.

G304 PRTY OPERAND OMITTED OR INVALID

G305 PROCLIM OPERAND OMITTED OR INVALID

G306 MSGTYPE OPERAND OMITTED OR INVALID

G307 TRANSACT SPECIFICATION LIMIT EXCEEDED

Maximum is 255.

G308 CODE OPERAND MUST BEGIN WITH ALPHA - name

G309 TRANSACT CODE - name - PREVIOUSLY SPECIFIED

G310 INQUIRY OPERAND IS INVALID - code

G311 TRANSACTION CODE - code - DEFINED AS AN LTERM warning message

G312 PRIORITY VALUES FOR TRANSACTION CODES USED BY BATCH PROGRAMS MUST BE NULL; VALUES ARE RESET TO PRTY= (0,0, limit count)

LINEGRP

G401 LINEGRP SPECIFICATION IS NOT COMPATIBLE WITH GENTYPE SPECIFICATION (BATCH) IN IMSCTRL MACRO

G402 UNITYPE OPERAND OMITTED OR INVALID

Must be one of the IMS/360-supported devices (1050, 2260, or 2740).

G403 DDNAME OPERAND OMITTED OR INVALID

Maximum of 8 characters.

G404 LINEGRP SPECIFICATION LIMIT EXCEEDED

Maximum number is 255.

G405 LINEGRP DDNAME - name - PREVIOUSLY SPECIFIED

G406 DDNAME OPERAND MUST BEGIN WITH ALPHA - name

G407 FEAT OPERAND OMITTED OR INVALID

LINE

G501 LINE SPECIFICATION IS NOT COMPATIBLE WITH GENTYPE SPECIFICATION (BATCH) IN IMSCTRL MACRO

G502 LINE SPECIFICATION CANNOT OCCUR BEFORE LINEGRP

G503 ADDR OPERAND OMITTED OR INVALID

G504 LINEGRP FEAT SPECIFICATION -- feat
IS NOT COMPATIBLE WITH LINE SPECIFICATION -- feat

G505 FEAT OPERAND OMITTED OR INVALID

G506 LINE SPECIFICATION LIMIT EXCEEDED

Maximum number is 255.

G507 DIAL ZONE CODE LIMIT EXCEEDED

Range limits 0 - 15.

SUBPOOL

G610 SUBPOOL SPECIFICATION IS NOT COMPATIBLE WITH GENTYPE SPECIFICATION (BATCH) IN IMSCTRL MACRO

G611 SUBPOOL SPECIFICATION CANNOT PRECEDE POOL

G612 TERMINAL/SUBPOOL SPECIFICATION LIMIT EXCEEDED

G613 TELNO OPERAND OMITTED OR INVALID

Number cannot exceed 16 digits.

TERMINAL

G601 TERMINAL SPECIFICATION IS NOT COMPATIBLE WITH GENTYPE SPECIFICATION (BATCH) IN IMSCTRL MACRO

G602 ADDR OPERAND OMITTED OR INVALID

G603 TERMINAL SPECIFICATION CANNOT PRECEDE LINE MACRO

G604 TERMINAL SPECIFICATION LIMIT EXCEEDED

Maximum number is 255.

G605 UNIT OPERAND OMITTED OR INVALID

Unit operand is mandatory for 2260 line groups.

G606 UNIT OPERAND SEQUENCE ERROR

The 2848 unit addresses must appear in ascending sequence within a line.

POOL

G510 POOL SPECIFICATION IS NOT COMPATIBLE WITH GENTYPE SPECIFICATION (BATCH) IN IMSCTRL MACRO

G511 POOL SPECIFICATION CANNOT OCCUR BEFORE LINEGRP

G512 POOL SPECIFICATION INVALID FOR NONSWITCH LINEGRP

G513 FEAT OPERAND INVALID - feat

G514 LINE/POOL SPECIFICATION LIMIT EXCEEDED

G515 DIAL ZONE CODE LIMIT EXCEEDED

NAME

G701 NAME SPECIFICATION IS NOT COMPATIBLE WITH GENTYPE SPECIFICATION (BATCH) IN IMSCTRL MACRO

G702 NAME SPECIFICATION MUST FOLLOW TERMINAL/SUBPOOL

G703 LTERM NAME OPERAND OMITTED OR INVALID

Maximum of 8 characters.

G704 LTERM - name - PREVIOUSLY SPECIFIED

G705 NAME SPECIFICATION LIMIT EXCEEDED

G706 NAME OPERAND MUST BEGIN WITH ALPHA - name

G707 WTOR PREDEFINED NAME - RESERVED FOR SYSTEM USE

G708 COMPT OPERAND IS INVALID - compt

Component value must be 0, 1, 2, or 3 and is mandatory for 1050 line groups.

G709 LTERM - name - DEFINED AS A TRANSACTION CODE

MASTTERM

G801 MASTTERM SPECIFICATION IS NOT COMPATIBLE WITH GENTYPE SPECIFICATION (BATCH) IN IMSCTRL MACRO

G802 MULTIPLE MASTER TERMINAL SPECIFICATIONS

G803 MASTER TERMINAL NAME OMITTED OR INVALID

Maximum of 8 characters.

G804 NAME OPERAND MUST BEGIN WITH ALPHA - name

G805 MASTER TERMINAL CANNOT BE ON A SWITCHED LINE LINEGRP - n LINE - N TERMINAL - n NAME - n

G806 MASTER TERMINAL NAME NOT DEFINED

Master terminal name must be defined on a previously encountered NAME macro.

G807 THE NAME SELECTED FOR MASTER MUST APPEAR AS THE 1ST NAME FOR TERMINAL - x of LINE - n IN LINEGRP - n

G808 MASTER TERMINAL MUST BE FIRST TERMINAL ON LINE X

MSGQUEUE

G801 MSGQUEUE SPECIFICATION IS NOT COMPATIBLE WITH GENTYPE SPECIFICATION (BATCH) IN IMSCTRL MACRO

G802 MULTIPLE MSGQUEUE SPECIFICATIONS

G803 QCRIN OPERAND OMITTED OR INVALID

G804 MSGIN OPERAND OMITTED OR INVALID

G805 QCROUT OPERAND SPECIFICATION IS INVALID

G806 MSGOUT OPERAND SPECIFICATION IS INVALID

G807 QCRIN DDNAME SUBFIELD IS INVALID

Must be 1 - 8 characters.

G808 QCRIN DDNAME OPERAND MUST BE ALPHA - name

First character must be alpha.

G809 QCRIN DSNAME SUBFIELD IS INVALID

Must be 1 - 16 characters.

G810 QCRIN DSNAME OPERAND MUST BE ALPHA - name

First character must be alpha.

G811 QCRIN UNIT SUBFIELD IS INVALID

Must be 2311, 2314, 2301, 2303.

G812 QCRIN SERIAL SUBFIELD IS INVALID

Must be 1 - 6 characters.

G813 through G818 same as G807 - G812 except for MSGIN operands.

G819 through G824 same as G807 - G812 except for QCROUT operands.

G825 through G830 same as G807 - G812 except for MSGOUT operands.

G831 REUSE OPERAND OMITTED OR INVALID - value

RESLIB, MACLIB

G901 NO MORE THAN ONE RESLIB CAN BE SPECIFIED

G902 PDS OPERAND OMITTED OR INVALID

G903 VOLNO OPERAND OMITTED OR INVALID

G904 UNIT OPERAND OMITTED OR INVALID

G905 PDS OPERAND MUST BEGIN WITH ALPHA - name

PSBLIB, DBDLIB, PROCLIB, PGMLIB

G901 - G905 SAME AS FOR RESLIB AND MACLIB

G906 IF VOLNO OR UNIT IS ENTERED BOTH MUST BE ENTERED

IMSGEN

G030 NO APPLCTN SPECIFICATIONS

G031 NO LINEGRP SPECIFICATIONS

G032 NO LINE SPECIFICATIONS FOR LINEGRP - n

G033 NO TERMINALS ON LINE - n

G034 MASTER TERMINAL CANNOT BE ON A SWITCHED LINE

G035 NO MASTER TERMINAL SPECIFICATION

G036 TERMINAL - n HAS NO LOGICAL NAME SPECIFICATION

G037 MSGQUEUE DATA SETS NOT SPECIFIED

G038 RESLIB NOT SPECIFIED

G039 UTISDS OPERAND OF IMSGEN OMITTED OR INVALID

Must be 1 - 16 characters.

G040 UNSUCCESSFUL IMS/360 SYSTEM DEFINITION

Occurs whenever any error occurs during definition. Definition is terminated.

IMSGEN WARNING MESSAGES

**** DDNAME CHECK TABLE FULL

G041 DLI PROCEDURE IS NOT INCLUDED; REQUIRED LIBRARIES ARE NOT SPECIFIED

G042 IMSCOBOL AND IMSPLI PROCEDURES ARE NOT INCLUDED; REQUIRED LIBRARIES ARE NOT SPECIFIED

G043 IMSCOBGO AND DLIPLIGO PROCS ARE NOT INCLUDED; REQUIRED LIBRARIES ARE NOT SPECIFIED

G044 PSBGEN PROCEDURE IS NOT INCLUDED; REQUIRED LIBRARIES ARE NOT SPECIFIED

G045 DBDGEN PROCEDURE IS NOT INCLUDED; REQUIRED LIBRARIES ARE NOT SPECIFIED

G046 IMS ONLINE PROCEDURES ARE NOT INCLUDED; REQUIRED LIBRARIES ARE NOT SPECIFIED

G047 NO DATABASE SPECIFICATIONS FOR PSB - name

G048 NO TRANSACT SPECIFICATIONS FOR PSB - name

G049 THE TERMINAL SELECTED FOR MASTER SHOULD HAVE MORE THAN ONE LOGICAL NAME ASSIGNED TO IT

IMSTEST

G090 IMSTEST MACRO MUST PRECEDE IMSCTRL MACRO

G091 CODE OPERAND OMITTED

G092 CODE OPERAND MUST BE ONLY ONE CHARACTER

G093 CODE OPERAND IS INVALID

G094 ONE OR MORE OF THE SPECIFIED LIBRARIES ARE OMITTED OR INVALID

DATA LANGUAGE/I STATUS CODES

	DATA BASE CALLS						MSG CALLS						
STATUS CODE	GU GHU	GN GHN	GNP GHNP	DLET REPL	ISRT (LOAD)	ISRT (ADD)	GU	GN	ISRT	CALL COMPLETED	ERROR IN CALL	I/O OR SYST.ERROR	DESCRIPTION
AB	X	X	X	X	X	X	X	X	X		X		SEGMENT I/O AREA REQUIRED, NONE SPECIFIED IN CALL
AC	X	X	X		X	X					X		HIERARCHICAL ERROR IN SSA'S
AD											X		INVALID FUNCTION PARAMETER
AE			X								X		ROOT SEGMENT SPECIFIED BY THIS CALL, NOT ALLOWED GNP CALLS
AF				X							X		DLET OR REPL CALLS CANNOT HAVE SSA'S SPECIFIED
AG	X				X	X					X		FIRST SSA SPECIFIED IS NOT LEVEL 1
AH	X				X	X					X		CALL REQUIRES SSA'S. NONE PROVIDED
AI	X	X	X	X	X	X						X	DATA MANAGEMENT OPEN ERROR
AJ	X	X	X		X	X					X		INVALID SSA QUALIFICATION FORMAT
AK	X	X	X		X	X					X		INVALID FIELD NAME IN CALL
AL	X	X	X	X	X	X					X		CALL USING TERM PCB IN TYPE 3 (BATCH)
AM	X	X	X	X	X	X					X		CALL FUNCTION NOT COMPATIBLE W/ PROCESSING OPTION
AN		X	X								X		GN CALL FOLLOWING ISRT CALL IS INVALID
AO	X	X	X	X	X	X						X	I/O ERROR ISAM OR BSAM
AP	X	X	X	X	X	X			X			X	I/O ERROR OSAM
AQ							X	X				X	READ I/O ERROR, MESSAGE CHAIN CANNOT BE FOLLOWED, MINIMUM OF ONE MESSAGE LOST
AR							X	X				X	READ I/O ERROR, MESSAGE SEGMENT HAS BEEN LOST. MESSAGE CHAIN IS STILL INTACT.
AS							X	X				X	QUEUES NOT AVAILABLE
AT									X		X		TRANSACTION CODE DOES NOT MATCH PCB NAME IN PGM-TO-PGM MSG SWITCH
DA				X							X		SEGMENT KEY FIELD HAS BEEN CHANGED
DJ				X							X		NO PRECEDING SUCCESSFUL GET HOLD CALL
GA		X	X							X			CROSSED HIERARCHICAL BOUNDARY INTO HIGHER LEVEL * (RETURNED ON UNQUALIFIED CALLS ONLY)
GB		X											END OF DATA SET. LAST SEGMENT REACHED.
GE	X	X	X			X							SEGMENT NOT FOUND
GK		X	X							X			DIFFERENT SEGMENT TYPE AT SAME LEVEL RETURNED (RETURNED ON UNQUALIFIED CALLS ONLY)
GP			X								X		A GNP CALL AND NO PARENT, OR REQUESTED SEGMENT LEVEL NOT LOWER THAN PARENT LEVEL
II						X							SEGMENT TO INSERT ALREADY EXISTS IN DATA BASE
LB					X								SEGMENT TO INSERT ALREADY EXISTS IN DATA BASE
LC					X								KEY FIELD OF SEGMENTS OUT OF SEQUENCE
LD					X								NO PARENT FOR THIS SEGMENT HAS BEEN LOADED
LE					X								SEQUENCE OF SIBLING SEGMENTS NOT THE SAME AS DBD SEQUENCE
QC							X						NO MORE INPUT MESSAGES
QD								X					NO MORE SEGMENTS FOR THIS MESSAGE
QE								X			X		GET NEXT REQUEST BEFORE GET UNIQUE
QF									X		X		SEGMENT LESS THAN FIVE CHARACTERS (SEG LENGTH IS MSG TEXT LENGTH PLUS FOUR CONTROL CHARACTERS)
QH									X		X		TERMINAL SYMBOLIC ERROR - OUTPUT DESIGNATION UNKNOWN TO IMS/360 (LOGICAL TERMINALS OR TRANSACTION CODE)
QI								X			X		GET NEXT AFTER END OF MESSAGE
BB MEANING BLANK BLANK	X	X	X	X	X	X	X	X	X	X			GOOD! NO STATUS CODE RETURNED. PROCEED!

* SEE PARAGRAPH ON CROSS-HIERARCHICAL BOUNDARY DEFINITION IN IMS/360 PDM

CHAPTER 8. IMS/360 SAMPLE PROBLEM

The IMS/360 basic distribution tape contains two data sets, IMS.GENLIB and IMS.LOAD. These data sets are unloaded versions of direct access partitioned data set libraries as produced by the Operating System/360 utility program IEHMOVE. Contained in these libraries are the program modules and macro-definitions which comprise the sample application.

A series of steps is involved in the creation of the sample application environment. Detailed background information regarding these steps is available from the references shown below:

• Copying IMS/360 distribution libraries to direct access storage devices	SOM Chapter 2 MOM Chapter 4
• Performing an IMS/360 system definition	SOM Chapter 4 MOM Chapter 4
• Performing a data base description (DBDGEN)	PDM Chapter 7 SOM Chapter 3
• Performing a program specification block generation (PSBGEN)	PDM Chapter 7 SOM Chapter 3
• Moving sample problem programs and control blocks	SOM Chapter 8
• Executing an IMS/360 data base load in the batch environment	MOM Chapter 4
• Initializing IMS/360 in an online environment. Executing the online application program from user terminals.	MOM Chapter 4

Before proceeding with the instructions for setting up the sample application, a description of the application and its data bases is appropriate.

DESCRIPTION OF SAMPLE PROBLEM

The application included within the sample problem is taken from the manufacturing industry. This application in its full sense includes the creation, use, and maintenance of the logical data bases associated with the product data systems. This product data can be contained in three subject data bases. The product data is either related to engineering drawings, part numbers, or systems equipment structure. These are three logical data bases, each organized under one of the above subjects.

To facilitate the implementation of these three logical data bases, they have been split into three data bases, comprising five data set groups (see Figure 28).

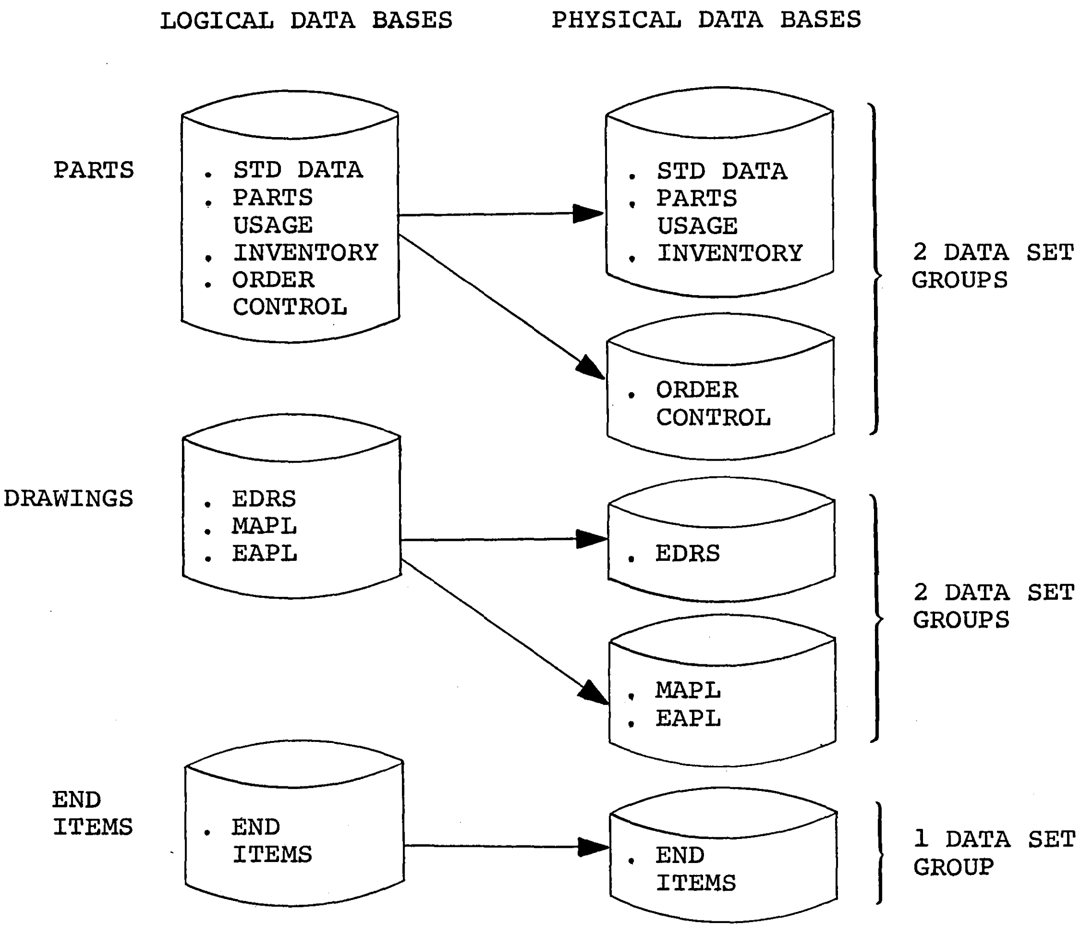


Figure 28. Logical and physical data bases

The five physical data bases and the segments contained within these data bases are described in Figures 29, 30, and 31.

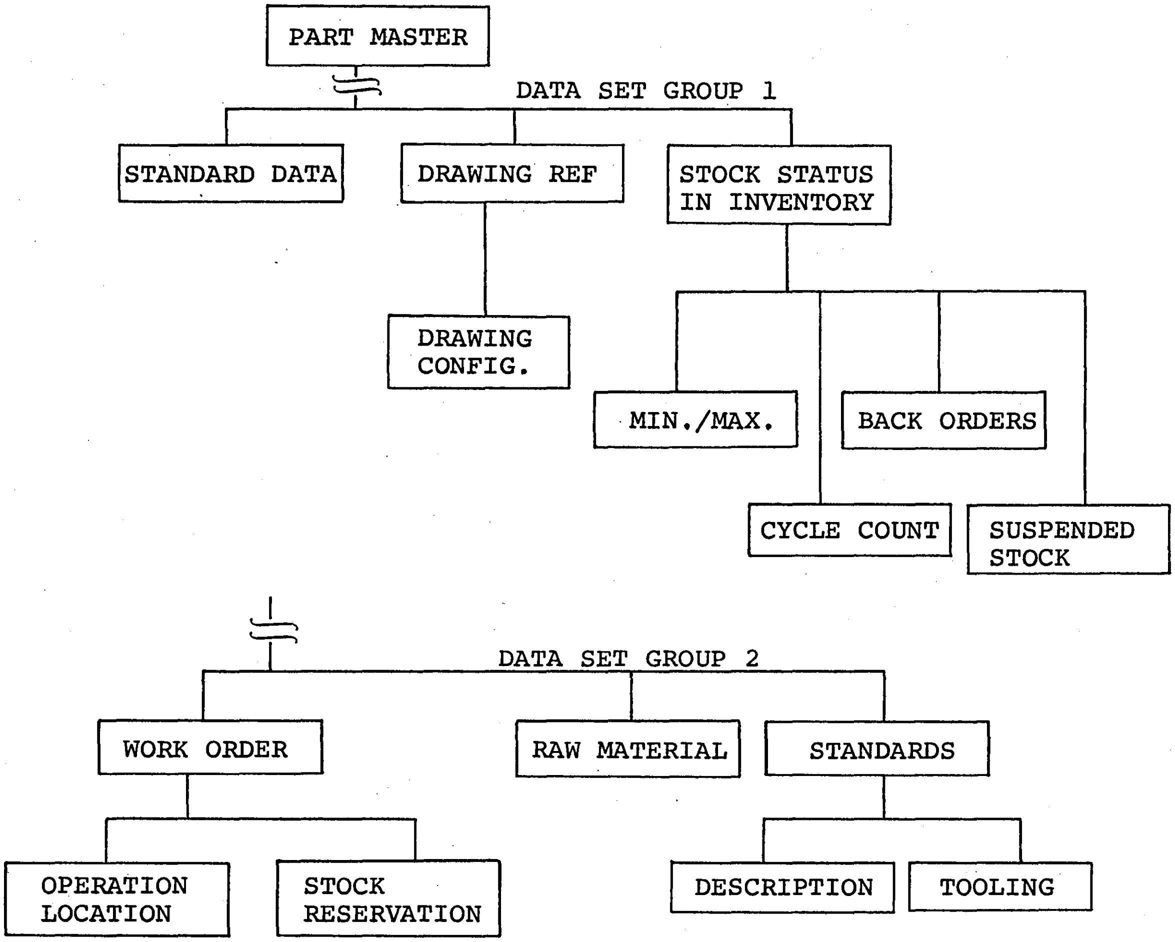


Figure 29. Part data base

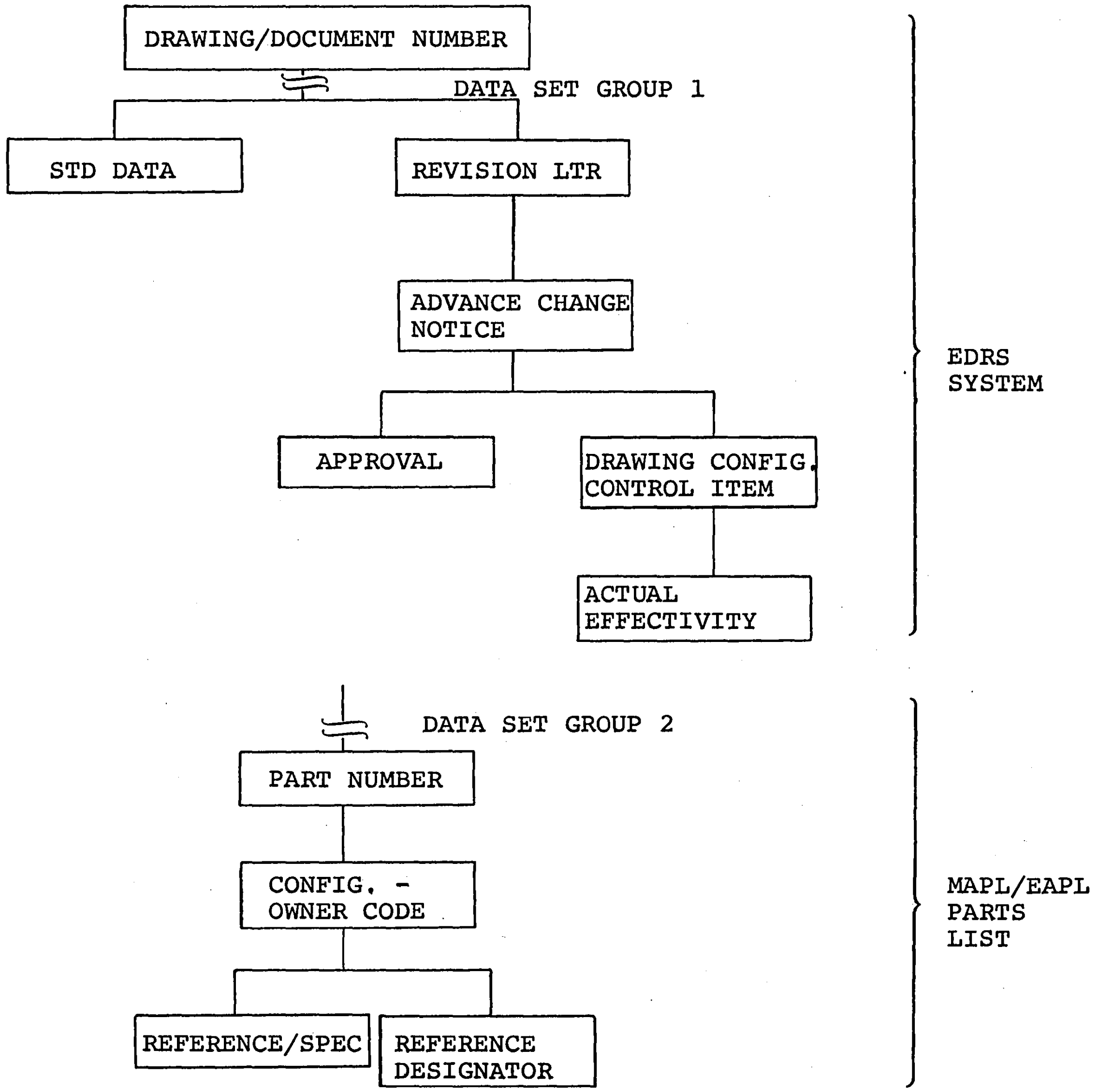


Figure 30. Drawing data base

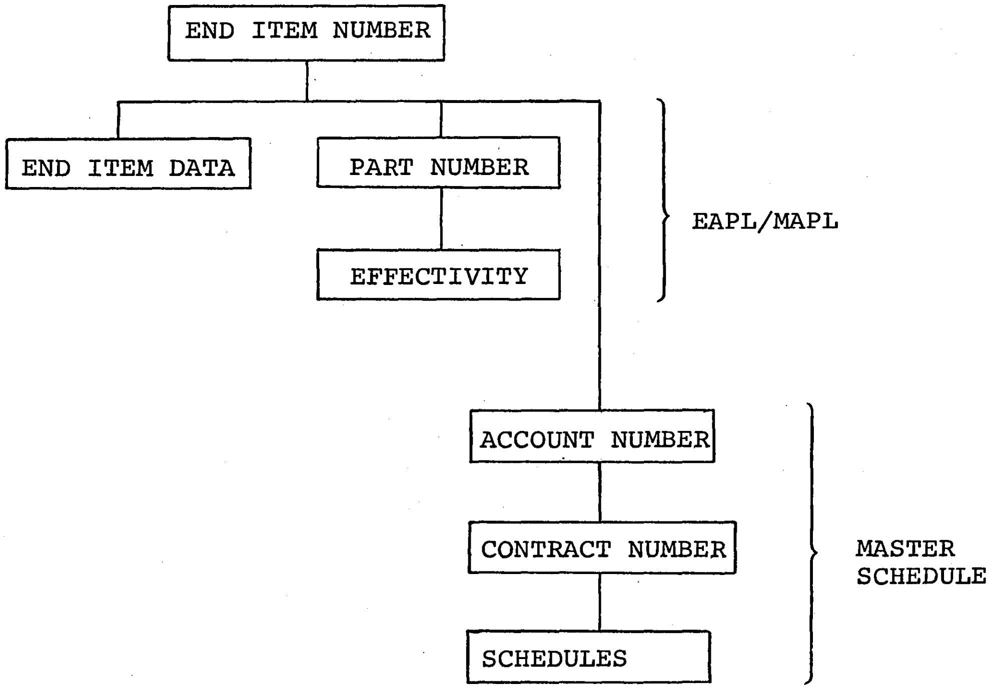


Figure 31. End item data base

The application portion of the IMS/360 sample problem includes the implementation of a small subset of this entire application. The data base structure of the application in the sample problem includes the segments and their structure described in Figure 32.

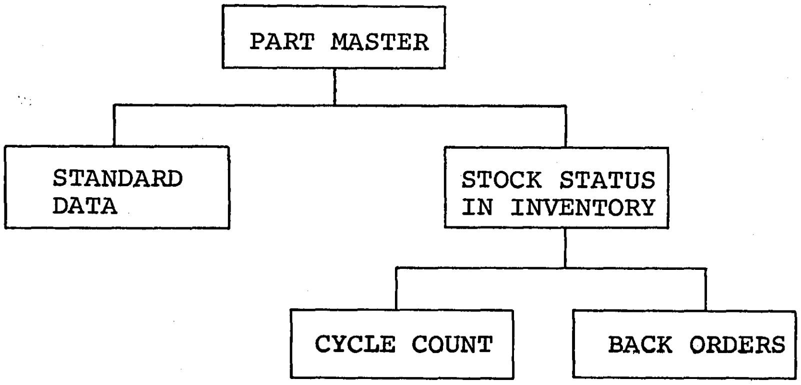


Figure 32. Sample problem -- application data base

This data base subset structure includes:

- One part number description segment for each part within the data base
- A standard data segment for each part. This segment provides additional information of a standard nature about the part.

- Inventory stock status segments for each part. The application is designed with multiple inventory locations permissible and normally required for any particular part.

- Zero to n cycle count and back-order segments for each inventory location of a particular part

In addition to the application data base substructure, the sample problem includes application programs:

1. To create the data base substructure in an IMS/360 Type 3 batch processing region. The input data for part, inventory, cycle count, back order, and standard part data to load into the data base substructure is provided.

2. For message processing programs and associated transactions to execute in an IMS/360 Type 1 region to:

   a. Inquire about a part and its description

   b. Inquire about a part's total inventory in all locations or by specific inventory location

   c. Add a new part and its description

   d. Add part inventory information by location to an existing part description

   e. Delete part inventory information by location

   f. Delete a part after deletion of all its subordinate part inventory information

   g. Close a part order to increase the part inventory at a specific location

   h. Disburse a specific quantity of a particular part on a planned or unplanned basis at a particular part inventory location, thereby reducing inventory

Figure 33 interrelates the sample problem transactions, programs, and data bases.

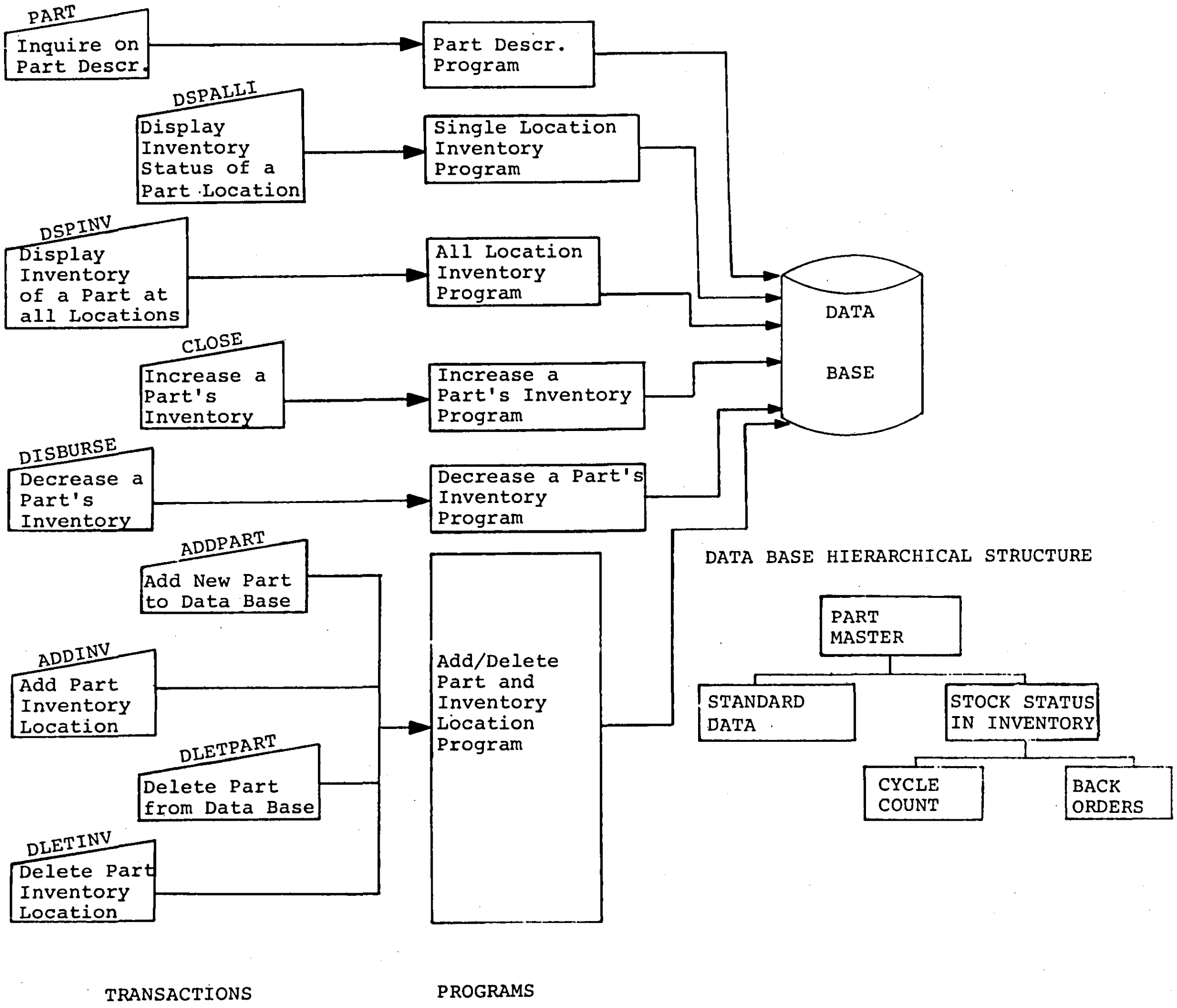

PART
Inquire on Part Descr.
Part Descr. Program
DSPALLI
Display Inventory Status of a Part Location
Single Location Inventory Program
DSPINV
Display Inventory of a Part at all Locations
All Location Inventory Program
DATA BASE
CLOSE
Increase a Part's Inventory
Increase a Part's Inventory Program
DISBURSE
Decrease a Part's Inventory
Decrease a Part's Inventory Program
ADDPART
Add New Part to Data Base
ADDINV
Add Part Inventory Location
DLETPART
Delete Part from Data Base
DLETINV
Delete Part Inventory Location
Add/Delete Part and Inventory Location Program
DATA BASE HIERARCHICAL STRUCTURE
PART MASTER
STANDARD DATA
STOCK STATUS IN INVENTORY
CYCLE COUNT
BACK ORDERS
TRANSACTIONS
PROGRAMS

Figure 33. Sample problem transactions, programs, and data bases

CREATING SAMPLE PROBLEM ENVIRONMENT

As outlined in the introduction to this chapter, a series of steps must be performed to create the sample problem environment. The remainder of this chapter describes these in detail or provides references for the required steps.

Copying IMS/360 Distribution Libraries

Figure 34 is an example of the JCL necessary for the allocation and cataloging of the data sets required for IMS/360 system definition and execution.

The tape move described in Chapter 2 should move the data sets into direct access libraries. If the DCB attributes of IMS.GENLIB and SYS1.MACLIB differ, it may be necessary to reblock IMS.GENLIB using the IEBCOPY utility prior to performing the IMS/360 system definition.

Performinq an IMS/360 System Definition

Prior to performing Stages 1 and 2 of IMS/360 system definition, certain data sets must be allocated and cataloged. Figure 34 is an example of the JCL required to allocate and catalog the data sets required by the sample problem. Space requirements should be adjusted if devices other than 2311 are to be used. If data set names are to be changed from those shown in Figures 34 and 35, refer to Chapter 4 of this manual for assistance.

```
//ALLOCATE JOB1.IMS,MSGCLASS=A,MSGLEVEL=1,PRTY=12
//EXEC PGM=IFHPROGM
//TWO  DD VOL=SER=222222,UNIT=2311,DISP=OLD
//THR  DD VOL=SER=333333,UNIT=2311,DISP=OLD
//ILIB01    DD VOL=SER=ILIB01,UNIT=2311,DISP=OLD
//ILIB02    DD VOL=SER=ILIB02,UNIT=2311,DISP=OLD
//SYSPRINT DD SYSOUT=A
//SYSIN DD *,DCB=BLKSIZE=80
SCRATCH VTOC,VOL=2311=222222,PURGE
SCRATCH VTOC,VOL=2311=333333,PURGE
SCRATCH VTOC,VOL=2311=ILIB01,PURGE
SCRATCH VTOC,VOL=2311=ILIB02,PURGE
//LIBRARYS EXEC PGM=IEHPROGM,REGION=100K
//SYSPRINT DD SYSOUT=A
//SIDC01 DD UNIT=2311,VOL=SER=SIDC01,DISP=OLD
//TEMPSET DD DSN=TEMPSET,UNIT=2311,VOL=SER=222222,DISP=1(,CATLG),
//          SPACE=(TRK,(1,1))
//CATALOG DD DSN=SYSCTLG,UNIT=2311,VOL=SER=ILIB02,DISP=(,KEEP),
//          SPACE=(TRK,(2,1))
//RESLIB DD DSN=IMS.RESLIB,UNIT=2311,VOL=SER=ILIB02,DISP=(,KEEP),
//          SPACE=(CYL,(40,5,20)),DCB=SYS1.LINKLIB
//MACLIB DD DSN=IMS.MACLIB,UNIT=2311,VOL=SER=ILIB01,DISP=(,KEEP),
//          SPACE=(CYL,(30,5,15)),DCB=SYS1.MACLIB
//PGMLIB DD DSN=IMS.PGMLIB,UNIT=2311,VOL=SER=ILIB01,DISP=(,KEEP),
//          SPACE=(CYL,(10,2,10)),DCB=SYS1.LINKLIB
//PSBLIB DD DSN=IMS.PSBLIB,UNIT=2311,VOL=SER=ILIB02,DISP=(,KEEP),
//          SPACE=(CYL,(10,2,5)),DCB=SYS1.LINKLIB
//DBDLIB DD DSN=IMS.DBDLIB,UNIT=2311,VOL=SER=ILIB02,DISP=(,KEEP),
//          SPACE=(CYL,(10,2,5)),DCB=SYS1.LINKLIB
//IQCR DD DSN=IMS.IQCR,VOL=SER=ILIB01,DISP=(,KEEP),UNIT=2311,
//          SPACE=(CYL,(5,1)),DCB=DSORG=PS
//IMSG DD DSN=IMS.IMSG,VOL=SER=ILIB01,DISP=(,KEEP),UNIT=2311,
//          SPACE=(CYL,(5,1)),DCB=DSORG=PS
//OQCR DD DSN=IMS.OQCR,VOL=SER=ILIB02,DISP=(,KEEP),UNIT=2311,
//          SPACE=(CYL,(5,1)),DCB=DSORG=PS
//OMSG DD DSN=IMS.OMSG,VOL=SER=ILIB02,DISP=(,KEEP),UNIT=2311,
//          SPACE=(CYL,(10,2)),DCB=DSORG=PS
//SYSIN DD *,DCB=BLKSIZE=80
 RELEASE INDEX=IMS
  DLTX INDEX=IMS,CVOL=2311=SIDC01
  CONNECT INDEX=IMS,CVOL=2311=SIDC01,VOL=2311=ILIB02
 CATLG DSNAME=IMS.GENLIB,VOL=2311=222222,CVOL=2311=ILIB02
 CATLG DSNAME=IMS.LOAD,VOL=2311=333333,CVOL=2311=ILIB02
  CATLG DSNAME=IMS.RESLIB,VOL=2311=ILIB02,CVOL=2311=ILIB02

  CATLG DSNAME=IMS.MACLIB,VOL=2311=ILIB01,CVOL=2311=ILIB02
  CATLG DSNAME=IMS.PGMLIB,VOL=2311=ILIB01,CVOL=2311=ILIB02
  CATLG DSNAME=IMS.PSBLIB,VOL=2311=ILIB02,CVOL=2311=ILIB02
  CATLG DSNAME=IMS.DBDLIB,VOL=2311=ILIB02,CVOL=2311=ILIB02
  CATLG DSNAME=IMS.PROCLIB,VOL=2311=ILIB01,CVOL=2311=ILIB02
  CATLG DSNAME=IMS.IQCR,VOL=2311=ILIB01,CVOL=2311=ILIB02
  CATLG DSNAME=IMS.OQCR,VOL=2311=ILIB02,CVOL=2311=ILIB02
  CATLG DSNAME=IMS.IMSG,VOL=2311=ILIB01,CVOL=2311=ILIB02
  CATLG DSNAME=IMS.OMSG,VOL=2311=ILIB02,CVOL=2311=ILIB02
```

Figure 34. Example of allocation and cataloging

Having completed the allocation of required data sets, Stage 1 of IMS/360 system definition is performed next. Figure 35 describes the control cards needed as input to system definition. The TRANSACT, PROGRAM, and DATABASE cards describe the resources of the application sample. For the sake of simplicity, only one telecommunications line group, one line, and one physical terminal (2740) are described. Two logical terminals, the MASTER and one named HOWARD, are described.

Logical terminal HOWARD is used by the application as a destination for exception messages.

If the user of the sample problem desires to perform the sample problem by means of a 2260 Display Station, it must be included in his Stage 1 system definition. The user must follow the rules of Chapter 4 of this manual for system definition and modify the Figure 35 control cards accordingly. (Warning: IMS/360 does not support the 2260 Display Station as a master terminal.) Prior to Stage 1 of IMS/360 system definition, IMS.GENLIB and IMS.LOAD must be cataloged. Those parameters in the system definition control cards which are underlined can be redefined by the IMS/360 user to meet the requirements of his data processing environment with no effect on the application.

```
//IMSDEF    JOB 1,IMS,MSGLEVEL=1
//STEP      EXEC    PGM=IEUASM,PARM='DECK,NOLOAD'
//SYSLIB    DD  DSN=IMS.GENLIB,DISP=SHR
//                DD  DSN=SYS1.MACLIB,DISP=SHR
//SYSPRINT  DD  SYSOUT=A
//SYSPUNCH  DD  SYSOUT=B
//SYSUT1    DD  UNIT=SYSDA,SPACE=(1700,(500,50))
//SYSUT2    DD  UNIT=SYSDA,SPACE=(1700,(500,50))
//SYSUT3    DD  UNIT=(SYSDA,SEP=(SYSUT1,SYSUT2)),                          X
//                    SPACE=(1700,(500,50))
//SYSIN         DD  *

    IMSCTRL     SYSTEM=(MVT,ALL),MAXIO=7,MAXREGN=1,                        X
                COMMSVC=(244,245),OCENDA=Z8,                               X
                OSAMSVC=243,MSGBUFF=10,CKPT=500

    APPLCTN     PSB=DFSSAM02,PGMTYPE=TP
        DATABASE    DBD=DI21PART,INTENT=SHARE
          TRANSACT  CODE=PART,PRTY=(7,10,2),INQUIRY=YES
    APPLCTN     PSB=DFSSAM03,PGMTYPE=TP
        DATABASE    DBD=DI21PART,INTENT=SHARE
          TRANSACT  CODE=DSPINV,PRTY=(7,10,2),INQUIRY=YES
    APPLCTN     PSB=DFSSAM04,PGMTYPE=TP
        DATABASE    DBD=DI21PART,INTENT=UPDATE,LOG=YES
          TRANSACT  CODE=ADDPART,PRTY=(7,10,2),INQUIRY=NO
          TRANSACT  CODE=ADDINV,PRTY=(7,10,2),INQUIRY=NO
          TRANSACT  CODE=DLETPART,PRTY=(7,10,2),INQUIRY=NO
          TRANSACT  CODE=DLETINV,PRTY=(7,10,2),INQUIRY=NO
    APPLCTN     PSB=DFSSAM05,PGMTYPE=TP
        DATABASE    DBD=DI21PART,INTENT=UPDATE,LOG=YES
          TRANSACT  CODE=CLOSE,PRTY=(7,10,2),INQUIRY=NO
    APPLCTN     PSB=DFSSAM06,PGMTYPE=TP
        DATABASE    DBD=DI21PART,INTENT=UPDATE,LOG=YES
          TRANSACT  CODE=DISBURSE,PRTY=(7,10,2),INQUIRY=NO
    APPLCTN     PSB=DFSSAM07,PGMTYPE=TP
        DATABASE    DBD=DI21PART,INTENT=SHARE
          TRANSACT  CODE=DSPALLI,PRTY=(7,10,2),INQUIRY=YES
    LINEGRP     DDNAME=DD2740
       LINE     FEAT=POLL,ADDR=032
         TERMINAL   ADDR=E2
        NAME        MASTER
         NAME       HOWARD
MSGQUEUE        QCRIN=(IQCR,IMS.IQCR,2311,ILIB01),                         X
                QCROUT=(OQCR,IMS.OQCR,2311,ILIB02),                        X
                MSGIN=(IMSG,IMS.IMSG,2311,ILIB01),                         X
                MSGOUT=(OMSG,IMS.OMSG,2311,ILIB02)
    MASTTERM    MASTER
    MACLIB      COPY=ALL,UNIT=2311,VOLNO=ILIB01
    RESLIB  UNIT=2311,VOLNO=ILIB02
    PGMLIB
    PSBLIB
    DBDLIB
    PROCLIB     PDS=SYS1.PROCLIB
    IMSGEN      ASMPRT=ON,LEPRT=(XREF,LIST),UT1SDS=TEMPSET
    END
```

Figure 35. Input to system definition

Refer to Chapter 4 of this manual if assistance is required in making control card changes. In particular, the message queue data set DD names and the IMS/360 library names should be reviewed.

The communication line and terminal operands may be modified as required to conform to the user's System/360 and Operating System/360 specifications. Chapter 4 of this manual provides information on the various operands permitted.

Once Stage 2 of system definition is successfully completed, the IMS/360 user must perform the following.

1. Include the two Type 1 and the one Type 2 IMS/360 SVC modules in the Operating System/360 nucleus. This can be done with a relink-edit of the Operating System/360 nucleus if available user SVC numbers were generated at the time of Operating System/360 system generation. If available SVC numbers do not exist, the user must perform at least an Operating System/360 nucleus-only system generation to provide the required SVC numbers.

2. Copy the OSAM channel end appendage IGG019Z8 or the equivalent to SYS1.SVCLIB.

3. Allocate and catalog the four sequential data sets used for message queuing in this example. Their DD names are IQCR, OQCR, IMSG, and OMSG. The associated data set names are IMS.IQCR, IMS.OQCR, IMS.IMSG, and IMS.OMSG. Chapter 3 of this manual provides information for allocation of these data sets.

A narrative is provided at the end of the output listing from IMS/360 system definition Stage 1. This narrative describes the additional functions a system user must perform prior to execution of his IMS/360 system. Please read this narrative.

Performing a Data Base Description (DBDGEN) Generation

Part of the sample problem is the generation of a data base description which will be used by the sample application. The generation process consists of an assembly and linkage edit. A member in IMS.MACLIB titled DI21PART contains the source input to generation of a DBD. A procedure is placed in SYS1.PROCLIB by system definition. The following JCL should be used to invoke this procedure and use the DBD source input to create the DBD. The output of the DBD generation becomes a member in the partitioned data set IMS.DBDLIB.

```
//DBD       JOB     SAMPLE,MSGLEVEL=1
//              EXEC    DBDGEN,MBR=DI21PART
//C.SYSIN   DD      DSNAME=IMS.MACLIB(DI21PART),DISP=SHR
```

Performing a Program Specification Block Generation (PSBGEN)

A part of the sample problem involves generation of a program specification block (PSB). The generation process is called PSBGEN. Like DBDGEN, the process consists of an assembly and linkage edit. A member of IMS.MACLIB named DFSSAP04 contains the source input which will generate the PSB for the online application program named DFSSAM04. IMS/360 system definition places a procedure named PSBGEN in SYS1.PROCLIB. The following JCL uses this procedure to place the output PSB in the partitioned data set IMS.PSBLIB as defined in the PSBLIB statement of system definition.

```
//PSB       JOB     1,IMS,MSGLEVEL=1
//STEP      EXEC    PSBGEN,MBR=DFSSAM04
//C.SYSIN   DD      DSN=IMS.MACLIB(DFSSAP04),DISP=SHR
```

<u>Note</u>: The input member DFSSAP04 creates an output PSB named DFSSAM04.

Moving Sample Problem Programs and Control Blocks

The next step in the sample problem is to have the IMS/360 user relink-edit the remaining PSB's and programs for data base creation and message processing from IMS.LOAD into their respective IMS libraries (that is, IMS.PSBLIB and IMS.PGMLIB).

A load module exists within IMS.LOAD for each PSB and application program. The following JCL and link-edit control statement are used to relink-edit the PSB's from IMS.LOAD to IMS.PSBLIB.

```
//PSBMOVE   JOB 1,IMS,MSGLEVEL=1
//          EXEC    PGM=IEWL,REGION=110K,                               X
//          PARM='XREF,LIST,LET,NCAL,SIZE=(100K,7248)'
//SYSLIB    DD  DSNAME=SYS1.COBLIB,DISP=SHR
//SYSLIN    DD  DDNAME=SYSIN
//SYSLMOD   DD  DSNAME=IMS.PSBLIB,DISP=OLD
//SYSPRINT  DD  SYSOUT=I
//SYSOBJ    DD  DSNAME=IMS.LOAD,DISP=SHR
//SYSUT1    DD  UNIT=2311,DISP=(NEW,DELETE),                        X
//                  SPACE=(CYL,(10,1),RLSE)
//SYSIN         DD  *

    INCLUDE     SYSOBJ(DFSSAM11)
    NAME    DFSSAM01(R)
    INCLUDE     SYSOBJ(DFSSAM12)
    NAME    DFSSAM02(R)
    INCLUDE     SYSOBJ(DFSSAM13)
    NAME    DFSSAM03(R)
    INCLUDE     SYSOBJ(DFSSAM15)
    NAME    DFSSAM05(R)
    INCLUDE     SYSOBJ(DFSSAM16)
    NAME    DFSSAM06(R)
    INCLUDE     SYSOBJ(DFSSAM17)
    NAME    DFSSAM07(R)
    INCLUDE     SYSOBJ(DFSSAM18)
    NAME    DFSSAM08(R)
    /*
```

The parameters in the JCL statements that are underlined should be modified to conform to the user's system configuration.

The following JCL and control card statements are used to relink-edit the application program load modules from IMS.LOAD to IMS.PGMLIB.

```
//PGMMOVE   JOB 1,IMS,MSGLEVEL=1
//       EXEC    PGM=IEWL,REGION=110K,                                    X
//            PARM='XREF,LIST,LET,NCAL,SIZE=(100K,7248)'
//SYSLIB    DD  DSNAME=SYS1.COBLIB,DISP=SHR
//SYSLIN    DD  DDNAME=SYSIN
//SYSLMOD   DD  DSNAME=IMS.PGMLIB,DISP=OLD
//SYSPRINT  DD  SYSOUT=I
//SYSOBJ    DD  DSNAME=IMS.LOAD,DISP=SHR
//SYSUT1    DD  UNIT=2311,DISP=(NEW,DELETE),                            X
//              SPACE=(CYL,(10,1),RLSE)
//SYSIN         DD  *

    INCLUDE     SYSOBJ(DFSSAM01)
                ENTRY DLITCBL
    NAME    DFSSAM01(R)
    INCLUDE     SYSOBJ(DFSSAM02)
                ENTRY DLITCBL
    NAME    DFSSAM02(R)
    INCLUDE     SYSOBJ(DFSSAM03)
                ENTRY DLITCBL
    NAME    DFSSAM03(R)
    INCLUDE     SYSOBJ(DFSSAM04)
                ENTRY DLITCBL
    NAME    DFSSAM04(R)
    INCLUDE     SYSOBJ(DFSSAM05)
                ENTRY DLITCBL
    NAME    DFSSAM05(R)
    INCLUDE     SYSOBJ(DFSSAM06)
                ENTRY DLITCBL
    NAME    DFSSAM06(R)
    INCLUDE     SYSOBJ(DFSSAM07)
                ENTRY DLITCBL
    NAME    DFSSAM07(R)
    INCLUDE     SYSOBJ(DFSSAM08)
                ENTRY DLITCBL
    NAME    DFSSAM08(R)
```

Executing an IMS/360 Data Base Load in a Batch Environment

Once the programs and PSB's have been relink-edited to their respective libraries, the application data base may be created. Before this data base is loaded, the user must allocate for the Operating System/360 data sets which represent the data base. One ISAM and one OSAM data set are required. The DD card ddnames for the ISAM and OSAM data sets are DI21PART and DI21PARO, respectively; the data set names for the ISAM and OSAM data sets are IMS.DI21PART and IMS.DI21PARO, respectively.

The user should now catalog these two data sets using the Operating System/360 utility IEHPROGM as shown in Figure 34. These data sets must be allocated for and cataloged before the data base load. IMS/360 system definition has placed into SYS1.PROCLIB a procedure to execute the data base load. The input data for the data base load execution, which contains the SYSIN for load, is a member of IMS.MACLIB. The name of the member is MFDFSYSN. The following JCL statements will invoke the procedure to create the data base.

```
//DBLOAD    JOB 1,IMS,MSGLEVEL=1
//STEP  EXEC    MFDBLOAD,PSER=333333,PUNIT=2311,OSER=222222,OUNIT=2311
```

The symbolic parameters designate the volume serial and unit for the prime and OSAM data sets.

The data base must be scratched and reallocated if a second execution of the MFDBLOAD procedure is desired.

A message is printed on the Operating System/360 console when the data base load is started, and another when the load is completed.

Initializinq IMS/360 in an Online Environment

At this point, the IMS/360 system has been defined for the user's environment, the application sample DBD has been created, the PSB's and programs have been relink-edited to their respective libraries, and the data base has been built.

The user is now ready to execute the IMS/360 telecommunications (Type 0) region control program and to perform message processing in an IMS/360 Type 1 region.

The system user should review Chapter 3 of this manual and the *Ims/360 Operations Manual, Volume II - Machine Operations* for information concerning IMS/360 cold start. The procedure named IMS1, which is described in this manual, should be used to start the IMS/360 control program. The user must use the following JCL override statements which allocate the data sets created in the prior data base load.

```
//IMS       JOB       MSGLEVEL=1,PRTY=13
//      EXEC         IMS1
//NUCLEUS.DI21PART DD DSN=IMS.DI21PART,DISP=OLD,VOL=SER=333333,UNIT=2311
//NUCLEUS.DI21PARO DD DSN=IMS.DI21PARO,DISP=OLD,VOL=SER=222222,UNIT=2311
/*
```

After the IMS/360 Type 0 region has been initiated as an Operating System/360 job, a message is printed on both the Operating System/360 system console and the IMS/360 master terminal indicating IMS READY.

At this point, the master terminal operator should enter the restart command message:

```
/NRESTART CHKPT 0 FORMAT ALL
```

The FORMAT ALL parameter will cause the IMS/360 message queues to be formatted. Formatting is required only at the initial cold start or after an I/O error occurs in the queue data sets. Formatting requires about 2-1/2 seconds per 2311 cylinder and 10 seconds per 2314 cylinder. These times are approximately doubled if write-checking is included. Immediately upon entry of the cold start command, the IMS/360 system responds with a message:

```
*NRESTART IN PROGRESS
```

After completion of the restart, which includes opening the message log and message queue data sets and formatting the message queue data sets, the following message is generated:

```
*IMS COLD START COMPLETE, ENTER START COMMANDS
```

The system, via the Operating System/360 console, will request the mounting of a standard-label, nine-track tape for the system log during cold start.

Although the IMS/360 control program is now available for message entry, no message region exists for message processing. This may be accomplished by entering the /START REGION command from the master terminal.

The start region command causes an Operating System/360 reader, which will read the JCL packet for a message region into the Operating System/360 job queue, to be started. The JCL packet for the message region is obtained from the PROCLIB library specified in IMS/360 system definition. Once the message region has been started and has communicated with the IMS/360 Type 0 region, a message, IMS MESSAGE REGION STARTED, is transmitted to the master terminal. Message processing may now begin.

Executing the Online Applications from User Terminals

At this point, each transaction code is discussed. Both input and output information and format are included in the discussion. Figure 36, at the end of this discussion, provides a list of some part number records placed into the data base at the time of data base load. Those part numbers may be used by the system user to enter transactions. The generic transaction format for all the following transactions is:

```
TRANSCODEbOPERAND1,OPERAND2,...
```

The transaction code is separated from the first operand by one blank (b). All transactions described here are defined during system definition as INQUIRY=NO. Therefore, the transactions cannot be entered on an inquiry logical terminal associated with a dial communication line.

The first transaction, PART, allows the terminal operator to inquire into the part number data base for information from the part master and standard information segments of a particular part number. The input format is:

transaction code	part number
part	an960C10

The output or response format is:

```
part number     description     procurement code

PART=AN960C10   ; DESC=WASHER   ; PROC CODE=74

INV CODE=2  MAKE DEPT=12-00  PLAN REV NUM=    MAKE TIME= 63  COMM CODE=14
```

The second transaction, DSPALLI, allows the terminal operator to display all inventory, cycle count, and back-order information for a particular part. The input format is:

transaction code	part number
dspalli	an960c10

The output format is:

```
part number      description        procurement code

PART=AN960C10   ; DESC=WASHER       ; PROC CODE=74
```

followed by inventory description and detail information:

```
    AREA   INV   PROJ  DIV      UNIT   CURRENT     ON       IN     TOTAL  COUNT BACK
          DEPT    CD.           PRICE  REQMTS     ORDER    STOCK  DISBURSE TAKEN ORDR

1.          AA    165   11              126        85-      126      209     N    0
2.          AK    287   7F               88         0        88      137     N    0
3.   2      80    091   26              630         0       680     1057     N    0
```

The third transaction, DSPINV, allows the terminal operator to display inventory information at a particular inventory location. Assume that it is wished to display only the third inventory entry listed in the above output. The inventory location key is obtained by concatenating AREA, INVDEPT, PROJCD, and DIV.

The input format for this transaction is:

transaction code	part number	inventory key
dspinv	an960c10,	28009126

The resultant output is:

```
PART=AN960C10        ; DESC=WASHER          ; PROC CODE=74

AREA=2; INV DEPT=80; PRJ=091; DIV=26; PRICE=      .000; STK CT DATE=513; UNIT=EACH

CURR REQMTS=    630 ; ON ORDER=       0 ; TOTAL STOCK=    680

DISB PLANNED=   1053 ; DISB UNPLANNED=      4 ; STK CT VARIANCE=       0
```

The fourth transaction, ADDPART, allows the terminal operator to add a new part into the data base with its associated description.

The input format is:

transaction code	part number	description	proc.code
addpart	ab960c10,	rivet,	74

The resultant terminal output is:

```
PART NUMBER AB960C10    ADDED TO DATA BASE
```

The fifth transaction, ADDINV, allows the terminal operator to add inventory information to an existing part in the data base.

The input format is:

transaction code	part number	inventory key
addinv	ab960c10,	8009126

The resultant output is:

```
INVENTORY 8009126    ADDED TO PART NUMBER AB960C10
```

If the operator wishes to display the part's inventory information, he can enter:

```
DSPINV ab960c10,8009126
```

The resultant output is:

```
PART=AB960C10       ; DESC=RIVET         ; PROC CODE=74
AREA=8; INV DEPT=00; PRJ=912; DIV=6 ; PRICE=      .000; STK CT DATE=   ; UNIT=
CURR REQMTS=      0 ; ON ORDER=      0 ; TOTAL STOCK=      0
DISB PLANNED=      0 ; DISB UNPLANNED=      0 ; STK CT VARIANCE=      0
```

The sixth transaction code, DLETINV, allows the terminal operator to delete a specific inventory item for a specific part. The input format is:

transaction code	part number	inventory key
dletinv	ab960c10,	8009126

The resultant output is:

```
INVENTORY 8009126   DELETED FROM PART NUMBER AB960C10
```

If all the inventory items are deleted, then a particular part number may be deleted from the data base with the transaction code DLETPART.

The input format is:

transaction code	part number
dletpart	ab960c10

The resultant output is:

```
PART NUMBER AB960C10    DELETED FROM DATA BASE
```

The terminal operator may now wish to close an open order for a specific part in a specific inventory item. The transaction to close an open order is CLOSE. The input format is:

transaction code	part number	inventory key	quantity received
close	an960c10,	28009126,	15, 15

The resultant output is:

```
UPDATE  COMPLETE
```

The terminal operator may now wish to display inventory item 28009126 for part AN960C10. The output format is:

transaction code	part number	inventory key
dspinv	an960c10,	28009126

The resultant output is:

```
PART=AN960C10       ; DESC=WASHER         ; PROC CODE=74

AREA=2; INV DEPT=80; PRJ=091; DIV=26; PRICE=      .000; STK CT DATE=513; UNIT=EACH

CURR REQMTS=    630 ; ON ORDER=     15-; TOTAL STOCK=    695

DISB PLANNED=   1053 ; DISB UNPLANNED=      4 ; STK CT VARIANCE=      0
```

Notice that the on-order quantity has been reduced by 15 and the total stock quantity has been increased by 15 to 695 from the earlier display of this inventory information.

The final transaction code, DISBURSE, allows the terminal user to allocate a quantity on a planned or unplanned basis of a given part from a given inventory item. The input format is:

transaction code	part number	inventory key	disbursement planned or unplanned	quantity disbursed
DISBURSE	an960c10,	28009126,	u,	10

The resultant output is:

```
UPDATE COMPLETED
```

If the terminal operator now wishes to display the inventory information for key 28009126 and part number AN960C10, the input format is:

transaction code	part number	inventory key
dspinv	an960c10,	28009126

The resultant output is:

```
PART=AN960C10       ; DESC=WASHER         ; PROC CODE=74

AREA=2; INV DEPT=80; PRJ=091; DIV=26; PRICE=      .000; STK CT DATE=513; UNIT=EACH

CURR REQMTS=    630 ; ON ORDER=     15-; TOTAL STOCK=    685

DISB PLANNED=   1053 ; DISB UNPLANNED=     14 ; STK CT VARIANCE=      0
```

The user may now terminate the IMS/360 system with a checkpoint command such as described below.

Terminal input:

```
   /checkpoint purge
```

Resultant output:

```
   CHECKPOINT COMMAND IN PROGRESS

*CHKPT 99365/132102**IMSDBS**PURGE**
```

The following is a list of available part records in the data base which the user may employ for message processing. Those parts marked with an asterisk have dependent back-order segments. All parts have at least one dependent inventory status segment.

Part Numbers	Back Order Segments
AN960C10	
3003806	*
3007228	
3013412	
652799	
7438995P002	
7618032P101	*
922399-001	
82125-869	

A complete listing of the part numbers available on the data base may be obtained by executing the procedure MFDBDUMP as follows:

```
//DBDUMP     JOB 1,   IMS,MSGLEVEL=1
//STEP       EXEC     MFDBDUMP
```

This procedure assumes the data sets IMS.DI21PART and IMS.DI21PARO are cataloged.

INDEX

SH20-0635-1

Information Management S/360 for the IBM S/360 OM Printed in U.S.A. SH20-0635-1

IBM®

International Business Machines Corporation
Data Processing Division
112 East Post Road, White Plains, New York 10601
(USA only)

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

READER'S COMMENT FORM

Information Management S/360 for the IBM S/360
Operations Manual

SH20-0635-1

Please comment on the usefulness and readability of this publication, suggest additions and deletions, and list specific errors and omissions (give page numbers). All comments and suggestions become the property of IBM. If you wish a reply, be sure to include your name and address.

COMMENTS

fold

fold

fold

fold

- Thank you for your cooperation. No postage necessary if mailed in the U.S.A.
FOLD ON TWO LINES, STAPLE AND MAIL.

SH20-0635-1

YOUR COMMENTS PLEASE...

Your comments on the other side of this form will help us improve future editions of this publication. Each reply will be carefully reviewed by the persons responsible for writing and publishing this material.

Please note that requests for copies of publications and for assistance in utilizing your IBM system should be directed to your IBM representative or the IBM branch office serving your locality.

fold fold

FIRST CLASS
PERMIT NO. 1359
WHITE PLAINS, N. Y.

BUSINESS REPLY MAIL
NO POSTAGE NECESSARY IF MAILED IN THE UNITED STATES

POSTAGE WILL BE PAID BY...

IBM Corporation
112 East Post Road
White Plains, N. Y. 10601

Attention: Technical Publications

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]