File Number S360-31
Order Number GC28-6712-7

OS

IBM Systems Reference Library

OS SMF

This publication provides installation managers, system programmers, and operators with the information required to plan for, install, and use SMF (System Management Facilities).

SMF is an optional feature of the IBM System/360 Operating System that can be selected at system generation for the Multiprogramming with a Fixed number of Tasks (MFT) or Multiprogramming with a Variable number of Tasks (MVT) option of the operating system.

SMF collects system, job-management, and data-management information and links to user-written routines that can monitor the operation of jobs or job steps.

This publication introduces basic SMF concepts; describes SMF record formats, control program exits, and data-management requirements; tells how to plan, write, and test user-written exit routines; tells how to incorporate SMF into the operating system; and describes IPL and data-management procedures.

Eighth Edition (April 1973)

This edition is a major revision of, and makes obsolete, *IBM System/360 Operating System: System Management Facilities*, GC28-6712-6. Technical and editorial changes are summarized under "Summary of Major Changes for Release 21.7."

Technical changes to the text and illustrations are indicated by a vertical line to the left of the change.

This edition applies to Release 21.7 of the IBM System/360 Operating System. It also applies to all subsequent releases until otherwise specified in new editions or technical newsletters. To determine whether this edition is up to date, refer to *IBM System/360 and System/370 SRL Newsletter*, GN20-0360. The information contained in this publication is subject to significant change. Any such changes will be published in new editions or technical newsletters.

Requests for copies of this or other IBM publications should be made to the IBM branch office that serves you.

A form for reader's comments appears at the back of this publication. Address any additional comments concerning the publication to IBM Corporation, Programming Publications, Post Office Box 1900, Boulder, Colorado 80302.

Preface

This publication provides installation managers, system programmers, and operators with the information required to plan for, install, and use SMF.

This publication has the following major divisions:

- "Summary of Major Changes for Release 21.7," which summarizes the technical and editorial changes for Release 21.7.
- "Introduction," which describes the basic SMF functions, the relationship of SMF to the operating system and to user-written exit routines, and introduces SMF records and exits.
- "User-Written Routines," which provides information on planning, writing, and testing routines that may be supplied by the user.
- "Incorporating SMF into the System," which describes procedures for incorporating SMF into an operating system.
- "System Information and Requirements," which describes storage requirements, performance, and operational considerations such as IPL and data-management procedures.
- "SMF Records," which fully describes the SMF records.
- "Index," which is a subject index to this publication.

Required Publications

The following publications are required for use with the book you are now reading:

- *OS Messages & Codes*, GC28-6631, which contains a listing and explanation of the messages issued by SMF.
- *OS System Control Blocks*, GC28-6628, which contains additional information on the contents of SMF records.

Related Publications

The reader should be familiar with the information presented in the following publications:

- *OS Assembler (F) Programmer's Guide*, GC28-3756, which describes the ASMFCL cataloged procedure, which is used to link-edit sample exit routines.
- *OS JCL Reference*, GC28-6704, which describes the OUTLIM parameter, which is used in conjunction with a user-written exit routine.
- *OS PL/I (F) Programmer's Guide*, GC28-6594, and *OS PL/I (F) Language*, GC28-8201, which provide information needed to write exit routines in PL/I or to modify the sample report routine.
- *OS Sort/Merge*, GC28-6543, which describes the sort/merge program, which can be used to sort SMF records.
- *OS Data Management Services Guide*, GC26-3746, which describes the record descriptor word (RDW) used to write records in the SMF data set.
- *OS System Generation*, GC28-6554, which describes the system generation program used to include SMF into the operating system.
- *OS Utilities*, GC28-6586, which describes the IEBUPDTE and IEBDG utility programs, which are used to enter the SMFDEFLT data set into SYS1.PARMLIB and to generate samples of standard parameter lists for exit routines.

Notational Conventions

The format of the parameters and instructions shown in this publication is governed by the rules of notation discussed below.

Bold Type

Information in bold type (**NONE**, **NSL**, etc.) must be entered exactly as shown.

Italic Type

Information in italics (*xx*, *register address*, etc.) indicates data to be supplied by the user. In the following examples:

BUF = *n*
VOL = **SER** = *vol.ser.no.*

n is replaced by a number and *vol.ser.no.* is replaced by a volume serial number.

Special Characters

Special characters are used to indicate alternative items and required blank characters. Alternative items are separated by an OR sign (|). No more than one of the items separated by OR signs may be selected. In the following example:

1 | 2 | 3

only one of the values may be selected.

A required blank character is indicated by b̸. In the following example, a blank is required between DD and DUMMY:

DDb̸DUMMY

Punctuation

The punctuation used in the commands (commas, semicolons, colons, and apostrophes) must be entered as shown.

Braces

Braces { } indicate a choice of entry. You must include one entry. If there are several choices within braces, you may enter any one of the choices. The braces are never entered. In the following example:

{**NO | YES**}

you must enter either the word **NO** or the word **YES**.

Brackets

Brackets [ ] indicate an optional value. The brackets themselves are never entered. In the following example:

KEYWORD = *value*[,*value2*]

value2 is optional and need not be entered.

Underscores

Underscores indicate the value that is assumed if no value is entered. Values that are assumed are called defaults. In the following example:

{**Y̲E̲S̲ | NO**}

if no value is given, **YES** is assumed.

Contents

Figures and Tables

Figures

Tables

Summary of Changes for Release 21.7

- "Introduction" has been rewritten. A brief introduction of the data collection routines is now included in "Introduction," along with introductory information on selecting SMF records.
- Each record type begins on a new page.

Introduction

SMF (System Management Facilities) is a feature of the IBM System/360 Operating System OS/VS that provides the means for gathering and recording information that can be used for billing customers or evaluating system usage. Information is gathered and recorded by SMF data-collection routines and by user-written exit routines. Because the data-collection and exit routines are independent of one another, they may be used in combination or separately.

Note: SMF cannot be used for monitoring system tasks.

SMF data collection routines gather several types of information:

- Accounting information, such as CPU time and device and storage usage.
- Data-set activity information, such as EXCP count and the user of the data set.
- Volume information, such as the space available on direct access volumes and error statistics for tape volumes.
- System use information, such as system wait time and I/O configuration.

The type of data to be collected can be modified by the operator at each initial program loading (IPL).

Through user written analysis routines and report routines, this information can be used in a variety of ways. For example, this information can be used to prepare customer's bills. The information might also be used to measure system usage against departmental standards of efficiency and performance.

SMF is not, however, confined to after-the-fact analysis. SMF allows you to write exit routines that can monitor a job or a job step at various points during its processing cycle—from control statement analysis to termination of the job. Therefore, by adding installation routines at the appropriate exits, standards of identification, priority, resource allocation, and maximum execution time can be enforced.

Here's an example of using both facilities provided by SMF. By using and analyzing the information obtained by the data-collection routines, the installation manager determines the average time each job step uses the CPU. In general, he finds that job steps exceeding this time limit are in a loop, or an unending wait state. Time is being wasted and overall efficiency is being impaired. Therefore, the average is used to establish a time limit through an exit routine for each job or job step running on the system; a job exceeding the expected time limit will be terminated. However, there must be some way to allow a job to exceed the expected time limit. Therefore, a routine is coded for the time limit exit. This routine allows the operator to extend the run time for selected jobs, such as the inventory program at year's end.

Data Collection

SMF writes 31 types of records. Data is collected for background (batch) processing and for foreground processing when Time Sharing Option (TSO) is included in the operating system. The data collected in SMF records (briefly described under "SMF Record Types" later in this chapter) includes:

- System information, which is contained in record types 0 through 3, 7 through 13, and 21 for background jobs and in record types 30 through 33, 38, 41, and 42 for foreground jobs. These records describe the operation of the system and the use of input/output devices. SMF formats and writes these records without taking user exits.
- Data-set information, which is contained in record types 14 through 18 for both background and foreground jobs. These records describe the activity of data sets. The records reflect pertinent portions of system control blocks at the conclusion of processing a data set. SMF formats and writes these records without taking user exits. Record type 20 is written at job commencement when data-set information is being recorded. This record enables post-processing routines to perform any accounting for data-set records as they are encountered rather than waiting for a record type 5, which is created at job termination.

- Volume information, which is contained in record type 19 for both background and foreground jobs. This record describes the space available on direct-access volumes. The record reflects pertinent portions of system control blocks at the conclusion of processing a volume. SMF formats and writes this record without taking user exits.
- Job and step information, which is contained in record types 4 through 6 for background jobs and in record types 34, 35, and 40 for foreground jobs. These records describe how a job or job step used the system. The records are formatted before any user exit is taken and are written after return from any exit, unless writing is inhibited by a return code from a user-written exit routine. You may use or modify these records.

The system, job, and job-step information can be used by user-written management-information programs that report system efficiency, performance, and usage. Information about user data sets can be used by user-written routines that report the data sets used by each job or job step. Information about the status of all removable direct-access volumes can be used by user-written routines that address problems of volume deterioration (defective tracks) and space fragmentation.

System information is recorded at IPL, when a VARY ONLINE or VARY OFFLINE command is entered, or when job management brings a device on line through allocation recovery. This information includes main storage size, input/output devices that are on line, and (for MFT) the amount of storage assigned to each active reader, writer, and problem program partition.

Job and job-step information is recorded at job initiation and at job and job-step termination. This includes identification fields and accounting information from the JOB and EXEC statements, start time, CPU time (the time a job or job step actually uses the CPU), SYSIN and SYSOUT usage, and job or job-step termination status.

Data-set information is recorded each time a data set opened by a user program is scratched, renamed, closed, or processed by EOV. Information includes data-set names, number of volumes, volume serial numbers, and various control block fields.

Direct-access volume information is recorded at IPL, when a HALT EOD command is processed, and when the system requests that a volume be demounted. This information includes volume serial number, VTOC address, owner identification number, number of unallocated tracks, amount of unallocated space, and size of the largest free extent.

An installation selects which records it will collect by including SMF default parameters in a member (SMFDEFLT) of SYS1.PARMLIB prior to system generation. (See "Selecting SMF Records" later in this chapter.) Additional records can be written to the SMF data set by user-written routines using the SMFWTM macro instruction. (See "SMFWTM Macro Instruction" in the chapter "User-Written Routines.") Record types 128 through 255 are available for user-written records.

Analysis of SMF-collected data can consist of simply listing the SMF data set, using the operating system's sort/merge program to order the data, or of performing detailed analysis operations using user-written analysis and report routines.

SMF Record Types

Various routines within the control program format SMF records and write them to the SMF data set.

The installation can suppress the writing of SMF records at IPL time and can select certain groups of SMF-formatted records through the use of SMF control parameters. Installation-formatted records can be created in the user-written routines to supplement or replace the SMF-formatted records.

Every SMF record begins with a standard record header that contains:

- A record type code.
- The date and time (time stamp) of writing.
- The system identification.

The SMF record types are briefly described in the following paragraphs. See the chapter "SMF Records" for detailed descriptions and formats of SMF records.

IPL Record (Type 0)

This record is written during system initialization (after IPL). It includes the machine storage size and the SMF options in effect.

Wait Time Record (Type 1)

This record is written at SMF initialization and at the first job-step termination following the expiration of a ten-minute interval of elapsed system time. It contains the CPU wait time accumulated during all of the ten-minute intervals that expired between two step terminations.

Dump Header Record (Type 2)

This record consists of only the standard record header. It indicates the beginning of a dump of the SMF data set from a direct-access device to tape. Record type 2 is written directly to the dump data set by the SMF dump program.

Dump Trailer Record (Type 3)

This record consists of only the standard record header. It marks the end of an SMF dump to tape. Record type 3 is written directly to the dump data set by the SMF dump program.

Step Termination Record (Type 4)

This record is written after the normal or abnormal completion of a job step. The record provides operating information such as the time the job step was started and completed, the CPU time, the amount of main storage allocated and used, and the devices used.

Job Termination Record (Type 5)

This record is written at normal or abnormal job termination. The record includes the start and stop time for processing of the job by the reader/interpreter, and the device type and class of the reader device.

Output Writer Record (Type 6)

This record is written when processing of a SYSOUT class by a standard writer ends or when the form changes. At least one output writer record is written for each SYSOUT class used by a job. If two or more forms are used within a class, one output writer record is produced for each form.

Data Lost Record (Type 7)

This record is written if no SMF data set is available for recording. It contains a count of SMF records not written and the start and end times of the period during which no records were written. The record is the first written when an SMF data set again becomes available.

I/O Configuration Record (Type 8)

This record is written during system initialization (after IPL). It consists of the standard record header and an entry describing each device that is on line at IPL.

VARY ONLINE Record (Type 9)

This record is written during processing of the VARY ONLINE operator command. It identifies the system resource being added to the configuration.

Allocation Recovery Record (Type 10)

This record is written during allocation recovery. It identifies the device brought on line, or otherwise made available, by device class, unit type, and device address. It also identifies the task requiring the device.

VARY OFFLINE Record (Type 11)

This record is written during processing of the VARY OFFLINE operator command. It identifies the system resource being removed from the configuration.

End-of-Day Record (Type 12)

This record is written during processing of the HALT or SWITCH operator command. It records the system *wait time* since the last wait time record (record type 1).

Dynamic Storage Configuration Record (Type 13)

This record, which applies only to MFT operations, is written at IPL and after each DEFINE command is processed. It shows the amount of storage assigned to each active reader, writer, and problem program partition.

INPUT or RDBACK Data Set Activity Record (Type 14)

This record is written when a user data set on tape or direct-access device that was opened for INPUT or RDBACK processing is closed or processed by end-of-volume (EOV).[1] It contains data-set information, such as creation and expiration dates, device type, and EXCP count.

OUTPUT, UPDAT, INOUT, or OUTIN Data Set Activity Record (Type 15)

This record is written when a user data set on tape or direct-access device that was opened for OUTPUT, UPDAT, INOUT, or OUTIN processing is closed or processed by end-of-volume (EOV).[1] It contains the same information as record type 14.

Scratch Data Set Status Record (Type 17)

This record is written when a user data set is scratched. It contains the data-set name, number of volumes, and volume serial numbers.

Rename Data Set Status Record (Type 18)

This record is written when a data set is renamed. It includes the old data-set name, new data-set name, number of volumes, and volume serial numbers.

[1] For type 14 and 15 records, a *user data set* is a data set that is defined by a DD statement and opened by a user program. Record types 14 and 15 may be written for system data sets if the data sets are defined by DD statements and opened by a user program.

Direct Access Volume Record (Type 19)

This record is written for all direct-access devices on line at IPL or when a HALT EOD or SWITCH SMF command is processed. It is also written whenever a direct-access volume is demounted. It contains volume information, such as the number of unused alternate tracks, number of unallocated cylinders and tracks, and number of cylinders and tracks in the largest free extent.

Job Commencement Record (Type 20)

This record is written each time a job is initiated. It contains the job log number (job name, entry time, and entry date), programmer's name, user identification, number of accounting fields on the JOB statement, and accounting fields.

ESV Record (Type 21)

This record is written by the Error Statistics by Volume (ESV) option when a user data set on magnetic tape is closed or processed by End-of-Volume. It contains error statistics information about the tape volume. This information can be used by IFHSTATR or by user-written routines that address problems of volume deterioration (defective tracks) and space fragmentation. (See "IFHSTATR" in *OS/VS Utilities*, GC35-0005.)

Start TS Record (Type 30)

This record is written each time the time sharing option is started with a START TS command. It contains TSO information, such as the time sharing initiation procedure name, time sharing member name in SYS1.PARMLIB, SMF foreground options, and additional information indicating the characteristics of the time sharing task started.

TIOC Initialization Record (Type 31)

This record is written each time the Terminal Input/Output Controller (TIOC) initialization routine is entered by the time sharing control (TSC) task as the result of a START TS command. It contains input/output control initialization information, such as the total number and size of time sharing buffers, number of buffers reserved on the free queue, number of users that constitute slack time, size of one terminal sharing block, and logged-on user change.

Driver Record (Type 32)

This record is written each time the driver initialization routine is entered by the TSC task as a result of a START TS command. It contains, for example, guaranteed background execution percentage. It also contains information, used in the time-sharing dispatching algorithm, about the users on each subqueue, such as the minimum time slice to be given to a user on the subqueue.

Driver Modify Record (Type 33)

This record is written each time the driver modify routine is entered as a result of a modify driver command. It contains a record of modifications made to the driver; this information is similar to that contained in record type 32.

TS-Step Termination Record (Type 34)

This record is written each time the LOGOFF function processes a step termination. The record contains job-step information, such as LOGON time, main storage occupancy time, number of TGETs and TPUTs issued, job-step CPU time, termination status, and main storage used. Input/output activity is recorded for each data set used by this job step. This record is similar to record type 4, which is produced for background jobs.

Logoff Record (Type 35)

This record is written each time a LOGOFF process has been completed. It contains job information, such as LOGON time, number of TGETs and TPUTs, session termination status, LOGON priority, LOGON enqueue time, termination indicator, SYSOUT classes for the session, and session CPU time. This record is similar to record type 5, which is produced for background jobs.

Initial TS Configuration Record (Type 38)

This record is written each time the time sharing option is started and after the TSC task has determined the time sharing configuration. It contains time sharing configuration information, such as the number of swap devices and a device entry for each swap device.

Dynamic DD Record (Type 40)

This record is written when the dynamic allocation function processes a deallocation, concatenation, or deconcatenation request. It contains dynamic allocation information, such as the device class, unit type, channel/unit address, and EXCP count. Information about the devices used is recorded in this record because record type 34 is not yet available. Record type 34 and 40, together, can be considered the equivalent of record type 4 for background jobs.

Modify TS Record (Type 41)

This record is written each time a MODIFY TS command is issued. It contains a record of modifications to TSO; this information is similar to that contained in record type 30.

Stop TS Record (Type 42)

This record is written by the terminate function of the TSC during the termination of time sharing operations. It is also written when the TSC abnormally terminates one or more time sharing regions and none is restarted. It includes the time sharing initiation procedure name and time sharing task identifier.

SMF Operation

Figure 1 shows the IBM System/360 Operating System incorporating SMF. The following paragraphs, which describe Figure 1, assume that user-written exit routines are supplied for all SMF exits, that all SMF-formatted records are written to the SMF data set, and that user-written analysis and report routines are supplied. In any real application, of course, the exit routines that are supplied and the records specified to be written to the SMF data set depend on the installation's requirements.

At IPL, an SMF initialization routine receives control and reads in the member (SMFDEFLT) of SYS1.PARMLIB that contains SMF control parameters. The options specified in these parameters—including such options as whether both the data-collection routines and the control program exits will be active and the definition of the SMF data sets—will be typed out at the console if the operator is allowed to modify them for the current work day. (The default parameters can be permanently changed by replacing the member in SYS1.PARMLIB.) Before job processing begins, the SMF initialization routine records information about IPL, initial input/output configuration, and the amount of storage assigned to each active reader, writer, and problem program partition (MFT only).

As the input stream is read in for processing, the reader/interpreter routine passes control to a user-written exit routine just before each job control statement is interpreted. This routine may verify any fields in the statement, modify JCL, or reject jobs that do not meet installation standards. As each job is enqueued, the same user-written routine receives control for further validity checking. Other exit routines receive control just before job and job-step initiation. These routines can decide whether to continue processing the job or job step.

Before a job is initiated, the initiator routines pass control to a user-written job initiation exit routine. In this routine, the user can decide whether to cancel or continue job processing based on accounting parameters associated with the job. Upon return from the routine, the SMF job commencement record is created and written to the SMF data set. This record is not written if the exit routine specifies that the job be canceled.

Before a job step is initiated, control is passed to a step initiation user-written exit routine. Here, processing similar to that done in the job initiation exit can be performed. When a step within the job ends, normally or abnormally, terminator routines create a job-step record and, before the record is written, pass control to a user-written accounting routine. The accounting routine may modify the SMF records, add a record to its own accounting data set, or add records to the SMF data set. This accounting routine also indicates whether the job is to continue (if a job-step termination is being processed) and whether the job-step termination record is to be written. Upon return, the indicators are examined and the job-step record is written to the SMF data set if requested.

At job termination, SMF creates a job information record and again passes control to the user-written accounting routine. Upon return from this routine, the SMF record indicators are examined and the SMF job-termination record is written if requested.

Data-set information is recorded whenever a data set opened by a user program is scratched, renamed, closed, or processed by end-of-volume (EOV). Direct-access volume information is recorded for online, direct-access devices at IPL when a volume is demounted and for online, direct-access devices when a HALT EOD or SWITCH command is issued.

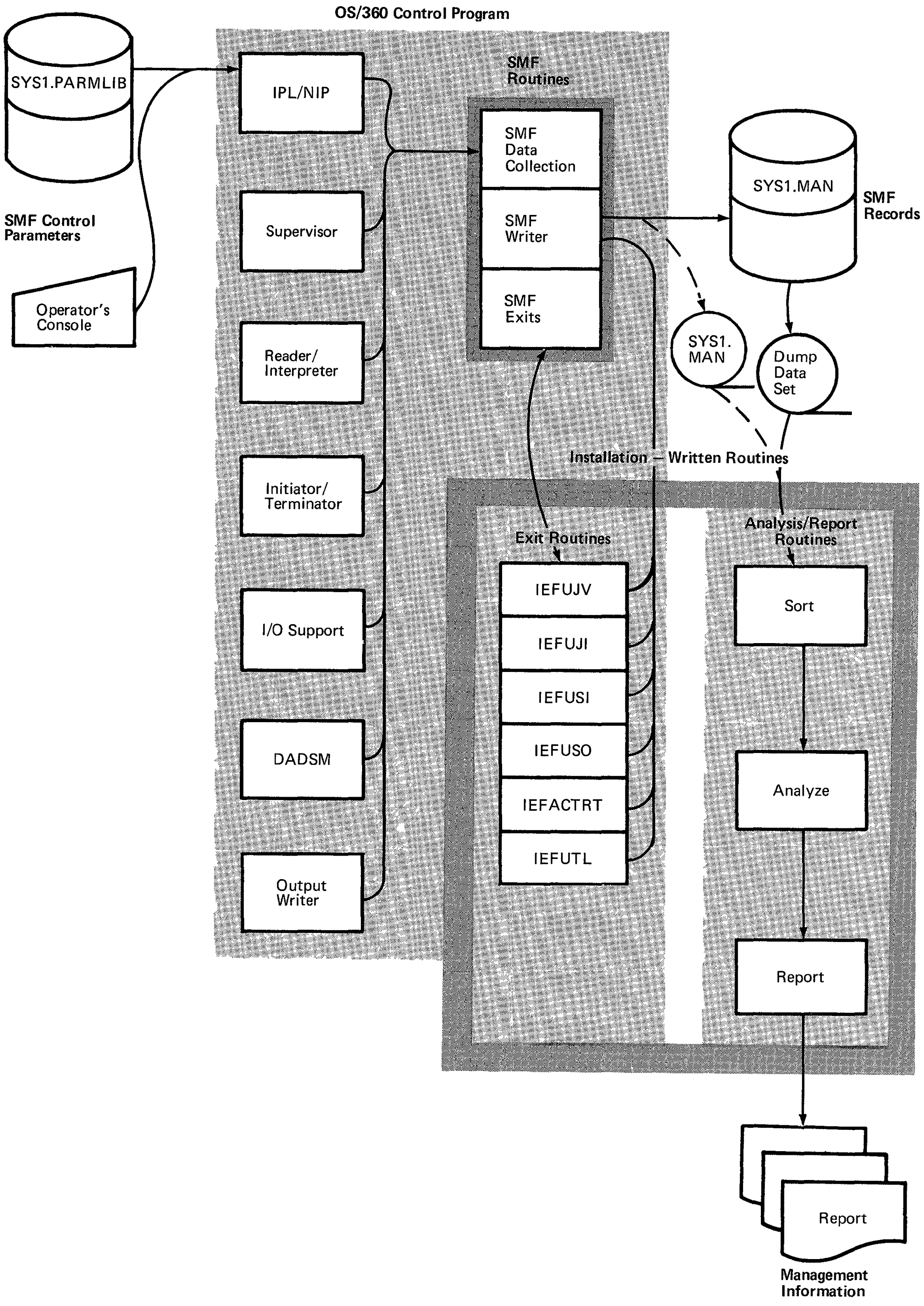

Figure 1. SMF in the Operating System

If the SMF data set is defined on a direct-access device, records continue to be written to the primary SMF data set (SYS1.MANX) until the end of the allocated extent is reached. When the end of the allocated extent is reached, SMF opens the alternate SMF data set (SYS1.MANY) and continues recording. The operator should then use the SMF dump program to copy SYS1.MANX to a dump data set.

Similar recording and copying operations continue throughout the work day, with SMF adding special records whenever a VARY command or allocation recovery changes the system configuration and, at the end of other intervals, to record system wait time. When the work day ends, the HALT EOD command from the operator causes the system wait time to be recorded and the SMF buffers to be emptied into the active SMF data set. This data set is closed and the alternate data set is selected for recording. The SMF dump program may then be used to copy the inactive SMF data set to the dump data set, which contains a complete history of the day's processing. The dump data set can serve as input to the user-written analysis and report routines, which may be executed as ordinary problem programs under the operating system.

A user-written routine may be used to list and total system usage by account number. A separate analysis program can process the SMF records in the order recorded, to detect excessive system wait time or inefficient use of input/output devices. This information can lead to improved system throughput by suggesting changes in the job mixture or device allocation.

SYSOUT Messages

In addition to the records written to the SMF data set, SMF writes four messages to the SYSOUT data set. These messages are assigned message numbers IEF373I, IEF374I, IEF375I, and IEF376I, and indicate, respectively, the start and end times for each job step and for each job. None of the SYSOUT messages are written for time sharing jobs. The text of these messages and an explanation of each is provided in *OS Messages & Codes*, GC28-6631.

Selecting SMF Records

The types of records written to the SMF data set can be controlled by four SMFDEFLT parameters (MAN, OPT, DSV, and REC) and also, to a limited extent, by the user-written exit routines, as follows:

- The MAN parameter is used to specify whether user-defined records, user and IBM-defined records, or no records are to be written to the SMF data set. This option allows the installation to make use of the SMF data-management routines for user-defined records. By overriding the suppression of all but installation-defined records at IPL, the full set of SMF records can be recorded when needed.
- The OPT parameter is used to specify that records containing job-step information are to be suppressed. This option allows the installation to reduce its SMF overhead if job-step information is not required. Job-step information, if suppressed by the SMFDEFLT parameter, can be overridden at IPL to make job-step information (record type 4) available when needed. The OPT keyword used in selecting records for TSO jobs is described in the chapter "Incorporating SMF into the System."
- The DSV parameter is used to specify that records containing data-set and direct-access volume information are to be recorded.
- User-written job and job-step termination routines can be used to provide additional control over records written to the SMF data set. After inspecting the job-step or job termination record, the routine specifies whether the record is to be written to the SMF data set by the code it returns to the system. This option allows installations to selectively write records to the SMF data set according to user-defined algorithms.
- The REC parameter is used to specify that record type 17 is to be written for temporary data sets[1] when DSV is equal to 2 or 3.

[1] The system determines that a data set is *temporary* if it has a system generated name. The system generates a name when the DD statement for a new data set does not include the DSNAME parameter, or when it contains a parameter of the form DSNAME= & name or DSNAME= & & name.

Table 1 summarizes the use of the SMFDEFLT parameters to select SMF records.

Table 1. Use of SMFDEFLT Parameters to Select SMF Records

Keyword	*Value*	*Meaning*	*Effect on SMF Records*
MAN	ALL	All SMF records.	Record types 0 through 255 are to be written to the SMF data set.
	NONE	No SMF records.	The SMF data set is not used.
	USER	Only user-formatted SMF records.	Only record types 128 through 255 are to be written to the SMF data set.
OPT	1	System and job information.	Record types 0 through 3 and 5 through 13 are created, but record type 4 is suppressed. Messages IEF375I and IEF376I are provided.
	2	System, job, and job-step information.	Record types 0 through 13 are created, and messages IEF373I through IEF376I are provided.
DSV	0	No information for data sets or direct-access volumes.	Record types 14, 15, and 17 through 20 are suppressed.
	1	Direct access volume information.[1]	Record type 19 is created, but record types 14, 15, 17, 18, and 20 are suppressed.
	2	Data-set information.[1]	Record type 19 is suppressed, but record types 14, 15, 17, 18, and 20 are created.
	3	Data-set and direct-access information.[1]	Record types 14, 15, and 17 through 20 are created.
REC[2]	0	No information for temporary data sets.	Record type 17 is created for non-temporary data sets only.
	2	Temporary data set information.	Record type 17 is created for temporary data sets as well as for non-temporary data sets.

[1] OPT must equal 2 if DSV is 2 or 3. If OPT equals 1 and DSV equals 2 or 3, the value OPT = 2 will be substituted.

[2] The REC parameter no longer governs Error Statistics by Volume (ESV) Collections.

The types of records written for TSO foreground jobs can also be controlled by two keyword parameters (OPT and EXT) specified in the member of SYS1.PARMLIB used in processing the START TS command. The contents of SMFDEFLT and the parameter formats are described in the chapter "Incorporating SMF into the System."

Control Program Exits

SMF provides exits in the control program that can be used by user-written routines. User-written exit routines have the facility to monitor each job at specific points from the time it is encountered in the input stream to termination. These routines are called from exits in the job scheduler and nucleus, and may be referred to as exit routines.

Like execution time for any other part of the control program, the execution time of the exit routines is added to system overhead and will degrade system throughput. The amount of the degradation depends on the length of the routines and the number of times each is performed during processing of a job. The advantages of including exit routines must be weighed against the factors affecting system throughput when choosing which exits to activate. Note that it is possible (by the SMFDEFLT parameters) to suppress all exits or step-related exits at IPL time, permitting the system to operate without the exit routines.

The exits from the reader/interpreter and the initiator/terminator can be taken by jobs in either the foreground or background. Exits for time-limit and output-limit exceeded can be taken for background jobs only.

An installation can make use of any or all of these exits by providing user-written exit routines and including them in system library SYS1.CI505 before system generation or in the link library SYS1.LINKLIB after system generation. Dummy routines are automatically provided for all unused exits.

The user-written exit routines can cancel jobs, write user-defined records to the SMF data set, open and close other user-defined data sets, or enforce installation standards, such as identification, priority, and resource allocation. All user-written exit routines receive control via a BALR instruction. Because these routines become part of the control program where errors can cause repeated system failure, thorough debugging is important.

All exit routines should be reenterable.

The formats of the parameters passed to each exit routine are described in the chapter "User-Written Routines." The procedure for adding user-written routines to the system is described in "Incorporating SMF into the System."

Job Validation (IEFUJV)

The job-management routine of the control program passes control to this user-written routine before interpreting each job control statement (or cataloged procedure) encountered in the input stream. One final entry is made just before the job is queued for scheduling.

A return code from this exit to job management specifies whether processing of this job is to continue.

Job Initiation (IEFUJI)

The initiator/terminator routine of the control program enters this user-written routine when a job on the input queue is selected for initiation.

A return code from this exit to the job-initiation routine specifies whether the job is to be started or canceled.

Step Initiation (IEFUSI)

The initiator/terminator enters the user-written job-step initiation routine just before each job step is started (prior to allocation).

A return code from this exit specifies whether the step is to be started or whether the job is to be canceled.

SYSOUT Limit (IEFUSO)

The I/O supervisor enters the SYSOUT limit routine when the number of logical records written to a SYSOUT data set exceeds the output limit specified on the DD statement.

A return code from this exit specifies whether the job is to be terminated or whether processing is to continue using a new SYSOUT limit.

Termination (IEFACTRT)

The terminator enters the user-written termination routine on the normal or abnormal termination of each job step and job.

Return codes to the initiator/terminator specify whether the job is to be continued or terminated (for job-step entry only), and whether the SMF record is to be written or skipped.

Time Limit (IEFUTL)

The timer interruption handler enters the time limit routine whenever one of the following time limits expires:

- The job-execution time limit (from the JOB statement or reader procedure).
- The step-execution time limit (from the EXEC statement or reader procedure).
- The job wait time limit (from SMFDEFLT).

The user-written routine returns a code that specifies whether the job step is to be terminated or whether processing is to continue.

User-Written Routines

This chapter contains information for planning and writing user routines. Your installation should provide two types of routines to take full advantage of the features of SMF:

- Exit routines that periodically monitor jobs and write user records to the SMF data set. (In MVT, all SMF exit routines can also write user records to installation-defined data sets. In MFT, only exit routine IEFUJV can write user records to installation-defined data sets.)
- Analysis and report routines that process and format information contained in the SMF and user data sets.

This chapter describes:

- Exit routine facilities and restrictions, including information on communication among user-written exit routines, parameters passed to exit routines, and the SMFWTM macro instruction, which can be used by exit routines to write a record to the SMF data set.
- Exit routines, including a full description of each of the SMF exit routines.

Exit Routine Facilities and Restrictions

The functions performed by your exit routines are determined solely by the requirements of your installation. The following paragraphs describe exit routine restrictions, formats of parameters passed to each exit, and the required return codes. Sample Assembler language exit routines are provided in a member (SMFEXITS) of SYS1.SAMPLIB. Figure 2 shows the JCL required to retrieve a listing of these sample routines.

```
//PRINT      JOB 123456,SMITH
//          EXEC PGM=IEBPTPCH
//SYSPRINT   DD  SYSOUT=A
//SYSUT1     DD  DSNAME=SYS1.SAMPLIB,DISP=(OLD,KEEP),UNIT=2314,
//               VOLUME=SER=DLIB02
//SYSUT2     DD  SYSOUT=A
//SYSIN      DD  *
     PRINT       TYPORG=PO,MAXNAME=1,MAXFLDS=1
     MEMBER      NAME=SMFEXITS
     RECORD      FIELD=(80)
/*
```

Figure 2. Obtaining a Listing of Sample Exit Routines

User-written exit routines should be written in reenterable code because the link edit attributes of the load module are subject to change from release to release. They must save registers when they receive control and restore registers before returning to the control program. Register 13 will contain the address of the register save area; register 14, the return address; and register 15, the entry point address.

User-written exit routines can communicate with each other via the user-commmunication field and the user-identification field passed to every exit routine. The user-communication field can be used to communicate between exits taken for different jobs, and it can be used to communicate between exits of the same job. These fields are passed to every exit routine and are logically part of each job. The user-communication field is initialized to binary zeros at reader initialization (starting a reader or restoring a transient reader). The user-identification field is initialized to blanks as each new job is read.

Any exit routine may obtain an additional work area by issuing a GETMAIN macro instruction that specifies an appropriate subpool in system queue space. Table 2 shows the characteristics of the subpools that can be specified. The address of the work area can be placed in the user-communication field. (You must consider the storage required by this work area when estimating the size of system queue space.)

Note: Communication areas obtained by exit IEFUJV are not maintained if the system is restarted.

If an exit routine cancels a job during or after job initiation, a job termination record (record type 5) is written to the SMF data set if the writing of records is permitted. If you require job cancellation information in the System Output Message Data Set, you may pass a message to module IEFYS from the termination exit routine IEFACTRT. At job termination time, you can determine if an IEFUJI or IEFUSI exit routine canceled a job by examining the job termination indicators in record type 5. At step termination time, you can determine if an IEFUSO or IEFUTL exit routine canceled a job by examining the completion code field in record type 4.

Table 2. Characteristics of Subpools in System Queue Space

Subpool Number	*Storage is Allocated*	*Storage is Deallocated*
253	In system queue space (MVT) or high in the partition (MFT).	Automatically by the system when the task terminates (MVT) or when the partition is freed (MFT).
254	In system queue space (MVT) or high in the partition (MFT).	Automatically by the system when the task terminates (MVT) or when the partition is freed (MFT).
255	In system queue space (MVT or MFT).	Explicitly, by issuing a FREEMAIN macro instruction.

Note: Only routines having a protect key of zero can obtain storage in system queue space (MVT or MFT) or high in a partition (MFT).

Your routines can use the SMFWTM macro instruction to write to the SMF data set. Some routines can also write to installation-defined data sets. (In MVT, all SMF exit routines can write to installation-defined data sets. In MFT, only exit routine IEFUJV can write to installation-defined data sets.) If you want to use your own data sets, you must define them, as follows:

- A data set used by exit routine IEFUJV requires a DD statement in the reader/interpreter cataloged procedure. If exit routine IEFUJV is to write to an installation-defined data set under the Remote Job Entry (RJE) or Graphic Job Processor (GJP) options, you must include a DD statement in the reader/interpreter cataloged procedure for RJE or GJP.
- A data set used by exit routines IEFUJI, IEFUSO, IEFUSI, IEFACTRT, and IEFUTL requires a DD statement in the initiator procedure.

Note: In MFT there is no initiator procedure; therefore, exit routines IEFUJI, IEFUSO, IEFUSI, IEFACTRT, and IEFUTL cannot write to installation-defined data sets.

Exit Routine Parameters

When an exit routine receives control, register 1 points to a list of four-byte addresses. The first entry in the list is common to all exit routines; it points to a 36-byte parameter area, which contains:

- Job log number, which includes the job name and the time of day and date (the time stamp) when the job entered the system.
- System identification.
- Job-step number.

Table 3 describes the format of this parameter area.

Table 4 summarizes the information available to each exit (in addition to common exit parameters, described below), when each exit is called, and the return from each exit to the control program. The names in parentheses are mandatory load-module and entry-point names that must be assigned.

Two additional fields are provided, one for communication among the installation routines, and one for an installation job identification to be included on job and job-step termination records and SYSOUT writer records originated by the job.

SMFWTM Macro Instruction

You may use the SMFWTM macro instruction in exit routines to write a record to the SMF data set. Routines using the macro instruction must have a protection key of 0.

Note: Record types 128 through 255 are available for user-written records.

The format of the SMFWTM macro instruction is:

[*symbol*] **SMFWTM** {*record address* | (*r*)}

where:

record address
is the symbolic address of the record to be written.

(*r*)
is a register containing the address of the record. You may use either the absolute register number or a symbolic designation. In either case you must enclose the value in parentheses; for example, (2) or (REG2).

The record to be written should include a standard SMF record header; see the chapter "SMF Records" for the header format. The record must be preceded by a record descriptor word (RDW); for a discussion of the RDW, refer to *OS Data Management Services Guide*, GC26-3746.

Table 3. Common Exit Parameter Area

Displacement from Pointer	*Length (in bytes)*	*Data Format*	*Description*
0	8	EBCDIC	Job name.
8	4	binary	Reader start time (in hundredths of a second) for the job.
12	4	packed decimal	Reader start date for the job. Format is 00YYDDDF. (F is a sign.)
16	4	EBCDIC	System identifier (two bytes) and model identifier (two bytes).
20	8	EBCDIC	User identification field. SMF places this data in all subsequent records for this job. This field is initialized to EBCDIC blanks between jobs.
28	1	binary	Number of the step being processed.
29	1	binary	Eight indicators of SMF options that are selected by the user. A bit setting of 1 indicates the related option was selected. *Bit* *Option* 0 Job accounting 1 Step accounting 2 Dynamic exits 3 Data-set accounting 4 Volume accounting 5 Reserved 6 Temporary data set scratch records 7 0—Background job 1—Foreground job
30	2	Binary	Reserved.
32	4	Binary	User communication field. This field is intended to be a user exit routine communication field. The reader/interpreter initializes this field to zeros *only* when the reader is started.

Note: The operating system sort/merge program will not process records less than 18 bytes long.

Table 4. Exit Routine Characteristics

Exit Routine	*Parameters Passed*	*Entered*	*Return*
Job Validation (IEFUJV)	JCL image, JCL type.	Each JCL card, job enqueue.	Continue or cancel.
Job Initiation (IEFUJI)	Programmer name, priority, account field.	Job initiation.	Continue or cancel.
Step Initiation (IEFUSI)	Step name, program name, step accounting fields.	Step initiation.	Continue or cancel.
SYSOUT Limit (IEFUSO)	Address of DCB.	SYSOUT limit exceeded.	Continue with new limit or cancel.
Time Limit (IEFUTL)	Type Code.	Job CPU limit exceeded, step CPU time limit exceeded, job wait time limit exceeded.	Continue with new time limit or cancel.
Termination (IEFACTRT)	Programmer name, job CPU time (in hundredths of a second), job accounting fields, step CPU time (in hundredths of a second), step accounting fields, completion code, SMF record.	Step termination, job termination.	Continue or cancel; write or skip SMF record.

The SMFWTM macro instruction returns a code in register 15, indicating the disposition of the user record, as follows:

- 0, which indicates that the record was written without error.
- 4, which indicates that the record was truncated because it would not completely fit in an empty SMF data set.
- 8, which indicates that the record was not written because the specified length was less than five bytes.
- 12, which indicates that the record was not written because the routine was not authorized to write to the SMF data set. (The requesting routine had a non-zero protection key.)
- 16, which indicates that the record was not written because: (a) the writing of records to the SMF data set is prohibited (that is, MAN = NONE was specified in SMFDEFLT), or (b) the writing of records is allowed but the SMF data set was full. If the SMF data set is full, it must be dumped before additional SMF records can be written. (See "Operational Considerations" in the chapter "System Information and Requirements" for the procedure for executing the SMF dump program.)

Exit Routines

The exits available for user-written routines are fully described in the topics that follow.

IEFUJV—Job Validation Exit

The IEFUJV exit can be taken from both the foreground and background. IEFUJV receives control just before each JCL statement for a job is interpreted and just before the job is queued for scheduling. At entry to the routine, register 1 points to a list of four-byte addresses, as follows:

1. The address of the common parameter area. (See Table 3.)
2. The address of an 80-character JCL statement image (in EBCDIC). JCL statements are identical to those listed in the SYSOUT data set; control statements containing only comments, however, are not made available. If a cataloged procedure is being executed, it is expanded before this exit routine receives control; the sequence of statements is JOB, EXEC PROC = ..., EXEC PGM = ..., followed by the other statements of the procedure. Override statements immediately precede the statement being overridden.

3. The address of a one-byte area that indicates the type of JCL statement being presented to the exit routine. The indicator will have one of the following binary values:

   0, which indicates a null statement.

   1, which indicates a JOB statement.

   2, which indicates an EXEC statement.

   4, which indicates a DD statement.[1]

   8, which indicates a PROC statement (for symbolic parameter definition).

   16, which indicates that no statement is being presented; this is the job enqueue entry to the exit routine.

The job validation routine may test and modify any of the operand fields in the job control statements and indicate, through a return code passed to job management, whether processing of this job is to continue. If the user modifies a job control statement, the modified statement is passed to the reader/interpreter for processing. The modified statement appears in the SYSOUT listing.

Editing of the job control statements must not result in additional job control statements or continuation cards.

Operand fields being added must not precede the first operands of any JCL statement image.

A user-written routine entered at this exit might perform any or all of the following:

- Validate any account fields included in the JOB and EXEC statements by comparison with a standard list.
- Validate or assign the priority.
- Validate or assign the REGION request.
- Validate or assign job time and job-step time parameters.
- Control output stream data by assigning a SPACE parameter to SYSOUT statements.
- Check for authorization to use data sets.
- Create installation-defined accounting records.
- Assign the user-identification field to be included in SMF termination records and the SYSOUT records for the job.

Note: If a job is canceled within the reader/interpreter, the job is not initiated and no job-termination record (record type 5) is written to the SMF data set. If you want a record of the canceled job, you may write one to the SMF data set from this exit routine.

Depending on the processing you want to perform, it may be more efficient to check JOB and EXEC statement accounting fields at the exits provided for job initiation and the first job-step initiation, respectively. The fields are passed as parameters to those routines, making a statement scan routine unnecessary. You may assign user identification at either of these exits and may write messages to the system output message data set from the IEFACTRT exit routine.

Before the IEFUJV exit routine returns to the control program, it must place a return code in register 15. A value of 0 indicates that processing of the job should continue; a binary value of 4 indicates that the job is to be canceled.

Sample IEFUJV Routine

The sample IEFUJV exit routine provided in SYS1.SAMPLIB checks the validity of a continued JOB statement and of values supplied for REGION, PRTY, TIME, and accounting parameters in JOB statements. Characters from the account number are used to index a table that contains allowable values for these parameters. If any value is found to be invalid, the job is terminated.

[1] The two high-order bits of the byte that indicates that this is a DD statement have the following meaning:

Bit	*Meaning When Set*
0	An out-of-sequence DD statement
1	A misplaced SYSCHK DD statement

IEFUJI—Job Initiation Exit

The IEFUJI exit can be taken from both the foreground and the background. IEFUJI receives control just before each job is initiated. At entry to the routine, register 1 points to a list of four-byte addresses, as follows:

1. The address of the common parameter area. (See Table 3.)
2. The address of a 20-byte area containing the programmer's name (in EBCDIC) from the JOB statement. This area is aligned left and, if necessary, padded with blanks.
3. The address of a one-byte area indicating the requested job priority.
4. The address of an area containing accounting information from the JOB statement. If the JOB statement contains no accounting information, the area is one byte of zeros.

The user-communication field is initialized to binary zeros when the reader is started or a transient reader is restored. Data placed in the user-communication field by the IEFUJV exit routine will become part of each job when it is enqueued and will be accessible by the exit routines entered in processing each job. Data placed in the user-communication field will be the same the next time this routine is entered, unless a new reader is started or a transient reader is restored.

The accounting fields are placed in a formatted list for easy access. Table 5 shows the format of the accounting information that is available to IEFUJI.

Table 5. Format of Accounting Information

Decimal Displacement	*Hexadecimal Displacement*	*Field Size*	*Data Format*	*Contents*
0	0	1	binary	Number of accounting fields in statement. (0 for no fields.)
1	1			Consecutive accounting fields.[1]

[1] Each entry for an accounting field contains the length of the field (one byte, binary), followed by the field (variable length, EBCDIC). The entry for a null accounting field contains a length of zero. (Null accounting fields are indicated by inserting consecutive commas in the accounting field of a JOB statement.)

If the installation uses major and minor account numbers with several fields, this exit is easier to use for account number processing than IEFUJV because of the formatted list. The requested job priority can be changed by this exit routine.

Before the IEFUJI exit routine returns to the control program, it must place a return code in register 15. A value of 0 indicates that processing of the job is to continue; a binary value of 4 indicates that the job is to be canceled.

Sample IEFUJI Routine

The sample IEFUJI exit routine provided in SYS1.SAMPLIB determines how long a job has been in the input job queue before it is initiated. This value and the job priority are written to the SMF data set as a user record.

IEFUSI—Step Initiation Exit

The IEFUSI exit can be taken from both the foreground and the background. IEFUSI receives control just before each job step is initiated, that is, prior to allocation. (If OPT = 1 was specified in the SMFDEFLT data set, this exit is not taken.) At entry to the routine, register 1 points to a list of four-byte addresses, as follows:

1. The address of the common parameter area. (See Table 3.)
2. The address of an eight-byte area containing the job-step name (in EBCDIC) from the EXEC statement. This area is aligned left and, if necessary, padded with blanks.
3. The address of an eight-byte area containing the program name (in EBCDIC) from the EXEC statement. This area is aligned left and, if necessary, padded with blanks.
4. The address of an area containing step accounting information from the EXEC statement. This area has the format shown earlier in Table 5. If the EXEC statement contains no accounting information, the field contains one byte of zeros. The accounting fields are listed in the same format as in the job initiation exit.

Before the IEFUSI exit routine returns to the control program, it must place a return code in register 15. A value of 0 indicates that processing of the job should continue; a binary value of 4 indicates that the job is to be canceled.

No sample IEFUSI exit routine is provided in SYS1.SAMPLIB.

IEFUSO—SYSOUT Limit

The IEFUSO exit can be taken from the background only. IEFUSO receives control when the number of logical records written to a SYSOUT data set exceeds the output limit specified by the OUTLIM parameter on the DD statement. The asynchronous exit interface routine sets a step "must-complete" status before SMF installation exit IEFUTL is given control. Consequently, the system will become interlocked if the exit routine enqueues on a resource already enqueued on by the job-step task or any of its subtasks. This enqueue can come from within SVCs, (for example, the SMFWTM and WTO macros). The form for the parameter is:

OUTLIM = *number*

where the number is the number of logical records plus the number of EXCPs from the OPEN and CLOSE macro instructions. (Refer to *OS JCL Reference*, GC28-6704.) If OUTLIM is not specified, or if OPT does not equal 2, output limiting is not done.

At entry to IEFUSO, register 1 points to a list of four-byte addresses, as follows:

1. The address of the common parameter area. (See Table 3.)
2. The address of the DCB for the data set.

Before the IEFUSO exit routine returns control to the control program, it must place a return code in register 15. A value of 0 indicates that the job step is to be terminated; a binary value of 4 indicates that the output limit is to be increased by a value placed in register 1, and processing is to continue.

This exit receives control in supervisor state, protect key 0, and disabled. It must return control in the same state.

Note: Unless the output limit has been increased, the exit to the user-written routine will again be taken when the next record is written to this SYSOUT data set.

No sample IEFUSO exit routine is provided in SYS1.SAMPLIB.

IEFACTRT—Termination Exit

The IEFACTRT exit can be taken from both the foreground and the background. IEFACTRT receives control when execution of a job or job step is terminated. (If OPT = 1 was specified in the SMFDEFLT data set, this exit is taken only at job termination.) At entry to the routine, register 1 points to a list of four-byte addresses, as follows:

1. The address of the common parameter area. (See Table 3.) At entry for job termination the parameter at displacement 28 contains the number of steps in the job.
2. The address of an eight-byte area containing the job-step name (in EBCDIC). This area is aligned left and, if necessary, padded with blanks. At job termination the address is zero.
3. The address of a 20-byte area containing the programmer's name (in EBCDIC). This area is aligned left and, if necessary, padded with blanks.
4. The address of a four-byte area that contains, in the first three bytes, job CPU time in hundredths of a second (a binary value). The last byte contains the number (binary) of accounting fields in the JOB statement.
5. The address of an area that contains accounting information from the JOB statement. This area has the format shown earlier in Table 5, excluding the first field shown (the number of accounting fields). If the JOB statement contains no accounting information, the area contains one byte of zeros.
6. The address of a four-byte area that contains, in the first three bytes, step CPU time in hundredths of a second (a binary value). The last byte contains the number (binary) of accounting fields in the EXEC statement. At job termination the address is zero.
7. The address of an area that contains accounting information from the EXEC statement. This area has the format shown earlier in Table 5, excluding the first field shown (the number of accounting fields). If the EXEC statement contains no accounting information, the area contains one byte of zeros. At job termination the address is zero.

8. The address of a two-byte area. The first byte is an indicator: if bit 7 is set to 1 when the exit routine is entered, the job has been canceled; if the exit routine sets bit 7 to 1, the job will be canceled. The second byte contains the number of the job step currently being processed.
9. The address of a two-byte area containing the termination status (condition or completion code) of the job or job step.
10. The address of an area containing a four-byte record descriptor word (RDW) immediately followed by the job-step termination record (record type 4) or the job termination record (record type 5) to be written to the SMF data set.

The parameters to this routine are passed in a list that is compatible with the parameter list supplied to user-written accounting routines that were written for previous versions of the operating system. The interface for this exit is designed so that such previous accounting routines, which were also named IEFACTRT, will operate correctly under SMF, except that the routine is not entered at job-step initiation.

As with past versions of IEFACTRT, output may be directed to the console or SYSOUT. Under SMF, installation information may be written either to the SMF data set or to an installation-supplied data set.

At entry to the routine, register 0 contains a binary code indicating the reason for entry; a value of 12 indicates job-step termination, and a value of 16 indicates job termination.

If your IEFACTRT exit routine writes messages for system output, the contents of register 12 must be the same as when the routine was entered, and register 13 must contain the address of a 45-word area. Figure 3 shows the technique that must be used when IEFACTRT writes to the System Output Message Class.

```
         .
         MVC    36(4,12),MSGADDR    MOVE MESSAGE ADDRESS AND
         MVC    42(2,12),MSGLEN     LENGTH TO SYSTEM TABLE
         L      REG15,VIEFYS        BRANCH AND LINK TO MESSAGE
         BALR   REG14,REG15         ROUTINE
         .
MSGADDR  DC     A(MSG)
MSG      DC     C'message text'
MSGLEN   DC     H'xx'               MESSAGE LENGTH
VIEFYS   DC     V(IEFYS)
```

Figure 3. Writing System Output Messages from IEFACTRT

Before the IEFACTRT exit routine returns to the control program, it must place return codes in registers 1 and 15, as follows:

- If register 1 contains a value of 4, the termination record is not to be written to the SMF data set; if it contains a value other than 4, the termination record is to be written.
- If register 15 contains a value of 4, the remaining job steps are to be canceled; if it contains a value other than 4, processing is to continue.

Sample IEFACTRT Routine

The sample IEFACTRT exit routine provided in SYS1.SAMPLIB changes the SMF job termination and job-step termination records (unless the job-step is flushed) to user records and attempts to write them to the SMF data set. If the data set is full, a message indicating lost SMF records is written to the console. At job termination a record containing the job name, programmer's name, and account number is written to the SYSOUT device.

Note: Record type 4 is not written for a job step that was not executed.

IEFUTL—Time Limit Exit

The IEFUTL exit can be taken from the background only. The asynchronous exit interface routine sets a step "must-complete" status before SMF installation exit IEFUSO is given control. Consequently, the system will become interlocked if the exit routine enqueues on a resource already enqueued on by the job-step task or any of its subtasks. This enqueue can come from within SVCs, (for example, the

SMFWTM and WTO macros). IEFUTL receives control when one of the following time limits expires:

- The job-execution time limit (from the JOB statement or reader procedure).
- The step-execution time limit (from the EXEC statement or reader procedure) or the remaining job time.
- The job wait time limit (from SMFDEFLT).

If a job time limit is specified on the JOB statement, job-step time will be set to job-step time limit (selected from the TIME value coded on the EXEC statement or time limit in the reader procedure) or the remaining job time, whichever is smaller.

If no time limit is specified on the JOB statement, each job-step is timed individually by using the TIME value from the EXEC statement or the time limit value from the reader procedure.

Specifying TIME = 1440 on the JOB statement eliminates job timing. Specifying TIME = 1440 on the EXEC statement without JOB time limit specification eliminates job-step timing for the step.

At entry to the routine, register 1 points to the address of the common parameter area. (See Table 3.) Register 0 contains a binary code indicating which time limit has expired, as follows:

- 0, which indicates that the job CPU time limit expired.
- 4, which indicates that the job-step CPU time limit expired.
- 8, which indicates that the job wait time limit expired.

Before the IEFUTL exit routine returns to the control program it must place a return code in register 15. A value of 0 indicates that the job-step is to be canceled; a binary value of 4 indicates that the job-step is to continue processing with an additional time allocation. The additional time (in timer units) must be placed in register 1. The number of timer units is determined via the following algorithm:

units = microseconds/26.04

The exit routine should control the number of extensions for a given job-step to prevent looping. The expiration may be recorded in the SMF data set or may be recorded as a message to the console or SYSOUT.

This exit routine receives control in the supervisor state, protect key 0, and disabled. It must return control in the same state.

Sample IEFUTL Routine

The sample IEFUTL exit routine provided in SYS1.SAMPLIB causes a job to be terminated if the job CPU time limit or job-step CPU time limit has been exceeded. If the job wait time limit has been exceeded, the limit is extended twice; on the third entry for an exceeded job wait time limit, the job is canceled. Each time the routine is entered for an exceeded job wait time limit, it writes a record describing the action taken to the SMF data set.

Testing Exit Routines

Because the exit routines provided by your installation will become a part of the control program, you must test them thoroughly. A test procedure (TESTEXIT) is provided in SYS1.SAMPLIB to aid in your testing.

Special Requirements for Testing Exit Routines

For testing purposes only, your exit routines must conform to the following requirements:

- Subpool 0 must be specified in GETMAIN macro instructions. (When testing is completed, subpool 253, 254, or 255 must be specified for the area used to communicate between exit routines, as shown earlier in Table 2.
- If the SMFWTM macro instruction is used in any of your routines, you must provide a special macro definition in the routine. When testing is completed the macro definition must be removed.

Figure 4 shows the SMFWTM macro instruction that is required for testing.

TESTEXIT Procedure

Included in the test procedure is an Assembler language source program (also named TESTEXIT). This source program attaches the data generator utility program (IEBDG) to create sample parameter lists; TESTEXIT then calls each exit routine being tested, passing to it the appropriate parameter list. Figure 5 illustrates the input/output and control flow of the TESTEXIT routine.

```
         MACRO
&NAME    SMFWTM   &MSGAD
         AIF      ('&MSGAD' EQ '').E1
         AIF      ('&MSGAD' EQ '(1)').BAL
         AIF      ('&MSGAD'(1,1) EQ '(').REGA
         AGO      .LODIT
.E1      MNOTE    '*** NO OPERAND SPECIFIED ***'
         MEXIT
.BAL     ANOP
         CNOP     0,4
&NAME    BAL      15,*+8
.LIST    DC       V(TSMFWTM)
         L        15,0(15)
         BALR     14,15
         MEXIT
.REGA    ANOP
&NAME    LR       1,&MSGAD(1)
         CNOP     0,4
         BAL      15,*+8
         AGO      .LIST
.LODIT   ANOP
&NAME    LA       1,&MSGAD
         CNOP     0,4
         BAL      15,*+8
         AGO      .LIST
         MEND
```

Figure 4. SMFWTM Macro Definition Required for Testing

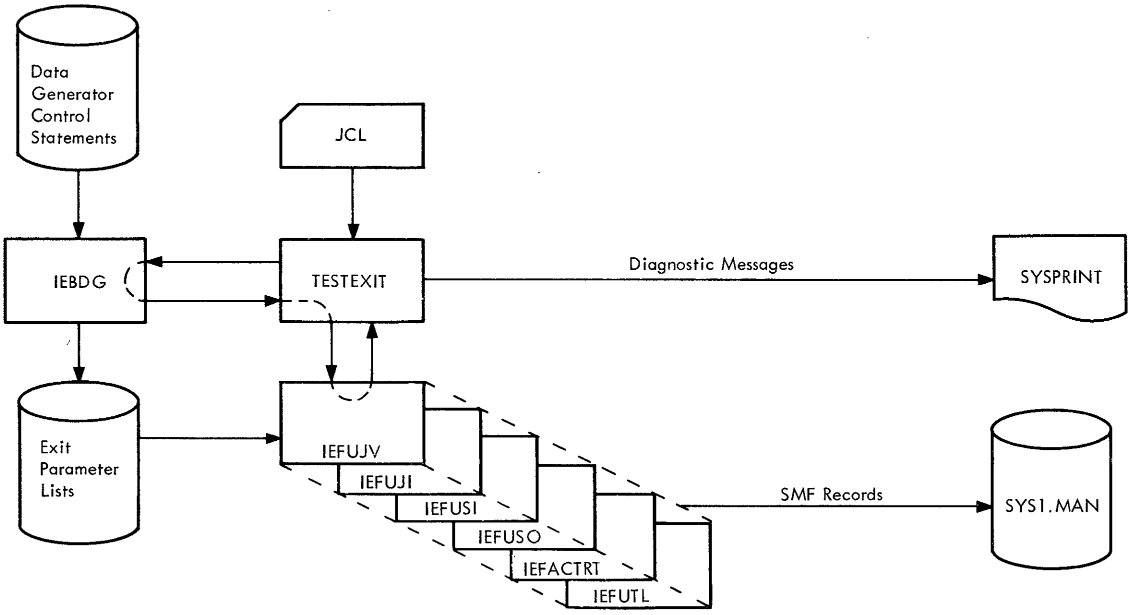

Figure 5. TESTEXIT Input/Output and Control Flow

Figure 6 illustrates the JCL included in the TESTEXIT procedure in an unmodified system.

```
//TESTEXIT     JOB  MSGLEVEL=1
//TEST         EXEC ASMFCL
//ASM.SYSIN    DD   *
```

(TESTEXIT Source Module)

```
/*
//LKED.SYSLMOD DD   DSNAME=TESTLIB,VOLUME=SER=231400,
//                  UNIT=2314,SPACE=(TRK,(5,2,1)),
//                  DISP=(NEW,KEEP)
//LKED.EXITS   DD   DSNAME=EXITLIB,VOLUME=SER=231400,
//                  UNIT=2314,DISP=OLD
//LKED.SYSIN   DD   *
 INCLUDE EXITS(IEFUJV,IEFUJI,IEFUSI,IEFUSO,IEFACTRT,IEFUTL)
 ENTRY TESTEXIT
 NAME TESTEXIT
/*
//DATAGEN      JOB  MSGLEVEL=1
//             EXEC PGM=IEBUPDTE,PARM=NEW
//SYSUT2       DD   DSNAME=DGINPUT,UNIT=2314,DISP=(,KEEP),
//                  VOLUME=SER=231400,SPACE=(TRK,(10,5,1)),
//                  DCB=(LRECL=80,BLKSIZE=400,RECFM=FB)
//SYSPRINT     DD   SYSOUT=A
//SYSIN        DD   DATA
./  ADD      NAME=UJV
```

(IEBDG Control Statements for IEFUJV)

```
./  ADD      NAME=UJI
```

(IEBDG Control Statements for IEFUJI)

```
./  ADD      NAME=USI
```

(IEBDG Control Statements for IEFUSI)

```
./  ADD      NAME=USO
```

(IEBDG Control Statements for IEFUSO)

```
./  ADD      NAME=UTL
```

(IEBDG Control Statements for IEFUTL)

```
./  ADD      NAME=ACT
```

(IEBDG Control Statements for IEFACTRT)

```
./  ENDUP
/*
//TESTING      JOB  MSGLEVEL=1
//JOBLIB       DD   DSNAME=TESTLIB,VOLUME=SER=231400,
//                  UNIT=2314,DISP=(OLD,KEEP)
//             EXEC PGM=TESTEXIT,
//                  PARM='UJV=26,UJI=10,USI=5,USO=2,
                    UTL=5,ACT=2'
//INUJV        DD   DSNAME=DGINPUT(UJV),DCB=(LRECL=80,
//                  BLKSIZE=400,RECFM=FB),DISP=(OLD,PASS),
//                  UNIT=2314,VOLUME=SER=231400
//INUJI        DD   DSNAME=DGINPUT(UJI),DCB=(LRECL=80,
//                  BLKSIZE=400,RECFM=FB),DISP=(OLD,PASS),
//                  UNIT=2314,VOLUME=SER=231400
//INUSI        DD   DSNAME=DGINPUT(USI),DCB=(LRECL=80,
//                  BLKSIZE=400,RECFM=FB),DISP=(OLD,PASS),
//                  UNIT=2314,VOLUME=SER=231400
//INUSO        DD   DSNAME=DGINPUT(USO),DCB=(LRECL=80,
//                  BLKSIZE=400,RECFM=FB),DISP=(OLD,PASS),
//                  UNIT=2314,VOLUME=SER=231400
//INUTL        DD   DSNAME=DGINPUT(UTL),DCB=(LRECL=80,
//                  BLKSIZE=400,RECFM=FB),DISP=(OLD,PASS),
//                  UNIT=2314,VOLUME=SER=231400
//INACT        DD   DSNAME=DGINPUT(ACT),DCB=(LRECL=80,
//                  BLKSIZE=400,RECFM=FB),DISP=(OLD,PASS),
//                  UNIT=2314,VOLUME=SER=231400
//OUTUJV       DD   DSNAME=UJV(OUT),UNIT=2314,DISP=(,PASS),
//                  SPACE=(TRK,(10,5,1)),VOLUME=SER=231400,
//                  DCB=(LRECL=80,BLKSIZE=400,RECFM=FB)
```

Figure 6. (Part 1 of 2) TESTEXIT Procedure JCL

```
//OUTUJI         DD   DSNAME=UJI( OUT ),UNIT=2314,DISP=( ,PASS ),
//                    SPACE=( TRK,( 10,5,1 ) ),VOLUME=SER=231400,
//                    DCB=( LRECL=80,BLKSIZE=400,RECFM=FB )
//OUTUSI         DD   DSNAME=USI( OUT ),UNIT=2314,DISP=( ,PASS ),
//                    SPACE=( TRK,( 10,5,1 ) ),VOLUME=SER=231400,
//                    DCB=( LRECL=80,BLKSIZE=400,RECFM=FB )
//OUTUSO         DD   DSNAME=USO( OUT ),UNIT=2314,DISP=( ,PASS ),
//                    SPACE=( TRK,( 10,5,1 ) ),VOLUME=SER=231400,
//                    DCB=( LRECL=80,BLKSIZE=400,RECFM=FB )
//OUTUTL         DD   DSNAME=UTL( OUT ),UNIT=2314,DISP=( ,PASS ),
//                    SPACE=( TRK,( 10,5,1 ) ),VOLUME=SER=231400,
//                    DCB=( LRECL=80,BLKSIZE=400,RECFM=FB )
//OUTACT         DD   DSNAME=ACT( OUT ),UNIT=2314,DISP=( ,PASS ),
//                    SPACE=( TRK,( 10,5,1 ) ),VOLUME=SER=231400,
//                    DCB=( LRECL=130,BLKSIZE=130,RECFM=FB )
//MANX           DD   UNIT=2314,VOLUME=SER=231400,
//                    SPACE=( TRK,( 3,1 ) ),DISP=( NEW,PASS ),
//                    DCB=( BLKSIZE=144,LRECL=140 )
//SYSPRINT       DD   SYSOUT=A,DCB=( BLKSIZE=136,LRECL=132 )
//DGPRINT        DD   SYSOUT=A
//SYSABEND       DD   SYSOUT=A
/*
```

Figure 6. (Part 2 of 2) TESTEXIT Procedure JCL

Following is a summary of the operations performed by the procedure shown in Figure 6:

- The TESTEXIT job assembles the TESTEXIT routine (not illustrated in Figure 6) and link-edits it with the exit routines being tested. The exit routines must reside in a partitioned data set (EXITLIB).
- The DATAGEN job, using the IEBUPDTE utility program, creates a partitioned data set (DGINPUT) containing control statements for the IEBDG utility program, which will be attached by the TESTEXIT program.
- The TESTING job includes the execution of the TESTEXIT program.

Using TESTEXIT

To use the TESTEXIT procedure you must do the following:

- Place your exit routines in a partitioned data set.
- Obtain a punched deck of TESTEXIT.
- Modify the procedure to meet your testing requirements.
- Execute the three jobs in the procedure.

The procedure provided in SYS1.SAMPLIB, without modification, can be used to link-edit the sample exit routines (also in SAMPLIB), generate sample parameter lists, and test the sample exit routines. You should consider linkage-editor and data-generator modifications in adapting the procedure to your testing requirements. These modifications are discussed in the topics that follow.

Linkage Editor Modifications

The linkage editor step of the first job (TESTEXIT), shown earlier in Figure 6, link-edits the TESTEXIT program with the exit routines. You must substitute an INCLUDE control statement specifying the names of the exit routines you are testing.

Data Generator Modifications

The second job (DATAGEN), shown earlier in Figure 6, creates a partitioned data set containing control statements for the IEBDG utility program. The control statements supplied with the procedure will generate samples of standard parameter lists. You should omit control statements and their associated ADD statements for any exit routines you are not testing. If you are testing for special conditions or require additional test parameters, you must make appropriate modifications and additions to the control statements.

Figure 7 shows the JCL that can be used to place user-written routines into EXITLIB, a partitioned data set.

Figure 8 shows the JCL that can be used to obtain a punched deck of TESTEXIT.

Note that you must provide control statements in such an order that the records subsequently generated by the IEBDG utility will be grouped as complete parameter lists that conform in length and format to the exit parameters previously defined in this chapter. (The code passed to exits IEFACTRT and IEFUTL in register 0 must be included as a one-byte parameter at the end of the parameter lists for those exits.)

For detailed information on the use of IEBDG control statements, refer to *OS Utilities*, GC28-6586.

```
//UPDTE        JOB  MSGLEVEL=1
//             EXEC PGM=IEBUPDTE,PARM=NEW
//SYSUT2       DD   DSNAME=EXITLIB,VOLUME=SER=231400,
//                  UNIT=2314,SPACE=(TRK,(10,5,1)),
//                  DCB=(LRECL=80,BLKSIZE=400,RECFM=FB)
//SYSPRINT     DD   SYSOUT=A
//SYSIN        DD   DATA
./  ADD        NAME=IEFUJV
(IEFUJV object deck)
./  ADD        NAME=IEFUJI
(IEFUJI object deck)
./  ADD        NAME=IEFUSI
(IEFUSI object deck)
./  ADD        NAME=IEFUSO
(IEFUSO object deck)
./  ADD        NAME=IEFUTL
(IEFUTL object deck)
./  ADD        NAME=IEFACTRT
(IEFACTRT object deck)
./  ENDUP
/*
```

Figure 7. Entering Exit Routines into EXITLIB

```
//PUNCH    JOB   MSGLEVEL=1
//         EXEC  PGM=IEBPTPCH
//SYSPRINT DD    SYSOUT=A
//SYSUT1   DD    DSNAME=SYS1.SAMPLIB,DISP=(OLD,KEEP),
//               UNIT=2314,VOLUME=SER=DLIB02                1
//SYSUT2   DD    UNIT=2540-2
//SYSIN    DD    *
      PUNCH    TYPORG=PO,MAXNAME=1,MAXFLDS=1
      MEMBER   NAME=TESTEXIT
      RECORD   FIELD=(80)
/*
```

[1] The volume on which SYS1.SAMPLIB resides depends upon the distribution package.

Figure 8. Obtaining a Punched Deck of TESTEXIT

TESTEXIT Execution Modifications

The third job (TESTING), shown earlier in Figure 6, includes execution of the TESTEXIT program. Values for the PARM parameter of the EXEC statement specify which exit routines are to be tested and the number of times each is to be tested. This parameter has the format PARM = 'xxx = nnn,...,xxx = nnn' where nnn is the number of times an exit routine is to be tested (maximum value 255), and xxx is an exit routine identifier. The DD statements to be included depend upon the exit routines to be tested.

Table 6 shows the exit-routine identifiers, specified on the EXEC statement, and the DD statements that must be included for each exit routine to be tested. DD statements for any other data sets used by your exit routines must be included in the TESTEXIT JCL.

Report Programs

Producing a report usually requires at least two operations—sorting the SMF records and writing them in an appropriate format. Your installation's requirements will determine what further analysis of SMF data is necessary.

Sorting SMF Records

You may use the IBM System/360 Operating System Sort/Merge Program (product number 5734-SM1) to sort SMF records. Note, however, that sort/merge does not process records that are less than 18 bytes long. Your report format and analysis requirements will determine the fields on which to sort and the sorting sequence. Sort exit E15 allows you to extract or delete selected records as the SMF dump data set is sorted, and sort exit E35 allows you to insert records into the final sorted output data set.

Table 6. Parameters and DD Statements for Executing TESTEXIT

Exit Routine	*Identifier*	*DD Statements*
IEFUJV	UJV	INUJV, OUTUJV
IEFUJI	UJI	INUJI, OUTUJI
IEFUSI	USI	INUSI, OUTUSI
IEFUSO	USO	INUSO, OUTUSO
IEFUTL	UTL	INUTL, OUTUTL
IEFACTRT	ACT	INACT, OUTACT
Any		MANX, SYSPRINT, DGPRINT, SYSABEND

Sample Sort Exit Routines

Three sample sort/merge exit routines are provided in SYS1.SAMPLIB: an E15 exit program (called SMFE15) and two sample E35 exit routines (called SMFE35A and SMFE35B). The SMFE15 routine is designed for use with either SMFE35A or SMFE35B. These programs are described below.

The SMFE15 routine extracts all non-job oriented SMF records (that is, records without a job log number) from the SMF dump data set. Dump header and dump trailer records (SMF record types 2 and 3) are retained in a temporary data set (DD name HDRDATA), and all other system-oriented records are retained on another temporary data set (DD name SORDATA). Only job-oriented records (that is, records having a job log number) are sorted.

The SMFE35A routine places in the sort output data set all records extracted by the SMFE15 exit routine. The system-oriented records precede the sorted job-oriented records in the output data set and are inserted in the following order: dump header records, dump trailer records, and all other system records.

The SMFE35B routine also places in the output data set the system-oriented records extracted by the SMFE15 exit routine. Dump header and dump trailer records are inserted before all job-oriented records. All other system records are inserted after the job records relating to the job being processed at the time the system records were written.

An example of the JCL required to execute the sort/merge program is provided in SYS1.SAMPLIB. This example has been extracted from the sample sort procedure described below. Figure 9 shows JCL that can be used to obtain a listing of sample sort exit routines and of sample sort JCL from SYS1.SAMPLIB.

If you plan to include the sample exit routines in your sort application, you must assemble and link-edit them before executing the sort/merge program. Figure 10 illustrates this procedure, including one possible sort application. In this example, SMF records are to be sorted first on the job log number (major control field), then on the date and time portions of the time stamp (minor control fields). Displacements of these fields (from the beginning of the physical record) are 19, 11, and 7 bytes, respectively.

```
//PRINT  JOB   123456,SMITH
//       EXEC  PGM=IEBPTPCH
//SYSPRINT DD  SYSOUT=A
//SYSUT1   DD  DSNAME=SYS1.SAMPLIB,DISP=(OLD,KEEP),
//             UNIT=2314,VOLUME=SER=DLIB02                  1
//SYSUT2   DD  SYSOUT=A
//SYSIN    DD  *
     PRINT    TYPORG=PO,MAXNAME=4,MAXFLDS=4
     MEMBER   NAME=SMFSORT
     RECORD   FIELD=(80)
     MEMBER   NAME=SMFE15
     RECORD   FIELD=(80)
     MEMBER   NAME=SMFE35A
     RECORD   FIELD=(80)
     MEMBER   NAME=SMFE35B
     RECORD   FIELD=(80)
/*
```

[1] The volume on which SYS1.SAMPLIB resides depends upon the distribution package.

Figure 9. Obtaining a Listing of Sample Sort Exit Routines

```
//SMFSORT  JOB  MSGLEVEL=1                                         1
//STEP1  EXEC  ASMFCL                                              1
//ASM.SYSIN DD *
    E15 SOURCE DECK
/*
//LKED.SYSLMOD DD DSNAME=SMF1.EXITS,VOL=REF=SYS1.LINKLIB,          2
//      DISP=(NEW,KEEP),SPACE=(TRK,(10,5,1))
//LKED.SYSIN DD *
 NAME   E15(R)                                                     3
/*
//STEP2  EXEC  ASMFCL                                              1
//ASM.SYSIN DD *
    E35 SOURCE DECK
/*
//LKED.SYSLMOD DD DSNAME=SMF1.EXITS,DISP=(OLD,KEEP),               2
//      VOL=SYS1.LINKLIB
//LKED.SYSIN DD *
 NAME   E35(R)                                                     3
/*
//SORTSTEP EXEC PGM=SORT,REGION=100K                               4
//SYSOUT DD SYSOUT=A
//SORTLIB  DD  DSNAME=SYS1.SORTLIB,DISP=SHR
//EXITLIB  DD  DSNAME=SMF1.EXITS,DISP=(OLD,KEEP),                  5
//      VOLUME=REF=SYS1.LINKLIB
//SORTIN   DD  UNIT=2400,VOLUME=SER=SYSMAN,DISP=OLD,               6
//      LABEL=(,NL),DCB=(RECFM=VBS,LRECL=600,BLKSIZE=200)          7
//SORTWK01 DD UNIT=2314,SPACE=(TRK,(50),,CONTIG)                   8
//SORTWK02 DD UNIT=2314,SPACE=(TRK,(50),,CONTIG)                   8
//SORTWK03 DD UNIT=2314,SPACE=(TRK,(50),,CONTIG)                   8
//SORTOUT  DD UNIT=2400,DSNAME=SMF1.SORTOUT,LABEL=(,NL),           9
//      DISP=(,KEEP),DCB=(RECFM=VBS,LRECL=600,BLKSIZE=200)         7
//SORDATA  DD  UNIT=SYSDA,SPACE=(CYL,(1,1)),                       10
//             DCB=(RECFM=VBS,LRECL=600,BLKSIZE=200)               7
//HDRDATA  DD  UNIT=SYSDA,SPACE=(TRK,(5,5)),                       10
//             DCB=(RECFM=VBS,LRECL=600,BLKSIZE=200)               7
//SYSIN  DD    *
 SORT  FIELDS=(19,16,A,11,4,A,7,4,A),FORMAT=BI,SIZE=E4000          11
 MODS  E15=(E15,700,EXITLIB,N),E35=(E35,1500,EXITLIB,N)            11
 END
/*
```

1 EXEC statement for cataloged procedure ASMFCL (assemble and link-edit).
2 The sample sort exit routines will be link-edited into data set SMF1.EXITS.
3 Linkage editor control statements specifying that E15 and E35 will be the load module names of the exit routines.
4 EXEC statement for the sort/merge program.
5 Data set SMF1.EXITS is specified as the library in which sort exit routines may be found.
6 Input to the sort program is the SMF dump data set, contained on a tape having a volume serial number of SYSMAN.
7 The BLKSIZE is shown as 200 bytes to represent half of the minimum SMF buffer size, which is also the default defined in the SMFDEFLT BUF parameter. The LRECL, shown as 600 bytes which covers most users, must be no less than the largest SMF record collected plus four bytes for the RDW. The LRECL may be larger than the BLKSIZE because the RECFM specifies spanned records. The user is expected to vary these parameters according to his SMF buffer and record length.
8 Three sort work units are defined as being direct-access devices.
9 The sort output data set is to be written on tape.
10 Two data sets required by the sample sort exit routines are defined on direct-access devices.
11 The sort/merge control statements define the sort control fields and exit routines to be used in this sort application.

Figure 10. Sample Sort Procedure

For a description of the cataloged procedure ASMFCL (assemble and link-edit) refer to *OS Assembler (F) Programmer's Guide*, GC26-3756. For a detailed discussion of the sort/merge program refer to *OS Sort/Merge*, GC28-6543.

Designing a Report Program

The basic operations of a report program are formatting and printing data from SMF records. The input to a report program is normally the sorted SMF dump data set. Your installation's report requirements will determine the amount of data modification, analysis, and formatting your report program must perform.

A sample source report program (SMFPOST), written in PL/I, is provided in SYS1.SAMPLIB. This program processes SMF record types 0 through 13, ignoring all other record types. Data from each input record is converted to an appropriate format and written as an output record. DECLARE, assignment, and PUT EDIT statements are used to format these records.

Figure 11 illustrates sample output from the SMFPOST program.

If you plan to modify the sample program or to write your own report program in PL/I, refer to the following publications for detailed information on PL/I: *OS PL/I Language*, GC28-8201, and *OS PL/I (F) Programmer's Guide*, GC28-6594.

```
RECTME       - RECORD TIME STAMP       - HH.TTT IS THE HOUR(HH) AND 1000THS/HOUR(TTT) WHEN THE RECORD WAS ORIGINATED
RT           - RECORD TYPE             - XX IS THE RECORD TYPE(XX)
JOBLOGNUMBER  - JOB LOG NUMBER         - CCCCCCCCHH.TTT IS THE JOB NAME(CCCCCCCC) AS SPECIFIED ON THE JOB CARD AND
                                        THE HOUR(HH) AND 1000THS/HOUR(TTT) JOB CARD WAS PROCESSED
SC           - STEP NUMBER             - XXX IS THE STEP NUMBER(XX) WITHIN THIS JOB (STEP TERMINATION RECORD)
             - TOTAL JOB STEPS         - XX IS THE TOTAL NUMBER OF STEPS WITHIN THIS JOB (JOB TERMINATION RECORD)
             - OUTPUT WRITER CLASS     - CC IS THE OUTPUT WRITER CLASS FOR THIS WRITER (OUTPUT WRITER RECORD)
STRTME       - STEP INITIATION         - HH.TTT IS THE HOUR(HH) AND 1000THS/HOUR(TTT) STEP INITIATED (STEP TERM. RECORD)
             - JOB INITIATION          - HH.TTT IS THE HOUR(HH) AND 1000THS/HOUR(TTT) JOB INITIATED (JOB TERM. RECORD)
             - OUTPUT WRITER START     - HH.TTT IS THE HOUR(HH) AND 1000THS/HOUR(TTT) WRITER STARTED (WRITER RECORD)
RDRSTP       - READER STOP TIME        - HH.TTT IS THE HOUR(HH) AND 1000THS/HOUR(TTT) READER STOPPED READING FOR THIS JOB
SYSIN        - SYSIN COUNT             - XXXXX IS THE NUMBER(XXXXX) OF LOGICAL RECORDS IN THE DD DATA OR DD * DATA
                                        SETS STEP TERMINATION RECORD - NUMBER OF RECORDS THIS STEP
                                        JOB TERMINATION RECORD - TOTAL NUMBER OF RECORDS THIS JOB
SYSOUT       - SYSOUT COUNT            - XXXXX IS THE NUMBER OF LOGICAL RECORDS FOR A WRITER PER JOB PER FORM NUMBER
                                        PER CLASS
P/PTIME      - PROB PROG CPU TIME      - SSSS.HH IS THE NUMBER OF SECONDS(SSSS) AND 1000THS/SECONDS(HH) OF PROB PROG
                                        EXECUTION
                                        STEP TERMINATION RECORD - STEP PROB PROG EXECUTION(CPU) TIME
                                        JOB TERMINATION RECORD - SUM OF STEP CPU TIMES FOR THIS JOB
PROGNAME     - PROGRAM NAME            - CCCCCCCC IS THE NAME(CCCCCCCC) OF THE PROGRAM EXECUTED BY THIS STEP
REG          - REQUESTED REGION SIZE   - XXX IS THE REGION SIZE(XXX) IN K BLOCKS REQUESTED BY A STEP.
                                        TAKEN FROM JOB OR EXEC STATEMENT OR FROM THE READER PROCEDURE DEFAULT
COR          - ACTUAL MAIN CORE USED   - XXX IS THE ACTUAL REGION CORE(XXX) IN K BLOCKS USED BY THIS STEP
JP           - JOB SELECTION PRIORITY  - XX IS THE PRIORITY(XX) WITH WHICH A JOB WAS SELECTED FOR INITIATION
SP           - STEP DISPATCH PRIORITY  - XXX IS THE ACTUAL PRIORITY(XXX) WITH WHICH A STEP WAS DISPATCHED
CODE         - COMPLETION CODE         - XXXX IS THE COMPLETION CODE(XXXX) WITH WHICH A JOB/STEP TERMINATED
RC/T         - READER CLASS/TYPE       - CCTT IS THE DEVICE CLASS(CC) AND UNIT TYPE(TT) OF THE SYSTEM READER FOR
                                        THIS JOB
FORM         - FORM NUMBER             - CCCC IS THE FORM NUMBER(CCCC) REQUESTED FOR THIS OUTPUT CLASS
DC           - DATA SET COUNT          - XX IS THE NUMBER(XX) OF DATA SETS PER OUTPUT CLASS PER FORM PROCESSED
                                        BY THE WRITER FOR THIS JOB
IODEVICE     - I/O DEVICE              - CCTT CUA IS THE DEVICE CLASS(CC), UNIT TYPE(TT), AND CHANNEL/UNIT ADDRESS (CUA)
EXCP         - EXCP COUNT              - XXXX IS THE NUMBER (XXXX) OF EXCP COUNTS PER DATA SET PER DEVICE
SWTIME       - SYSTEM WAIT TIME        - MMM.HH IS THE NUMBER OF MINUTES (MMM) AND 100THS/MINUTES (HH) OF CUMULATIVE
                                        SYSTEM WAIT TIME SINCE THE LAST SYSTEM WAIT PERIOD

             MEANING OF HEADING APPEARING ON RECORD TYPE 0 (IPL RECORD)
                                       SYSTEM          SYSTEM IDENTIFICATION
                                       MODEL           CPU MODEL NUMBER
                                       SMFBUF          ONE-HALF SMF SUFFER SIZE (BYTES)
                                       MAIN CORE (K)   MAIN STORAGE SIZE (1K BLOCKS)
                                       JWT             JOB WAIT TIME (MINUTES)
                                       SMFOPT          J - SYSTEM/JOB DATA COLLECTION
                                                       S - SYSTEM/STEP DATA COLLECTION
                                                       E - SMF EXITS REQUESTED
                                       DATE            DATE OF IPL (YYDDD)
------------------------------------------------------------------------------------------------------------------------

                                                                 DATE 690219      PAGE          1

RECTME   RT JOBLOGNUMBER SC STRTME   RDRSTP SYSIN SYOUT P/PTIME PROGNAME REG COR JP SP CODE RC/T FORM DC   IODEVICE EXCP SWTIME
15.698   0 SYSTEM=4J MODEL=40 JW=    0 SMFBUF= 200 MAIN CORE (K)= 512 SMFOPT=JSE DATE=69049
15.698   8                                 INITIAL I/O CONFIGURATION                                       0820  009
                                                                                                           0801  00C
                                                                                                           0808  00E
                                                                                                           4092  01A
                                                                                                           4092  01B
                                                                                                           2001  190
                                                                                                           2001  192
                                                                                                           2001  193
                                                                                                           8001  282
                                                                                                           8001  283
15.773   4 IPCT003815.732 1 15.737 ------     65 -----10.59 LINKEDIT 100 100   --   0 0000 ---- ---- --    2001  192  28
                                                                                                           2001  193   0
                                                                                                           2001  192 106
                                                                                                           2001  190  42
                                                                                                           2001  192   0
                                                                                                           2001  193 219
                                                                                                           2001  190  66
15.779   4 IPCT003815.732 2 15.737 ------     65 -----3.40 PBM=*.DD 100   74   --   0 0000 ---- ---- --    2001  192  28
                                                                                                           2001  190   0
                                                                                                           2001  190   2
```

Figure 11. Sample Output from PL/I Report Program

Associated with SMFPOST in SYS1.SAMPLIB is sample JCL for cataloged procedure PL1LFCL (compile and link edit). Figure 12 illustrates sample JCL for cataloged procedure PL1FCLG (compile, link edit, and execute). If you plan to use SMFPOST, you must substitute JCL appropriate to your application. Note that a compiler size option of 100K is necessary for successful compilation.

```
//A319P186 JOB PS24010101,MSGLEVEL=1,REGION=150K
//STEP1 EXEC PROC=PL1LFCLG,
//        PARM.PL1L='SIZE=100000,L,E,A,X,O,NT,DP'
//PL1L.SYSPUNCH DD SYSOUT=B
//PL1L.SYSIN DD *
//GO.SYSPRINT DD SYSOUT=Q,DCB=(LRECL=132,BLKSIZE=3828)
//GO.DATAIN DD UNIT 2400,LABEL=(,NL),
//        DCB=(RECFM=VBS,BLKSIZE=3625),
//        VOLUME=SER=(123456,AAAAAA,BBBBBB)
```

Figure 12. Sample JCL for Procedure PL1LFCLG

You may obtain a punched source deck or a listing of SMFPOST by using the IEBPTPCH utility program. Figure 13 shows JCL that can be used to obtain a source deck and listing of SMFPOST.

```
//PNCHPRNT JOB 123456,SMITH
//        EXEC  PGM=IEBPTPCH
//SYSPRINT  DD  SYSOUT=A
//SYSUT1    DD  DSNAME=SYS1.SAMPLIB,DISP=(OLD,KEEP),
//              UNIT=2314,VOLUME=SER=DLIB02             1
//SYSUT2    DD  UNIT=2540-2
//SYSIN     DD  *
     PUNCH  TYPORG=PO,MAXNAME=1,MAXFLDS=1
     MEMBER NAME=SMFPOST
     RECORD FIELD=(80)
/*
//        EXEC  PGM=IEBPTPCH
//SYSPRINT  DD  SYSOUT=A
//SYSUT1    DD  DSNAME=SYS1.SAMPLIB,DISP=(OLD,KEEP),
//              UNIT=2314,VOLUME=SER=DLIB02             1
//SYSUT2    DD  SYSOUT=A
//SYSIN     DD  *
     PRINT  TYPORG=PO,MAXNAME=1,MAXFLDS=1
     MEMBER NAME=SMFPOST
     RECORD FIELD=(80)
/*
```

[1] The volume on which SYS1.SAMPLIB resides depends upon the distribution package.

Figure 13. Obtaining a Source Deck and Listing of SMFPOST

Incorporating SMF into the System

SMF is specified as a system generation option. To add SMF to your operating system, you must include SMF in the system generation statements, define your use of SMF either in SMFDEFLT or from the operator's console, allocate storage for the SMF data set (if required), and add any user-written exit routines to the control program. The information needed to add SMF to the system is given in the topics that follow.

SYSGEN Procedure

Two system generation macro instructions are specifically related to SMF. Table 7 lists the macro instruction parameters used to specify SMF.

Table 7. Required SYSGEN Macro Instructions

Macro Instruction	*Parameter Required for SMF*
SCHEDULR	ACCTRTN = SMF
SUPRVSOR	TIMER = JOBSTEP

The other parameters you will supply for these macro instructions depend on your installation requirements. (For example, if you require a record type 21 to be written, the ESV parameter of the SCHEDULR macro instruction must specify SMF.) The value of the JOBQLMT parameter in the SCHEDULR macro instruction must be increased to include two additional messages per job and per job step. Additional system queue space must be specified in the CTRLPROG macro instruction, using the QSPACE parameter for MVT or the SYSQUE parameter for MFT. For a complete discussion of the system generation procedure, refer to *OS System Generation*, GC28-6554.

The member containing SMF options (SMFDEFLT) in SYS1.PARMLIB should be tailored to installation requirements before the first IPL of the generated system.

Including Exit Routines in the System

You should incorporate your exit routines into system library SYS1.CI505 before you generate your system; otherwise, they must be link-edited into the appropriate system load module.

Note: If an exit routine is link-edited into an active system, a link-edit failure may render inoperative the load modules into which the exit was being link-edited. This is particularly important in the case of termination exits because a failure would result in the loss of the output indicating the cause of failure.

Figure 14 illustrates the JCL required to add exit routines to SYS1.CI505 prior to system generation. If you do not require all exit routines in your system, simply omit the object deck and NAME control statement for those you do not need.

Figure 15 shows the required JCL for MVT Systems. Similar JCL is required for MFT systems, except that exit routines IEFUSO and IEFUTL must be link-edited into the nucleus.

Note: You must refer to your system generation listing for exact load module names and aliases.

```
//EXITLNK  JOB 123456,SMITH
//         EXEC  PGM=IEWL,PARM=Link-edit Parameters
//SYSPRINT DD   SYSOUT=A
//SYSLMOD  DD   DSNAME=SYS1.CI505,DISP=(OLD,KEEP)
//SYSUT1   DD   UNIT=SYSDA,DISP=(,DELETE),SPACE=(TRK,(20,5))
//SYSLIN   DD   *
```

(object deck for IEFUJV)

```
   NAME   IEFUJV(R)
```

(object deck for IEFUSO)

```
   NAME   IEFUSO(R)
```

(object deck for IEFUTL)

```
   NAME   IEFUTL(R)
/*
//         EXEC  PGM=IEWL,PARM=(DC,LET,LIST,RENT,NCAL)
//SYSPRINT DD   SYSOUT=A
//SYSLMOD  DD   DSNAME=SYS1.CI505,DISP=(OLD,KEEP)
//SYSUT1   DD   UNIT=SYSDA,DISP=(,DELETE),SPACE=(TRK,(20,5))
//SYSLIN   DD   *
```

(object deck for IEFUJI)

```
   NAME   IEFUJI(R)
```

(object deck for IEFUSI)

```
   NAME   IEFUSI(R)
```

(object deck for IEFACTRT)

```
   NAME   IEFACTRT(R)
/*
```

Figure 14. Adding Exit Routines to SYS1.CI505 Prior to System Generation

Link-editing parameters must be specified according to the characteristics of your exits.

Note: You must refer to your system generation listing for exact load module names and aliases.

```
//EXITLNK JOB 123456,SMITH
//         EXEC  PGM=IEWL,PARM=Link-edit Parameters
//SYSPRINT DD   SYSOUT=A
//SYSLMOD  DD   DSNAME=SYS1.LINKLIB,DISP=(OLD,KEEP)
//SYSUT1   DD   UNIT=SYSDA,DISP=(,DELETE),SPACE=(TRK,(20,5))
//SYSLIN   DD   *
```

(object deck for IEFUJV)

```
  ENTRY    IEFUJV
  INCLUDE  SYSLMOD(IEFUJV)
  NAME     IEFUJV(R)
```

(object decks for IEFUSO and IEFUTL)

```
  ENTRY    IEFSD263
  INCLUDE  SYSLMOD(IEFSD263)
  NAME     IEFSD263(R)
//         EXEC  PGM=IEWL,PARM=Link-edit Parameters
//SYSPRINT DD   SYSOUT=A
//SYSLMOD  DD   DSNAME=SYS1.LINKLIB,DISP=(OLD,KEEP)
//SYSUT1   DD   UNIT=SYSDA,DISP=(,DELETE),SPACE=(TRK,(20,5))
//SYSLIN   DD   *
```

(object decks for IEFUJI, IEFUSI, and IEFACTRT)

```
  ENTRY    IEFSD061
  INCLUDE  SYSLMOD(IEFSD061)
  ALIAS    Aliasname1,Aliasname2, . . .
  NAME     IEFSD061(R)
/*
```

Figure 15. Adding Exit Routines to SYS1.LINKLIB after System Generation in MVT

If you wish to add or replace exit routines after system generation, you must link-edit the routines into the appropriate load module. Table 8 shows the load module assignments for MVT and MFT.

Table 8. Required Load Module Assignments for Exit Routines

Object module for this exit routine	*Must be link-edited into this load module in MVT*	*Must be link-edited into this load module in MFT*[1]
IEFUJV	Scheduler load module IEFUJV.	Scheduler load module IEFUJV.
IEFACTRT	Scheduler load module IEFSD061.	Depends on the size of the MFT scheduler.[2]
IEFUJI	Scheduler load module IEFSD061.	Scheduler load module IEFSD512.
IEFUSI	Scheduler load module IEFSD061.	Scheduler load module IEFSD512.
IEFUSO	Scheduler load module IEFSD263.	Nucleus load module IEANUC01.
IEFUTL	Scheduler load module IEFSD263.	Nucleus load module IEANUC01.

[1] Load modules IEFUJV and IEFSD512 are link-edited with the attributes NCAL, LIST, LET, and XREF.

[2] The object module for IEFACTRT is link-edited into scheduler load module IEFSD510 for the 44K MFT scheduler; it is link edited into scheduler load modules IEFSD515 and IEFSD517 for the 30K MFT scheduler.

When adding exit routines after MVT system generation, as shown in Figure 15, the region size in the reader/interpreter must be increased to allow for exit routine IEFUJV; the region size specified in the initiator procedure must be increased to allow for exit routines IEFUJI, IEFUSI, and IEFACTRT; and the region size allocated to the link pack area must be increased to allow for exit routines IEFUSO and IEFUTL. The region size is specified in the MINPART parameter of the system generation SCHEDULR macro instruction.

When adding exit routines after system generation in either MVT or MFT, refer to your system generation listing for exact load module names aliases, and link-edit parameters.

Specifying SMFDEFLT Parameters

SMFDEFLT parameters, which control SMF operations, can be specified either (1) before the first IPL of a newly generated system by adding SMFDEFLT as a member in SYS1.PARMLIB or (2) at each initialization of SMF by entering SMFDEFLT parameters at the console at IPL time.

SMFDEFLT Contents and Format

SMFDEFLT consists of a series of parameters contained in 80-character, card-image records. Each parameter has the format:

keyword = {*value* | (*value*$_1$,*value*$_2$,...*value*$_n$)}

Consecutive parameters are separated by commas. No embedded blanks are permitted. Although parameters may be included in any order, you must conform to the following restrictions in coding SMFDEFLT parameters:

- Do not place parameters in columns 72 through 80; these columns are ignored.
- Indicate continuation of records by placing a non-blank character in column 72. (All records in the member except the last will include a continuation character.)
- A parameter cannot be divided between two records. Each set of parameter and value must be complete within a single record.
- Start your data in column 1 on the first record, and in column 16 on all continuation records.

Figure 16 shows the parameters, contained in card-image records, that make up a sample SMFDEFLT data set.

```
OPT=2,JWT=6,EXT=YES,BUF=2000,MDL=50,                                   X
               SID=4A,OPI=NO,MAN=USER,PRM=(,281,NL)
```

Figure 16. Sample SMFDEFLT Data Set

Following is a definition of the purpose and format of the SMFDEFLT parameters.

OPT Parameter

The OPT parameter is an optional parameter that specifies the type of system, job, and job-step information to be collected by SMF.

Note: If a job is recovered in a warm start, the setting of this parameter will be that setting in effect when the job is read in, rather than the setting done during the warm start IPL.

The format of the OPT parameter is:

```
OPT = {1|2}
```

where:

1

specifies that only system and job information is to be collected by SMF, that is, record type 4, which contains job-step information, is suppressed.

2

specifies that system, job, and job-step information is to be collected by SMF. If the OPT parameter is omitted, 2 is assumed.

Note: If OPT = 1 is specified, and if DSV = 2 or DSV = 3 is also specified, the value OPT = 2 is used instead of OPT = 1.

DSV Parameter

The DSV parameter is an optional parameter that specifies the type of data-set and direct-access volume information to be collected by SMF.

The format of the DSV parameter is:

```
DSV = {0|1|2|3}
```

where:

0

specifies that neither data-set nor direct-access volume information is to be collected by SMF, that is, record types 14 through 20, which contain data-set and direct-access volume information, are suppressed. If the parameter is omitted, 0 is assumed.

1

specifies that only direct-access volume information is to be collected by,SMF, that is, record types 14, 15, 17, 18, and 20, which contain data set information, are suppressed.

2

specifies that only data-set information is to be collected by SMF, that is, record type 19, which contains direct-access volume information, is suppressed.

3

specifies that both data-set information and direct-access volume information is to be collected by SMF.

Note: If data-set information, specified when DSV = 2 or DSV = 3, is requested, OPT = 2 is assumed, even if OPT = 1 was specified.

REC Parameter

The REC parameter is an optional parameter that specifies whether record type 17 will be written for temporary data sets.[1]

The format of the REC parameter is:

```
REC = {0|2}
```

where:

0

specifies that temporary data set information is not to be written (that is, record type 17 is not to be written for temporary data sets). If the parameter is omitted, 0 is assumed.

2

specifies that record type 17 is to be written for temporary data sets as well as for non-temporary data sets.

[1] The system generates a name when: (1) the DD statement for a new data set does not include the DSNAME parameter, or (2) when it contains a parameter in the form of DSNAME = & name or DSNAME = & & name.

EXT Parameter

The EXT parameter is an optional parameter that specifies whether the SMF exits are to be taken.

Note: If a job is recovered in a warm start, the setting of this parameter will be that setting in effect when the job is read in, rather than the setting done during the warm start IPL.

The format of the EXT parameter is:

```
EXT = {NO | YES}
```

where:

NO
specifies that exits are not to be taken.

YES
specifies that exits are to be taken. If the parameter is omitted, YES is assumed.

If EXT = YES is specified, the exits actually taken will depend on the data-collection parameter (OPT). If OPT = 2 is specified, all exits will be taken; if OPT = 1 is specified, job-step initiation and job-step termination exits will not be taken.

JWT Parameter

The JWT parameter is a required parameter that specifies the number of minutes a job is allowed to remain continuously in the wait state. When the specified limit has been reached, the time limit exit (IEFUTL) is entered if a user-written exit routine has been provided.

The format of the JWT parameter is:

```
JWT = n
```

where:

n
is a one- to three-digit decimal number that specifies, in minutes, the job wait time limit. The value specified must be 1 or greater.

BUF Parameter

The BUF parameter is an optional parameter that specifies the size of the SMF buffer. Buffer size requirements are discussed in "Main Storage Requirements" in the chapter "System Information and Requirements." If neither SMF records nor user records are to be recorded, this parameter is not required.

The format of the BUF parameter is:

```
BUF = n
```

where:

n
is a three- to five-digit decimal number defining the size (in bytes) of the SMF buffer. Minimum buffer size is 400 bytes; maximum buffer size is 65,536 bytes. If the value is not a multiple of 4, it is rounded to the next lower multiple of 4.

Note: You must dump the SMF data set(s) before you reduce the buffer size from the size specified at the previous IPL; otherwise, the SMF data set cannot be retrieved successfully.

SID Parameter

The SID parameter is a required parameter that identifies the system on which SMF is active.

Note: If a job is recovered in a warm start, the setting of this parameter will be that setting in effect when the job is read in, rather than the setting done during the warm start IPL.

The format of the SID parameter is:

```
SID = nn
```

where:

nn
is two alphameric characters identifying the system on which SMF is active.

Note: The information specified by the SID and MDL parameters is treated as one field in SMF records. Therefore, the last character of the SID parameter will be the first character of the MDL specification when the model being specified by MDL contains three digits. (See "MDL Parameter.")

MDL Parameter

The MDL parameter is a required parameter that defines the user-supplied model identifier of the system.

Note: If a job is recovered in a warm start, the setting of this parameter will be that setting in effect when the job is read in, rather than the setting done during the warm start IPL.

The format of the MDL parameter is:

MDL = *nn*

where:

nn
is two alphameric characters selected by the user to identify his system.

Note: The information specified by the SID and MDL parameters is treated as one field in SMF records. Therefore, if you need to specify three digits for the MDL parameter, enter the first digit as the last digit of the SID parameter. For example, if the system identification is "B" and the model is "145," specify SID = B1 and MDL = 45.

OPI Parameter

The OPI parameter is an optional parameter that specifies whether the operator will be permitted to modify SYS1.PARMLIB parameters from the console during IPL or when he issues a START TS or MODIFY TS command. The OPI parameter is ignored if it is entered from the console.

The format of the OPI parameter is:

OPI = {YES | NO}

where:

YES
specifies that the operator is allowed to modify parameters.

NO
specifies that the operator is not allowed to modify parameters. If the parameter is omitted, NO is assumed.

MAN Parameter

The MAN parameter is an optional parameter that specifies the type of records to be written to the SMF data set.

The format of the MAN parameter is:

MAN = {NONE | USER | ALL}

where:

NONE
specifies that no records are to be written to the SMF data set.

USER
specifies that only user records (from user-written exit routines) are to be written to the SMF data set, that is, record types 128 through 255 are written and record types 0 through127 are suppressed.

ALL
specifies that both SMF and user records are to be written to the SMF data set. If the parameter is omitted, ALL is assumed.

The MAN, OPT, and DSV parameters interact. If MAN = NONE, no records are written to the SMF data set, regardless of the values specified in the OPT and DSV parameters. If MAN = ALL, all SMF records are written, unless suppressed by the OPT or DSV parameter or by a user-written exit routine.

PRM Parameter

The PRM parameter is an optional parameter that specifies the primary SMF data set (SYS1.MANX). If no SMF or user records are to be written, this parameter is not required.

The format of the PRM parameter is:

PRM = ([*vol.ser.no.*,][*dev.add.*],**[NL | NSL | SL]**)

where:

vol.ser.no.
is one-to-six alphameric characters specifying the serial number of the direct-access volume on which the data set resides. If the data set is to reside on tape, a volume serial number is invalid; scratch volumes are mounted.

dev.addr.
is three alphameric characters specifying the address of the device on which the volume is mounted. If the data set is to reside on tape, the device address is required.

NL
indicates that the data set is on an unlabeled tape.

NSL
indicates that the data set is on a non-standard labeled tape.

SL
indicates that the data set is on a standard labeled tape. If no tape-label value is given, SL is assumed.

The absence of one of the values is indicated by a comma only if another value is included following the omitted value.

If the data set is defined on direct access, either the volume serial number or the device address, or both, may be specified.

If the data set is defined on tape, the device address must be specified and an IPL procedure is required after 20 volumes have been used. If a 7-track tape is used, the byte converter feature must be available; recording is at 800 bits per inch. If a dual density tape is used, recording is at 1,600 bits per inch.

ALT Parameter

The ALT parameter is an optional parameter that specifies the alternate SMF data set (SYS1.MANY) when the SMF data set resides on direct access. If the primary data set (SYS1.MANX) is defined on direct access, this data set must be defined on the same type of direct-access device. If no SMF or user records are to be written, or if the alternate data set is not used, this parameter is not required. If the primary data set is defined on tape, this parameter is ignored.

The format of the ALT parameter is:

ALT = (*vol.ser.no.,dev.addr.*)

where:

vol.ser.no.
is one-to-six alphameric characters specifying the serial number of the volume on which the data set resides.

dev.addr.
is three alphameric characters specifying the address of the device on which the volume is mounted.

Entering SMFDEFLT into SYS1.PARMLIB

When you have determined the parameters for SMFDEFLT and have them punched into a card deck, you should add it as a member of SYS1.PARMLIB by executing the utility program IEBUPDTE. Figure 17 illustrates the JCL required to execute the utility program. If your parameters change, you may replace the entire SMFDEFLT member with a new version by again executing IEBUPDTE, substituting a "./ REPL" control statement for the "./ ADD" control statement. The REPL statement will have the same operand as the ADD statement. For further information on the IEBUPDTE program, refer to *OS Utilities*, GC28-6586.

If operator intervention is allowed (OPI = YES), you may change SMFDEFLT parameter values from the operator's console during IPL. You may supply missing required parameters in the same manner. Messages are issued to notify the operator when intervention is required or permitted.

```
//ENTER  JOB   123456,SMITH
//        EXEC  PGM=IEBUPDTE,PARM=MOD
//SYSPRINT  DD SYSOUT=A
//SYSUT1  DD DSNAME=SYS1.PARMLIB,DISP=(OLD,KEEP)
//SYSUT2  DD DSNAME=SYS1.PARMLIB,DISP=(OLD,KEEP)
//SYSIN   DD DATA
./   ADD  LIST=ALL,NAME=SMFDEFLT,LEVEL=01,SOURCE=0
(SMFDEFLT data set)
/*
```

Figure 17. Entering SMFDEFLT into SYS1.PARMLIB

Note: If you do not add SMFDEFLT as a member of SYS1.PARMLIB, the parameters must be entered from the operator's console during IPL.

Specifying SMF Parameters for TSO

When SMF is present in a system that has Time Sharing Option (TSO), SMF information and exits are available for the TSO jobs. The SMF defaults for TSO jobs can be the background defaults or they can be separately specified. The options for TSO are separately specified by (1) adding the SMF default parameters to a member in SYS1.PARMLIB that can be referenced on a START TS command or (2) overriding the default parameters in a START TS or MODIFY TS command if operator intervention is specified in the SMFDEFLT member.

The format of the SMF options that can be specified for TSO is:

```
SMF = ({OFF | OPT = 1 | OPT = 2},{EXT = YES | EXT = NO})
```

where:

OFF
specifies that only TSO system records (types 30 through 33, 38, 41, and 42) are to be generated.

OPT = 1
specifies that TSO system and session records will be written and that job-level exits will be taken if EXT = YES is specified. TSO step records (types 34 and 40) are not to be written and step-level exits are not to be taken.

OPT = 2
specifies that TSO system, session, and step records are to be written. Both job and step-level exits are to be taken if EXT = YES is specified.

EXT = YES
specifies that user exits are to be taken from TSO jobs.

EXT = NO
specifies that user exits are not to be taken from TSO jobs, regardless of what is indicated in the OPT field.

The OPT and EXT parameters for TSO jobs are separate from the same parameters for background jobs. However, the options assigned for TSO cannot be more comprehensive than those specified for background jobs; for example, if job-step records are not written for background jobs, they are not written for TSO jobs.

Note: If DSV = 2 or DSV = 3 is specified for the background, OPT = 2 must be specified for the foreground.

Table 9 shows the interaction of values specified for background and TSO jobs. The values assigned for background jobs are not affected by values assigned for TSO jobs.

Table 9. OPT and EXT Values for TSO

Background Job Value	*TSO Job Value*	*TSO Value Assigned*
OPT = 1	OPT = 1	OPT = 1
OPT = 1	OPT = 2	OPT = 1
OPT = 2	OPT = 1	OPT = 1
OPT = 2	OPT = 2	OPT = 2
EXT = YES	EXT = YES	EXT = YES
EXT = YES	EXT = NO	EXT = NO
EXT = NO	EXT = YES	EXT = NO
EXT = NO	EXT = NO	EXT = NO

System Information and Requirements

SMF system requirements, performance, and operational considerations are the subjects of this chapter.

System Requirements

SMF can be specified as an option at system generation only in conjunction with the Multiprogramming with a Fixed number of Tasks (MFT) configuration or the Multiprogramming with a Variable number of Tasks (MVT) option of the operating system. SMF can be used with the Model 65 multiprocessing (M65MP) option of MVT and the Remote Job Entry (RJE) and Graphic Job Processor (GJP) options of MFT and MVT.

Main Storage Requirements

SMF requires additional main storage space for the resident nucleus, the system queue area, and, if exit routines are included, the region or partition size for system tasks.

Resident Nucleus

The main storage required for the resident nucleus is 1,700 bytes in MVT or 2,300 bytes plus 148 bytes for each partition in MFT. (Storage requirements for the job-step timing option, specified on the SUPRVSOR macro instruction, are also included in these estimates.) In MFT, a separate 4,096-byte area is also required for the SMF resident writer routine. In MVT, the writer routines reside in the master scheduler region.

Resident Reenterable Routines

In MFT, space is required for Basic Sequential Access Method modules. IGC019BA and IGC019BB must always be resident. If the SMF data sets are on direct-access volumes, IGC019CD must be resident; if the SMF data sets are on tape, IGC019CC must be resident. If the SMF data sets are on a direct-access device with rotational position sensing (RPS), IGG019CJ and IGG019FN must be resident.

System Queue Space

Main storage for system queue space is required for the Timing Control Table (TCT), SMF Control Table, the communication area for user-written exit routines, and the SMF buffer.

One TCT is created for each active job step. If only job accounting is required (OPT = 1), the size of each TCT is 96 bytes. If job-step accounting is required (OPT = 2), the size of each TCT is determined by the following formula:

$$\text{TCT} = 112 + 12(a) + 8(b)$$

where:

a
is the maximum number of DD statements per job step.

b
is the number of devices allocated because of each DD statement.

The SMF Control Table is a permanent table occupying 124 bytes.

The area used for communication between user-written exit routines is discussed under "Exit Routine Facilities and Restrictions" in the chapter "User-Written Routines."

To determine the SMF buffer size required, you must understand how the buffer is used. SMF records are blocked internally in variable-length format and, if necessary, spanned before they are written to the SMF data set. To allow overlapping of blocking and writing, the buffer size defined by the BUF parameter in SYS1.PARMLIB is divided into two equal parts; while one half of the buffer is being filled, the other half can be written. If records are to be written to the SMF data set, you must supply a buffer size in the SMFDEFLT data set before or during IPL. The minimum size is 400 bytes; the size specified should be twice the size of the largest record[1] to be written to the SMF data set to eliminate the need to span records. More than one record can be written in half a buffer if record sizes permit; if a record exceeds half a buffer, it is written as segments of a spanned record. To improve performance, the SMF buffer size should be defined to eliminate the need to span records.

[1] The Block Descriptor Word (four bytes) and the Record Descriptor Word (four bytes) should be included in the calculation of the largest record size.

Table 10 shows a list of buffer sizes and the corresponding number of average jobs that can be written on a single track of various direct-access devices. See "SMF Data Sets" later in this chapter for a sample tabulation of some of the SMF records and their sizes. Table 10 is based on an average job producing 1000 bytes of SMF data. This corresponds to the job used in calculations for Table 12 when not doing data set and volume activity recording. If calculations are needed for devices not shown refer to the device characteristics manual for the byte capacity. If you plan to reduce the size of the buffer during consecutive IPLs, dump the SMF data set(s) by using the SMF dump program (IFASMFDP); otherwise, the SMF data set cannot be retrieved successfully.

Table 10. SMF Buffer Size and Use of Direct Access Space

Buffer	*Physical Record*	*Jobs per Track of Various Devices*			
Size	*Length*	*2301*	*2303*	*2311*	*2314*
400	200	15.0	8.0	6.5	10.5
600	300	22.5	9.0	6.75	12.0
800	400	30.0	9.0	7.0	13.0
1000	500	30.75	10.0	7.5	13.75
1400	700	42.0	10.5	7.0	14.0
1800	900	42.75	9.0	6.75	13.5
2200	1100	44.0	10.0	8.25	13.75
2600	1300	45.5	9.75	6.5	13.0
3000	1500	45.0	11.25	7.5	15.0
4000	2000	45.0	10.0	5.0	15.0
6000	3000	45.0	7.5	7.5	15.0

Note: Figures have been adjusted to allow for interrecord gaps.

Exit Routines

If user-written exit routines are to be included, the region sizes specified for the reader/interpreter procedure, the initiator procedure, and the link pack area must be increased to accommodate the exit routines. The procedures for including user-written exit routines in the system are described in the chapter "Incorporating SMF into the System."

Auxiliary Storage Requirements

SMF requires auxiliary storage for the SMF data sets and the expansion of system libraries.

SMF Data Sets

The SMF data set must be permanently resident on a direct-access device or tape unit. If the SMF data set is on a direct-access device, it must be periodically dumped (by the SMF dump program) to a dump data set (for example, to magnetic tape), which can serve as input to user-written analysis and report routines. Several factors, such as specific system configuration, amount of SMF data to be written, and report program requirements, will determine which type of device is more efficient for a particular installation.

If the SMF data set is defined on a direct-access device, space must be allocated for the primary SMF data set (SYS1.MANX) and for an alternate SMF data set (SYS1.MANY). The two data sets need not be defined on the same physical device, but if two devices are used, they must be of the same type. Only device types on which system resident volumes are supported may be used, but, if possible, a device and channel other than those specified for SYSRES should be used.

The devices used for the data sets are defined in SMFDEFLT and become permanently resident at IPL. The devices must be online and ready during IPL. Space for the SMF data sets must be allocated on appropriate volume types; the volumes must be mounted prior to IPL.

Switching between the primary and alternate data sets is automatic as each becomes filled. The SWITCH or HALT command, however, can be used to switch between the data sets whenever you choose. The SMF dump program must be used to transfer a full data set to tape. When a dump is completed, the status of the data set is reset to empty by the dump program.

Note: If the system fails after a HALT or SWITCH command has been executed and before a buffer has been filled, the records in that buffer are lost. Similarly, if not enough records were written to fill a buffer, the block of records in the buffer is not written to the SMF data set. Whether a buffer is filled depends on the SMF buffer size and the length of the records being collected.

The space to be allocated to the SYS1.MANX and SYS1.MANY data sets depends on the amount of data generated by each of your average jobs and how often you want to dump the alternate data sets. The method for allocating space for SMF data sets on direct-access devices is shown under "Data Management Procedures" later in this chapter. Table 11 shows the size of some of the records that can appear in an SMF data set.

Table 11. SMF Record Sizes

Category of Data	Event or Status		Use Factor Definition			Record Type No.	Record Size (in Bytes)
			MAN=	OPT=	DSV=		
Day Data	IPL		ALL			0	27
	MFT Partition Definition		ALL			13	16 + 10 per partition
	Online Devices at IPL		ALL			8	16 + 4 per device
	End of day		ALL			12	18
Machine Data	Accumulated Wait Time		ALL			1	18
	SMF Records Lost		ALL			7	24
	Devices Varied Online		ALL			9	16 + 4 per device
	Devices Varied Offline		ALL			11	20
	Devices Recovered by Allocation		ALL			10	44
Secondary Storage Data	Space Available on DASD Volumes at IPL, HALT EOD, and when Demounted		ALL		1,3	19	64
Processing Data	Job Processing		ALL	1,2		5	116 + 1 per accounting data item + 1 per accounting data character
	Step Processing		ALL	2		4	109 + 8 per DD statement + 1 per accounting data item + 1 per accounting data character
	SYSOUT Processing		ALL			6	57
Data Set Activity Data	Initiation of a Job with Data Set Activity Recording		ALL		2,3	20	61 + 1 per accounting data item + 1 per accounting data character
	Closing, or EOV Processing of a Data Set	Data Set Opened for INPUT or RDBACK	ALL		2,3	14	264 + 24 per UCB + 28 for ISAM
		Data Set Opened for OUTPUT, UPDATE, INOUT, OUTIN	ALL		2,3	15	264 + 24 per UCB + 28 for ISAM
	Scratching of a Data Set		ALL		2,3	17	88 + 8 per data set scratched
	Renaming of a Data Set		ALL		2,3	18	132 + 8 per data set renamed
VS Data Management	Record Descriptor Word		ALL				4
	Block Descriptor Word		ALL				4

Table 12 is an example of how the space requirements for an entire data set can be established, given certain assumptions. Time sharing records, the ESV record, and record type 7 (Data Lost) are not shown in the example.

Table 12. Example of Data Set Space Requirements

Event or Status	Record Type	Assumption for This Example	Example No. of Bytes per Record	Example No. of Records	Example Total	
					per Job	per Alternate Data Set
IPL[1]	0	Once per day	27	1		27
Online Devices at IPL	8	20 devices, including 6 DASD	96	1		96
	19		64	6		384
End of day	12	Once per day	18	1		18
	19	6 DASD	64	6		384
Accumulated Wait Time	1	Once every 10 min. for 4 hours	18	24		432
Device Varied Online[2]	9	Twice per hour for 4 hours	20	8		160
Device Varied Offline[2]	11	Twice per hour for 4 hours	20	8		160
Device Recovered at Allocation	10	Once per hour for 4 hours	44	4		176
Scratch a Non-Temporary Data Set	17	Once per 4 hours, 1 volume per data set	96	1		96
Rename a Data Set	18	Once per 4 hours, 1 volume per data set	140	1		140
		Total for these records		61		2,073
Job Processing	5	Accounting data[3]	128	1	128	
	20		73	1	73	
	19	Demount 2 DASD volumes	64	2	128	
Step Processing	4	4 DD statements per step, Accounting Data[3], 3 steps per job	153	3	459	
	14	1 EOV processing, close 2 data sets per step, 3 steps per job	288	9	2,592	
	15	1 EOV processing, close 2 data sets per step, 3 steps per job	288	9	2,592	
SYSOUT Processing	6	2 output writers per step, 3 steps per job	31	6	186	
		Total for one job		31	6,158	
		Total for 12 jobs per hour for 4 hours 48 Jobs		1,200		295,584
		Total SMF Data				297,657
Record Descriptor Word			4	1,261		5,044
Block Descriptor Word		3 records per block	4	421		1,684
		Total number of bytes for this example				304,385

[1] If you are using SMF with MFT, record type 13 is created at IPL. The length of record type 13 is 16 bytes plus 10 bytes for each active partition.

[2] If you are using SMF with a model 65 multiprocessor in partitioned or multisystem mode, a corresponding type 9 or 11 record is produced each time you vary the CPU, storage, or channels.

[3] Accounting Data consists of two 5-byte items.

Note: If you are using SMF with MFT, record type 13 is created each time you redefine your partitions. The length of record type 13 is 16 bytes plus 10 bytes for each active partition.

When the SMF data set is resident on tape, it may be used directly as input to user-written report and analysis routines, without the intermediate use of the SMF dump program.

If the SMF data set is defined on magnetic tape, it may reside on as many as 20 tape volumes. The system IPL procedure must be repeated before the last volume is filled. At each IPL the volume count is reset to one and the tape is rewound; a new tape must be mounted before each IPL to avoid writing over previous SMF data. Only a primary data set (SYS1.MANX) is required. Specification of an alternate data set (SYS1.MANY) is unnecessary and is, therefore, ignored. The device must be on line and ready during IPL when it is allocated to SMF. The device is not available to other jobs in the system.

System Libraries

SMF requires direct-access device space for expansion of system libraries. Table 13 shows the space required for system libraries for MVT and MFT; these requirements, which are rounded up to the nearest track, are based on the track size of the 2314.

Table 13. Space Requirements for System Libraries

MVT	*MFT*
3 tracks for SYS1.LINKLIB	2 tracks for SYS1.LINKLIB
1 track for SYS1.SVCLIB	1 track for SYS1.SVCLIB
1 track for SYS1.NUCLEUS	1 track for SYS1.NUCLEUS
1 track for SYS1.PARMLIB	1 track for SYS1.PARMLIB

Performance

SMF will reduce system throughput by various amounts depending upon such factors as:

- SMF options selected, especially buffer size, SMF data set size, and SMF data set device.
- Execution times of user-written exit routines.
- System configuration, especially the type and degree of multiprogramming and the use of resident or transient modules.
- Job stream characteristics, such as number and type of jobs and user data set requirements. For example, a job stream consisting of short-running job steps causes more system overhead than a job stream consisting of longer-running job steps, especially in MFT.

Performance can be improved by making some modules resident. Table 14 shows the modules that should be resident for improved performance based on the SMF options selected.

Table 14. Improving SMF Performance

Option	*Resident Module*
MAN = ALL or MAN = USER	IGC0008C (SVC 83 first load)
DSV = 2 or DSV = 3	IFG0202H IFG0202I

Operational Considerations

Once SMF has been generated as part of a system, the system operator is responsible for operational procedures, such as the requirements for IPL and for dumping the SMF data set, and for any special procedures required by user-written report and analysis routines.

IPL Procedures

SMF initialization is the final step of the system IPL procedure. The SMF initialization program checks for the existence and validity of the SMFDEFLT member and for the availability of the devices on which data sets SYS1.MANX and SYS1.MANY are defined.

If SMFDEFLT has not been entered into SYS1.PARMLIB, the initialization program writes a message to the console, allowing you to enter SMFDEFLT parameters from the console. If the verification program encounters an input/output error while reading SMFDEFLT, it writes a message to the console, allowing you to either repeat the IPL procedure or enter SMFDEFLT parameters from the console.

If the initialization program finds SMFDEFLT and reads it without error, each parameter is checked for accuracy. If any parameters are incorrectly specified, messages are issued, allowing you to enter the correct parameter from the console. If all parameters are correctly specified and operator intervention allowed (OPI = YES) was specified in SMFDEFLT, messages are issued, allowing you to examine and modify the parameters.

When the initialization program has completed verification of SMFDEFLT, it checks for the existence of data sets SYS1.MANX and SYS1.MANY and for the availability of the devices on which they are defined. The initialization routine checks for the following conditions:

- The data set is not defined in SMFDEFLT.
- The specified device is not available.
- The data set is not allocated on the specified direct-access device.

If an error condition is found, a message is issued, indicating that no recording of SMF records is allowed until the condition is corrected and the IPL procedure is repeated.

If no errors are found, the initialization routine determines which data set (SYS1.MANX or SYS1.MANY) should receive SMF records. If neither data set contains data, SYS1.MANX receives the SMF records. If one data set contains data and the other one is empty, the partially full data set receives the SMF records. If both data sets already contain data, the data set with the lesser amount of data receives the SMF records. SMF data is maintained in chronological sequence if both SYS1.MANX and SYS1.MANY are defined on the same type of device and have the same space attributes, and if the data sets are dumped when called for by the control program.

When data-set verification has been completed without error, the initialization routine writes the IPL and input/output configuration records (SMF record types 0 and 8) to the specified data set, and gives control to the system so that processing of input streams may begin.

Data Management Procedures

Procedures for handling the SMF data set depend on the type of device on which the data set is resident. If the data set is resident on tape, you must mount a new tape prior to each IPL of the system. If the data set is resident on a direct-access device, you must allocate space on each specified device before IPL. If possible, a device and channel other than those specified for SYSRES should be used. Figure 18 illustrates sample DD statements for allocating space to the SMF data set.

Note: Specification of a secondary space allocation is ignored.

```
//MANX   DD    DSNAME=SYS1.MANX,UNIT=190,VOLUME=SER=111111,
//             SPACE=(TRK,(20)),DISP=(NEW,CATLG)
//MANY   DD    DSNAME=SYS1.MANY,UNIT=191,VOLUME=SER=222222,
//             SPACE=(TRK,(20)),DISP=(NEW,CATLG)
```

Figure 18. Allocating Space for SMF Data Sets on Direct Access Devices

SMF Dump Program

When the SMF data set is resident on a direct-access device, you must use the SMF dump program to transfer full SMF data sets to another data set (usually on tape) for analysis and to reset the dumped data set for possible reuse as the active SMF data set. Figure 19 shows the JCL required to execute the SMF dump program. The output is a non-temporary data set on a standard labeled tape. Have the operator record the volume serial number, because it is needed to reference this data set from another job. The SMF dump program transfers data in its original format. It uses the Basic Sequential Access Method to perform a physical copy from the input data set, DUMPIN, to the output data set, DUMPOUT. In the copy process, the program writes two SMF records to the output data set: a Dump Header record (record type 2) at the beginning of the data set and a Dump Trailer record (record type 3) at the end of the data set.

```
//DUMPX       JOB    201,,MSGLEVEL=1,PRTY=12
//STEP1       EXEC   PGM=IFASMFDP
//DUMPIN      DD     DSNAME=SYS1.MANX,DISP=OLD                    1
//DUMPOUT     DD     UNIT=TAPE,DISP=(NEW,KEEP)
//SYSPRINT    DD     SYSOUT=A
```

[1] If the alternate SMF data set, SYS1.MANY, is being dumped to tape, SYS1.MANY appears in the DSNAME parameter.

Figure 19. Executing the SMF Dump Program

Note: The SMF dump program writes messages, as required, to SYSOUT.

When either the primary (SYS1.MANX) or alternate (SYS1.MANY) data set is filled, a message is written to the console, requesting a dump. In no case should a data set that is being filled be dumped. If the invoked SMF dump program attempts to dump the active SMF data set, a message is printed at the console notifying the operator of his error; in response, the operator must cancel the dump program. He may then invoke the dump program to dump the correct SMF data set.

The SWITCH or HALT command can be used to switch between the SMF data sets so the previously active data set can be dumped. The format of the SWITCH command is:

{SWITCH|I} SMF

The format of the HALT command is:

{HALT|I} EOD

When either the SWITCH or HALT command is entered, the following actions occur:

1. A record type 19 is written for each direct-access device that is online, if DSV = 1 or DSV = 3 was specified.
2. A record type 12 is written.
3. The SMF data sets are switched or interchanged, that is, a different data set is designated to receive SMF records. If direct-access data sets are being used, the previously inactive data set is designated to receive SMF records. If tape data sets are being used, a request is given for a new tape to be mounted to receive SMF records. If the tape is a standard label tape, the mount request is for a tape with the same label. The same applies for non-label tape requests.

Both the HALT and SWITCH commands cause the CPU wait time accumulated up to the time of the command to be recorded in record type 12. Thus, record type 12 contains the wait time accumulated from the expiration of the ten-minute wait time interval reflected in the last record type 1 to the HALT or SWITCH command, and the next record type 1 contains the wait time accumulated from the HALT or SWITCH command to the expiration of the last ten-minute wait time interval.

The HALT command also causes other system actions, which are described in *OS Operator's Reference*, GC28-6691; the SWITCH command causes only the actions described here.

Note: SMF records are blocked internally in a buffer before they are written to the output data set. If the system fails after a HALT or SWITCH command and before a buffer has been filled, the records in that buffer are lost.

You may enter jobs specifying execution of the dump program into the system and hold them on the job queue until a dump is required. You may then release the appropriate job to dump the specified data set. Another method for executing the dump is to simply start a reader to an input stream containing the JCL for the dump program. A high priority should be assigned to the dump job to ensure immediate initiation.

If a data lost condition is present (both data sets are full), either data set can be dumped.

SMF Records

This chapter describes all of the records written to the SMF data set.

Standard SMF Record Header

Each record written to the SMF data set by SMF routines will contain the standard SMF record header. Each record written to the SMF data set by user-written routines should also include the standard record header. The length is 14 bytes.

The format is:

Decimal Displacement	*Hexadecimal Displacement*	*Field Size*	*Data Format*	*Contents*
0	0	1	binary	Reserved (zero)
1	1	1	binary	Record type
2	2	4	binary	Time in hundredths of a second
6	6	4	packed	Date in the form 00YYDDDF, where F is the sign
10	A	2	EBCDIC	System identification
12	C	2	EBCDIC	System model identifier

Note: When the SMFWTM macro instruction is used to write a record to the SMF data set, a record descriptor word (four bytes) must be added to the beginning of the record header; the address passed to the macro instruction must point to the beginning of the record descriptor word. For a discussion of the record descriptor word, refer to *OS Data Management Services Guide,* GC26-3746.

Record Type 0 (IPL)

Record type 0 is written after every IPL of the system. It includes the main storage size and the SMF options in effect. The length is 27 bytes.

The format is:

Decimal Displacement	*Hexadecimal Displacement*	*Field Size*	*Data Format*	*Contents*
0	0	1	binary	Reserved (zero)
1	1	1	binary	Record type 0
2	2	4	binary	Time record was moved to SMF buffer
6	6	4	packed	Date record was moved to SMF buffer
10	A	2	EBCDIC	System identification
12	C	2	EBCDIC	System model identifier
14	E	4	binary	Minutes of job wait time (from value specified on JWT parameter)
18	12	4	binary	Number of bytes in SMF buffer
22	16	4	binary	Number of 1K bytes in main storage
26	1A	1	binary	SMF options

Bit	*Meaning When Set*
0	System and job data to be collected
1	System, job, and step data to be collected
2	Exits requested
3	Data-set accounting
4	Volume accounting
5	Reserved
6	Temporary data set scratch records
7	Always zero

Record Type 1 (Wait Time)

Record type 1 is written after every IPL of the system and at the first job-step termination following the expiration of a ten-minute interval.[1] The length is 18 bytes.

Elapsed time (ordinary CPU-processing time) is divided into ten-minute intervals for the purpose of collecting wait time. A wait time record is written at SMF initialization and contains the CPU wait time accumulated during the IPL process. This record marks the beginning of the first ten-minute interval. Subsequent wait time records are written at the first job or job-step termination following the expiration of a ten-minute interval. Each wait time record contains the wait time accumulated during all the ten-minute intervals that expired between two job or step terminations. At job or step termination, a check is made to see whether at least one ten-minute interval has expired. Processing continues, as follows:

- If a ten-minute interval has expired, a wait time record is created and the wait time accumulated during the expired ten-minute interval is moved into the record.
- If more than one ten-minute interval has expired, the accumulated wait time collected during the expired ten-minute intervals since the last wait time record was created, moved into the record.
- If a ten-minute interval has not expired, no wait time record is created.

Note that the only connection between a job or step and wait time records is that the termination of a job or step causes SMF to check whether at least one ten-minute interval has expired.

Figure 20 shows how wait time is collected. When job/step A terminates, three ten-minute intervals have expired. The total wait time collected in these three intervals (783 seconds) is moved to a wait time record. When job/step B terminates, no ten-minute interval has expired since the last wait time record was written; therefore, a wait time record is not written. When job/step C terminates, three intervals have expired. The total wait time collected in these intervals (809 seconds) is moved to a wait time record.

Ten-Minute Intervals[1]

Wait Time Interval	217	263	303	342	265	202	360	220	388	362
Wait Time Collected[2] in a Record Type 1			783			809		580		
Job/Step Termination			A	B		C		D		

[1] Elapsed time is divided into ten-minute intervals by SMF and the wait time is collected in seconds for each successive interval.

[2] Collected by totaling the time found in each wait time interval completed before or at each job step termination. A record type 1 is written when a job/step terminates if a ten-minute interval has expired.

Figure 20. Wait Time Collection

Note: If the stop button is pushed to suspend CPU processing on an IBM System/370, (1) timing of the ten-minute interval is suspended, but (2) the Time-of-Day clock continues to run. The ten-minute interval is based on CPU-processing time, not on the Time-of-Day clock. Therefore, the wait time interval reflected by time of day (time stamp) is equal to the normal ten-minute CPU-processing interval plus the time that CPU processing was stopped.

There is a relationship among wait time, elapsed time, job time, and system time. The following formula shows that relationship:

Elapsed time = Job time+Wait time+System time

[1] If a HALT or SWITCH command is issued before the completion of a ten-minute interval, the wait time collected for that interval is written in a record type 12. If the system continues processing, the next record type 1 contains the wait-time accumulated from the HALT or SWITCH command to the expiration of the ten-minute interval.

Elapsed time is the length of the measurement interval. It can be obtained by calculating the difference between the time stamp on the first type 1 record and the time stamp on the type 12 record written when a HALT command or SWITCH command was processed after all jobs processed during the measurement interval have terminated.

Job time is the total time required by all jobs processed in the interval reflected by elapsed time. This value can be obtained by summing the CPU time values from all the type 5 records produced during the elapsed time.

Wait time is the total CPU wait time collected during the interval. This value can be obtained by summing the wait time values from the type 12 record written at the end of the interval and all but the first type 1 records.

System time is the total time required to process system tasks. This value can be calculated when the other three values are known.

The time stamp of a wait time record reflects the time at which the record was written, not the expiration time of the last ten-minute interval. The expiration time of the last ten-minute interval can be calculated using the time stamp of the first wait time record—the wait time record written at SMF initialization—as a base.

The format is:

Decimal Displacement	*Hexadecimal Displacement*	*Field Size*	*Data Format*	*Contents*
0	0	1	binary	Reserved (zero)
1	1	1	binary	Record type 1
2	2	4	binary	Time record was moved to SMF buffer
6	6	4	packed	Date record was moved to SMF buffer
10	A	2	EBCDIC	System identification
12	C	2	EBCDIC	System model identifier
14	E	4	binary	System wait time, in hundredths of a second, for ten-minute intervals that have expired since the last record type 1

Record Type 2 (Dump Header)

Record type 2 is written by the SMF dump program at the beginning of a dump data set. The length is 14 bytes.

This record consists of only the standard record header. It indicates the beginning of a dump of the SMF data set from a direct-access device to tape. Record type 2 is written directly to the dump data set by the SMF dump program.

The format is:

Decimal Displacement	*Hexadecimal Displacement*	*Field Size*	*Data Format*	*Contents*
0	0	1	binary	Reserved (zero)
1	1	1	binary	Record type 2
2	2	4	binary	Time, in hundredths of a second, record was written to the dump data set
6	6	4	packed	Date record was written to the dump data set, in the form 00YYDDDF, where F is the sign
10	A	2	EBCDIC	System identification
12	C	2	EBCDIC	System model identifier

Record Type 3 (Dump Trailer)

Record type 3 is written by the SMF dump program at the end of a dump data set. The length is 14 bytes.

This record consists of only the standard record header. It marks the end of an SMF dump to tape. Record type 3 is written directly to the dump data set by the SMF dump program.

The format is:

Decimal Displacement	*Hexadecimal Displacement*	*Field Size*	*Data Format*	*Contents*
0	0	1	binary	Reserved (zero)
1	1	1	binary	Record type 3
2	2	4	binary	Time, in hundredths of a second, record was written to the dump data set
6	6	4	packed	Date record was written to the dump data set, in the form 00YYDDDF, where F is the sign
10	A	2	EBCDIC	System identification
12	C	2	EBCDIC	System model identifier

Record Type 4 (Step Termination)

Record type 4 is written at the normal or abnormal termination of a job step or when a job step is flushed during or after interpretation. The length is variable.

The job step is identified by job log number (job name and reader start time for the job), step name, the number of the job step within the job, the user identification field (which may be initialized by the installation to facilitate subsequent sorting of records), and the program name. If accounting numbers were specified in the EXEC statement, they are included.

The record provides operating information such as the time the job step was started and completed, the time device allocation started, the time the problem program was loaded, the CPU time, the amount of main storage allocated and used, the storage protect key, and the termination status. Input/output activity is recorded for each data set used by the job step; each entry lists the device class, type, and address, and the EXCP count for the data set.

The EXCP count appears in SMF record types 4, 14, and 15. It indicates the input/output activity required by the job. The EXCP count includes direct EXCPs, program controlled interruptions (PCIs), and channel end and abnormal end EXCP returns. When chained scheduling is used, the EXCP count may vary from run to run for the same job. It may also vary up or down when chained scheduling is not used. This system function is designed to optimize input/output activity, and, therefore, the number of EXCPs required will depend on system and program interaction at the time the input/output is performed. The variation due to chained scheduling will be reflected in the counts for any data set using chained scheduling.

Data sets are recorded in the order of the step DD statements; they are not identified by name. A user-written exit routine can record this order as each statement is validated if a report on data set activity is needed. The number of records in SYSIN data sets for the step is also included.

The format is:

Decimal Displacement	*Hexadecimal Displacement*	*Field Size*	*Data Format*	*Contents*
0	0	1	binary	Reserved (zero)
1	1	1	binary	Record type 4
2	2	4	binary	Time of end of step
6	6	4	packed	Date of end of step
10	A	2	EBCDIC	System identification
12	C	2	EBCDIC	System model identifier
14	E	8	EBCDIC	Job name[1]
22	16	4	binary	Reader start time[1] in hundredths of a second
26	1A	4	packed	Reader start date[1]
30	1E	8	EBCDIC	User identification field from common parameter area
38	26	1	binary	Step number (first step = 1, etc.)
39	27	4	binary	Step initiation time in hundredths of a second, which is the time of day when this step was selected by the initiator
43	2B	4	packed	Step initiation date
47	2F	4	binary	Number of card-image records in DD DATA or DD * data sets
51	33	2	binary	Step completion code[2]
53	35	1	binary	Step priority[3]
54	36	8	EBCDIC	Program name
62	3E	8	EBCDIC	Name of executed step
70	46	2	binary	Region/partition size in 1K blocks allocated in hierarchy 0[4]
72	48	2	binary	Region/partition size in 1K blocks allocated in hierarchy 1[4]

Decimal Displacement	*Hexadecimal Displacement*	*Field Size*	*Data Format*	*Contents*
74	4A	4	binary	Hierarchy 0 storage used[5]
78	4E	4	binary	Hierarchy 1 storage used[5]
82	52	1	binary	Storage protect key (See TCBPKF in *OS System Control Blocks*, GC28-6628)
83	53	3	binary	Reserved
86	56	4	binary	Device allocation time in hundredths of a second
90	5A	4	binary	Problem program load time of day in hundredths of a second
94	5E	8	binary	Reserved
102	66		binary	Devices used by the step (one entry per device)[6]
		1	binary	Total length of next three fields
		3	binary	Step CPU time in hundredths of a second[7]
		1	binary	Number of accounting fields
				EXEC statement accounting fields[8]

[1] The job name, reader start time, and reader start date fields constitute the job log number.

[2] The contents of the completion code field varies according to the condition of termination, as follows:

X'0ccc',	which indicates system ABEND in the job step; ccc is the system ABEND code (see *OS Messages & Codes*, GC28-6631).
X'8ccc',	which indicates user ABEND in the job step; ccc is the user ABEND code.
X'nnnn',	which indicates normal completion; nnnn is the contents of the two low-order bytes in register 15 at termination.
X'0000',	which indicates either (1) that the job step was not executed because of an error in a preceding job step or (2) a return code of 0, indicating normal job step completion. To distinguish between a job step flush code and a normal termination code, examine the CPU time used by the job step. A job-step cancellation during allocation will not show the cancel ABEND code. The job-step is flushed. The CPU time is zero if the job step was flushed.

Abnormal or normal termination can be determined from the job termination indicator starting at byte 62 of record type 5.

[3] See *OS Supervisor Services*, GC28-6646.

[4] The amount of storage allocated to a program usually is the same as the amount of storage requested by the user. However, in MVT, if the amount of requested storage is less than the value assigned to the MINPART parameter, the MINPART value is allocated to the program.

[5] Bytes 0 and 1 indicate the storage used (in 1K blocks) within the specified region or partition. This value reflects the largest amount of storage required to execute the job. This does not include storage required for execution of system functions required to process the job. Bytes 2 and 3 indicate additional storage (in 1K blocks) allocated to the region. (Bytes 2 and 3 are used only in systems that include the rollout/rollin feature.)

[6] Bytes 0 and 1 contain the length of the field, including bytes 0 and 1. For each device assigned to each data set there is an eight-byte entry having the following format:

Byte 0—Device class from UCBTYP field of unit control block.

Byte 1—Unit type from UCBTYP field of unit control block.

Bytes 2,3—Channel and unit address.

Bytes 4-7—Count of EXCPs issued for the device and data set.

For a DD DUMMY data set the entry is set to 0. (A DD dummy entry results when a forward reference to a DD name is encountered in the input stream, but a DD statement having that DD name is not found.)

[7] CPU time is not expected to be constant between different runs of the same step. One or more of the following factors may cause small variations in CPU times: channel program retries, CPU architecture (such as core buffering), cycle stealing with integrated channels, queue searching (such as task switching), and pending interruptions.

[8] Each entry for an accounting field contains the length of the field (one byte, binary), followed by the field (EBCDIC). An omitted field is represented by a length indicator of 0.

Record Type 5 (Job Termination)

Record type 5 is written at the normal or abnormal termination of a job; the record is not written, however, if a job is canceled in the reader/initiator. The length is variable.

The job is identified by job log number, programmer name, the installation-supplied user identification field, input class, requested priority, and the accounting fields from the JOB statement. Operating information includes the start and stop time for processing of the job by the reader/interpreter, and the device type and class of the reader device. (The device type and class of the reader device is not provided for foreground-initiated background jobs.) The number of records in SYSIN data sets for the job and the number of steps in the job are included. Job CPU time equals the sum of the job-step CPU times. The job completion code is recorded, along with the storage protect key and a termination code indicating which SMF user-written exit routine, if any, canceled the job. A flag marks each SYSOUT class used by the job.

The format is:

Decimal Displacement	*Hexadecimal Displacement*	*Field Size*	*Data Format*	*Contents*
0	0	1	binary	Reserved (zero)
1	1	1	binary	Record type 5
2	2	4	binary	Time of end of job
6	6	4	packed	Date of end of job
10	A	2	EBCDIC	System identification
12	C	2	EBCDIC	System model identifier
14	E	8	EBCDIC	Job name[1]
22	16	4	binary	Reader start time[1] for job in hundredths of a second
26	1A	4	packed	Reader start date[1] for job
30	1E	8	EBCDIC	User identification field
38	26	1	binary	Number of steps in the job
39	27	4	binary	Job initiation time in hundredths of a second, which is the time of day the job was selected by the initiator
43	2B	4	packed	Job initiation date
47	2F	4	binary	Number of card-image records in DD DATA or DD * data sets
51	33	2	binary	Job completion code[2]
53	35	1	binary	Job priority[3]
54	36	4	binary	Reader stop time for job in hundredths of a second
58	3A	4	packed	Reader stop date for job
62	3E	1	binary	Job termination indicator *Bit* *Meaning When Set* 0 Reserved (0) 1 Reserved (0) 2 Canceled at exit IEFUJI 3 Canceled at exit IEFUSI 4 Canceled at exit IEFACTRT 5 Reserved (0) 6 0—Normal completion 1—ABEND 7 Reserved
63	3F	5	binary	SYSOUT class indicator[4]
68	44	1	binary	Checkpoint/restart indicator *Bit* *Meaning When Set* 0 System restart 1,2 Reserved 3 Checkpoint taken for step 4 Checkpoint restart 5 Step restart 6,7 Reserved (must be zero)

Decimal Displacement	*Hexadecimal Displacement*	*Field Size*	*Data Format*	*Contents*
69	45	1	binary	Reader device class from UCB[5]
70	46	1	binary	Reader unit type from UCB[5]
71	47	1	EBCDIC	Job input class
72	48	1	binary	Storage protect key (xxxx0000, where xxxx is the key which is described under TCBPKF in *OS System Control Blocks,* GC28-6628)
73	49	19	binary	Reserved
92	5C	1	binary	Length of rest of record (not including this field)
93	5D	20	EBCDIC	Programmer's name
113	71	3	binary	CPU time used by the job[6]
116	74	1	binary	Number of accounting fields following
117	75			JOB statement accounting fields (variable length)[7]

[1] The job name, reader start time, and reader start date fields constitute the job log number.

[2] The contents of the completion code field varies according to the condition of termination of the last step processed by the scheduler, as follows:

X'0ccc', which indicates system ABEND in the last job step; ccc is the system ABEND code (see *OS Messages & Codes*, GC28-6631).

X'8ccc', which indicates user ABEND in the last job step; ccc is the user ABEND code.

X'nnnn', which indicates normal completion; nnnn is the contents of the two low-order bytes in register 15 at termination.

X'0000', which indicates either (1) that the last job step was not executed because of an error in a preceding job step or (2) a return code of 0, indicating normal job completion. To distinguish between a job-flush code and a normal termination code, examine the CPU time used by the last job step shown in the corresponding type 4 record. The CPU time is zero if the last job step was flushed.

Abnormal or normal termination can be determined from the job termination indicator starting at byte 62 of this record.

[3] The job priority will be the same as the job's priority in the input work queue at the time the job was selected for execution.

[4] Each bit of the indicator represents the following classes:

Byte 0	*Byte 1*	*Byte 2*	*Byte 3*	*Byte 4*
Bit–Class	*Bit–Class*	*Bit–Class*	*Bit–Class*	*Bit–Class*
0–A	0–I	0–Q	0–Y	0–6
1–B	1–J	1–R	1–Z	1–7
2–C	2–K	2–S	2–0	2–8
3–D	3–L	3–T	3–1	3–9
4–E	4–M	4–U	4–2	
5–F	5–N	5–V	5–3	
6–G	6–O	6–W	6–4	
7–H	7–P	7–X	7–5	

Usually, this is the class specified by the SYSOUT parameter on the DD statement. For output data sets processed by the RJE output writer, this is the RJE SYSOUT class and not the SYSOUT class specified in the DD statement.

[5] The reader device class and reader unit type fields contain zeros for foreground-initiated background jobs.

[6] Time used for the problem program by the CPU between job initiation and job termination. This time includes the time used by, for example, the supervisor program; it excludes, however, the time used by, for example, the scheduler, reader, and writer programs. CPU time is not expected to be constant between different runs of the same job. One or more of the following factors may cause small variations in CPU times: channel program retries, CPU architecture (such as core buffering), cycle stealing with integrated channels, queue searching (such as task switching), and pending interruptions.

[7] Each entry for an accounting field contains the length of the field (one byte, binary), followed by the field (EBCDIC). An omitted field is represented by a length indicator of 0.

Record Type 6 (Output Writer)

Record type 6 is written when the writer has finished processing a SYSOUT class or form within a class for a job. At least one output writer record is written for each SYSOUT class used by the job. If two or more forms are used within a class, one output writer record is produced for each form. The length is 57 bytes.

The output writer is identified by class and form number. The job is identified by job log number, the installation user identification, and the time the job was read in. Output writer activity is recorded by a count of the number of logical records processed, the number of SYSOUT data sets within the class and form, writer start and end times, and a code indicating any input/output errors.

If the standard writer is replaced by a user-supplied writer, SMF does not produce a complete record type 6. An incomplete record type 6 is written for each output class (but not for form changes within an output class) if the writing of records is allowed. In this case, the number of logical records, I/O status indicators, and form number fields are not provided.

The format is:

Decimal Displacement	*Hexadecimal Displacement*	*Field Size*	*Data Format*	*Contents*
0	0	1	binary	Reserved (zero)
1	1	1	binary	Record type 6
2	2	4	binary	Time record was moved to SMF buffer
6	6	4	packed	Date record was moved to SMF buffer
10	A	2	EBCDIC	System identification
12	C	2	EBCDIC	System model identifier
14	E	8	EBCDIC	Job name[1]
22	16	4	binary	Reader start time[1]
26	1A	4	packed	Reader start date[1]
30	1E	8	EBCDIC	User identification field
38	26	1	EBCDIC	SYSOUT class
39	27	4	binary	Time of SYSOUT start
43	2B	4	packed	Date of SYSOUT start
47	2F	4	binary	Number of records written per form number per class[2]
51	33	1	binary	I/O status indicator *Bit* *Meaning When Set* 0-3 Reserved 4 I/O discontinued (remote output only) 5 Input error 6 Output error 7 Input error on SYS1.SYSJOBQE
52	34	1	binary	Total number of data sets processed by writer for this job
53	35	4	EBCDIC	Form number

[1] The job name, reader start time, and reader start date fields constitute the job log number.

[2] Line transmission errors may increase this count for jobs processed under the Remote Job Entry (RJE) option.

Record Type 7 (Data Lost)

Record type 7 is the first record written when an SMF data set becomes available after a period when no SMF data sets were available for recording. The length is 24 bytes.

This record contains a count of SMF records not written, and the start and end times of the period during which no records were written.

The format is:

Decimal Displacement	*Hexadecimal Displacement*	*Field Size*	*Data Format*	*Contents*
0	0	1	binary	Reserved (zero)
1	1	1	binary	Record type 7
2	2	4	binary	Time record was moved to SMF buffer
6	6	4	packed	Date record was moved to SMF buffer
10	A	2	EBCDIC	System identification
12	C	2	EBCDIC	System model identifier
14	E	2	binary	Number of SMF records lost
16	10	4	binary	Time of start of data loss
20	14	4	packed	Starting date at which no data set was available for recording SMF records

Record Type 8 (I/O Configuration)

Record type 8 is written after completion of IPL, following the SET DATE command. The length is variable.

This record consists of the standard record header and an entry describing each device that is on line at IPL. Devices are identified by device class, unit type, and device address.

The format is:

Decimal Displacement	*Hexadecimal Displacement*	*Field Size*	*Data Format*	*Contents*
0	0	1	binary	Reserved (zero)
1	1	1	binary	Record type 8
2	2	4	binary	Time record was moved to SMF buffer
6	6	4	packed	Date record was moved to SMF buffer
10	A	2	EBCDIC	System identification
12	C	2	EBCDIC	System model identifier
14	E		binary	Online I/O devices. Bytes 0 and 1 contain the length of the field. For each online device there is a four-byte entry having the following format:

Byte	*Contents*
0	Device class from UCBTYP field of unit control block
1	Unit type from UCBTYP field of unit control block
2	Channel address
3	Unit address

Record Type 9 (VARY ONLINE)

Record type 9 is written when a VARY ONLINE command is processed. The length is variable.

This record identifies the system resource being added to the configuration.

The format is:

Decimal Displacement	*Hexadecimal Displacement*	*Field Size*	*Data Format*	*Contents*
0	0	1	binary	Reserved (zero)
1	1	1	binary	Record type 9
2	2	4	binary	Time record was moved to SMF buffer
6	6	4	packed	Date record was moved to SMF buffer
10	A	2	EBCDIC	System identification
12	C	2	EBCDIC	System model identifier
14	E		binary	CPU, channel, device, or storage added. Bytes 0 and 1 contain the length of the field. For each element added there is a four-byte entry with the following format:

Vary CPU

Byte	*Contents*
0,1	2065 (hexadecimal)
2	00 (hexadecimal)
3	01 or 02 (hexadecimal) indicating CPU A or CPU B

Vary Channel

Byte	*Contents*
0,1	2860 or 2870 (hexadecimal) indicating channel type (multiplexer or selector)
2	00 to 06 (hexadecimal) indicating the channel number
3	01 or 02 (hexadecimal) indicating CPU A or CPU B

Vary Device

Byte	*Contents*
0	Device class from UCBTYP field of unit control block
1	Unit type from UCBTYP field of unit control block
2	Channel address
3	Unit address

Vary Storage (separate entries for Box 1 through Box 8)

Byte	*Contents*
0,1	2365 (hexadecimal)
2	Address index (a hexadecimal number between 0 and 80, which, when converted to decimal and multiplied by 2,048, gives the starting address of the storage, relative to the beginning of the box); zero if box is unaffected
3	Size index (a hexadecimal number between 0 and 80, which, when converted to decimal and multiplied by 2,048 gives the number of bytes of varied-online storage); zero if box is unaffected

Record Type 10 (Allocation Recovery)

Record type 10 is written after successful allocation. The length is variable.

This record identifies the device brought on line, or otherwise made available, by device class, unit type, and device address. The job requiring the allocation is identified by job name, reader start time, and the user identification field. The record is not produced if the operator cancels the job instead of attempting recovery.

The format is:

Decimal Displacement	*Hexadecimal Displacement*	*Field Size*	*Data Format*	*Contents*
0	0	1	binary	Reserved (zero)
1	1	1	binary	Record type 10
2	2	4	binary	Time record was moved to SMF buffer
6	6	4	packed	Date record was moved to SMF buffer
10	A	2	EBCDIC	System identification
12	C	2	EBCDIC	System model identifier
14	E	8	EBCDIC	Job name[1]
22	16	4	binary	Reader start time[1] in hundredths of a second
26	1A	4	packed	Reader start date[1]
30	1E	8	EBCDIC	User identification field
38	26		binary	Devices being made available. Bytes 0 and 1 contain the length of the field. For each device there is a four-byte entry having the following format: *Byte* *Contents* 0 Device class from UCBTYP field of unit control block 1 Unit type from UCBTYP field of unit control block 2 Channel address 3 Unit address

[1] The job name, reader start time, and reader start date fields constitute the job log number. If allocation recovery is for a system task, the job name field contains blanks and the reader start time and reader start date fields contain binary zeros.

Record Type 11 (VARY OFFLINE)

Record type 11 is written when a VARY OFFLINE command is processed. The length is variable.

The record identifies the system resource being removed from the configuration.

The format is:

Decimal Displacement	*Hexadecimal Displacement*	*Field Size*	*Data Format*	*Contents*
0	0	1	binary	Reserved (zero)
1	1	1	binary	Record type 11
2	2	4	binary	Time record was moved to SMF buffer
6	6	4	packed	Date record was moved to SMF buffer
10	A	2	EBCDIC	System identification
12	C	2	EBCDIC	System model identifier
14	E		binary	CPU, channel, device, or storage removed. Bytes 0 and 1 contain the length of the field. For each element removed, there is a four-byte entry with the following format:

Vary CPU

Byte	*Contents*
0,1	2065 (hexadecimal)
2	00 (hexadecimal)
3	01 or 02 (hexadecimal) indicating CPU A or CPU B

Vary Channel

Byte	*Contents*
0,1	2860 or 2870 (hexadecimal) indicating the channel type (multiplexer or selector)
2	00 to 06 (hexadecimal) indicating the channel number
3	01 or 02 (hexadecimal) indicating CPU A or CPU B

Vary Device

Byte	*Contents*
0	Device class from UCBTYP field of unit control block
1	Unit type from UCBTYP field of unit control block
2	Channel address
3	Unit address

Vary Storage (separate entries for Box 1 through Box 8)

Byte	*Contents*
0,1	2365 (hexadecimal)
2	Address index (a hexadecimal number between 0 and 80, which, when converted to decimal and multiplied by 2,048, gives the starting address of the storage relative to the beginning of the box); zero if box is unaffected
3	Size index (a hexadecimal number between 0 and 80, which, when converted to decimal and multiplied by 2,048, gives the number of bytes of varied-offline storage); zero if box is unaffected

Record Type 12 (End-of-Day)

Record type 12 is written when the HALT or SWITCH command is processed. The length is 18 bytes.

This record includes the system wait time since the last wait time record (record type 1).

The format is:

Decimal Displacement	*Hexadecimal Displacement*	*Field Size*	*Data Format*	*Contents*
0	0	1	binary	Reserved (zero)
1	1	1	binary	Record type 12
2	2	4	binary	Time record was moved to SMF buffer
6	6	4	packed	Date record was moved to SMF buffer
10	A	2	EBCDIC	System identification
12	C	2	EBCDIC	System model identifier
14	E	4	binary	System wait time,[1] in hundredths of a second, since wait time recorded in last type 1 record

[1] If the system continues running after the HALT or SWITCH command, the next type 1 record will contain the wait time accumulated from the HALT or SWITCH command to the expiration of the next ten-minute interval.

Record Type 13 (Dynamic Storage Configuration)

Record type 13, which applies only if you are using SMF with MFT, is written at IPL and after each DEFINE command is processed. It shows the amount of storage assigned to each active reader, writer, and problem program partition. The length is variable.

The format is:

Decimal Displacement	*Hexadecimal Displacement*	*Field Size*	*Data Format*	*Contents*
0	0	1	binary	Reserved (zero)
1	1	1	binary	Record type 13
2	2	4	binary	Time record was moved to SMF buffer
6	6	4	packed	Date record was moved to SMF buffer
10	A	2	EBCDIC	System identification
12	C	2	EBCDIC	System model identifier
14	E	2	binary	Number of bytes remaining, including this field
16	10			Storage assigned to each active partition. For each active reader, writer, or problem program partition, there is a ten-byte entry with the following format: *Byte* *Contents* 0 Partition number (1 byte, binary) 1,2 Hierarchy 0 storage in 1K blocks (2 bytes, binary) 3,4 Hierarchy 1 storage in 1K blocks (2 bytes, binary) 5 Number of job classes (1 byte, binary) 6-9 Readers, writers, or job classes assigned to this partition (4 bytes, EBCDIC)[1]

[1] If a reader or writer is assigned to the partition, byte 5 (number of job classes) will contain "1" and bytes 6 thru 9 will contain "ƀRDR" or "ƀWTR." If one job class is assigned to the partition, byte 5 will contain "1" and bytes 6 through 9 will contain the job class letter (A-O) right aligned and padded to the left with blanks. For example: "ƀƀƀA" or "ƀƀƀB." If more than one job class is assigned to the partition, byte 5 will contain the number of job classes (up to a maximum of 3) and bytes 6 through 9 will contain the job class letters in their specified order, right aligned and padded to the left with blanks. For example: "ƀƀAB" or "ƀABC."

Record Type 14 (INPUT or RDBACK Data Set Activity)

Record type 14 is written whenever a data set that is defined by a DD statement and opened for INPUT or RDBACK processing by a user program is closed or processed by EOV. The length varies from 288 to 6,412 bytes, depending upon the number of volumes for the data set.

This record contains the device type, EXCP count, data-set indicator, data-set organization, record format, record length, number of volumes, volume serial numbers, and additional information that depends on whether the data set is on a tape unit or a direct-access device and on the access method used.

Record type 14 is not written for a data set defined by a "DD *" or "DD DATA" statement. For accounting purposes, the card image count for these data sets is provided in record type 4.

The format is:

Decimal Displacement	*Hexadecimal Displacement*	*Field Size*	*Data Format*	*Contents*
0	0	1	binary	Reserved (zero)
1	1	1	binary	Record type 14
2	2	4	binary	Time record was moved to SMF buffer
6	6	4	packed	Date record was moved to SMF buffer
10	A	2	EBCDIC	System identification
12	C	2	EBCDIC	System model identifier
14	E	8	EBCDIC	Job name[1]
22	16	4	binary	Reader start time[1] in hundredths of a second
26	1A	4	packed	Reader start date[1]
30	1E	8	EBCDIC	User-identification field
38	26	2	binary	Record indicators *Bit* *Meaning When Set* 0 Reserved (0) 1 Record written by EOV 2 DASD device 3 Temporary data set 4 DCBDSORG = DA 5 DCBDSORG = IS 6 JFCDSORG = IS 7-15 Reserved (0)
40	28	4	binary	Segment sizes *Byte* *Contents* 0 Size of DCB/DEB segment 1 Number of UCB segments[2] 2 Size of each UCB segment 3 Size of extension segment
44	2C	4	binary	Reserved (zeros)
48	30	16		TIOT[3] segment—a portion of the TIOT including: *Byte* *Contents* 0 TIOELNGH 1 TIOESTTA 2 TIOEWTCT 3 TIOELINK 4 TIOEDDNM 12 TIOEJFCB 15 TIOESTTC
64	40	176		JFCB[3] segment—the entire JFCB, not including JFCB extensions

Decimal Displacement	*Hexadecimal Displacement*	*Field Size*	*Data Format*	*Contents*	
240	F0	24		DCB/DEB[3] segment	
				Byte	*Contents*
				0	DCBDSORG
				2	DCBRECFM
				3	DCBMACRF
				5	DCBOFLGS
				6	DCBOPTCD
				7	Reserved (0)
				8	DEBOFLGS
				9	DEBOPATB
				10	DEBVOLSQ
				Tape extension	
				12	DCBBLKCT
				16	Data set serial number
				22	Reserved (0)
				DASD extension	
				12	Relative track (TTR) of the last record processed for a physical sequential data set (left order three bytes followed by a byte of zeros); The value of this field is valid only for writing physical sequential data sets;[4] This field contains zeros for all data set organizations other than physical sequential
				16	Number of tracks released by the DADSM routine
				20	Number of extents released by the DADSM routine
				21	Reserved (0)
264	108	24		UCB[3] segment (24 bytes for each UCB in the data set)	
				Bytes	*Contents*
				0	UCBCHA
				1	UCBUA
				2	SRTEVOLI/DCELVOLI
				8	UCBTYP
				12	SRTESTAB/DCELSTAB
				13	Number of extents
				14	Reserved (0)
				16	EXCP count[5]
				Tape extension	
				20	SRTEFSCT
				22	SRTEFSEQ
				DASD extension	
				20	Total number of tracks allocated on the device
		28		ISAM Extension for DCBDSORG = IS	
				Byte	*Contents*
				0	Reserved (0)
				2	DCBMAC
				3	DCBNLEV
				4	DCBRORG3
				8	DCBNREC
				12	DCBRORG2
				14	DCBNOREC
				16	DCBRORG1
				18	Reserved (0)
				19	DEBNIEE
				20	DEBNPEE
				21	DEBNOEE
				22	Number of cylinders in Independent Index Area
				24	Number of cylinders in Prime Area
				26	Number of cylinders in Independent OVFL Area

[1] The job name, reader start time, and reader start date fields constitute the job log number.

[2] For ISAM data sets, the number of UCB segments in the order stated is one for the index extent, one per volume for primary extents, and one for the overflow extent.

For BPAM concatenated data sets used as input, there is one UCB segment for each data set in the concatenated data set.

[3] For further information about the contents of the TIOT, JFCB, DCB, DEB, and UCB, refer to *OS System Control Blocks*, GC28-6628.

[4] These conditions can be determined by interrogation of bytes 0 and 3 of the DCB/DEB segment.

[5] The EXCP count accumulates over the entire step. Therefore, if a data set is opened and closed twice during a single step, the count in the second record is the sum of all EXCPs for both uses of the data set. The EXCP count in the last type 14 record for the step is equal to the corresponding entry for the data set in the type 4 record.

Record Type 15 (OUTPUT, UPDAT, INOUT, or OUTIN Data Set Activity)

Record type 15 is written whenever a data set that is defined by a DD statement and opened for OUTPUT, UPDAT, INOUT, or OUTIN processing by a user program is closed or processed by EOV. The length varies from 288 to 6,412 bytes, depending upon the number of volumes for the data set.

This record contains the device type, EXCP count, data-set indicator, data-set organization, record format, record length, number of volumes, volume serial numbers, and additional information that depends on whether the data set is on a tape unit or a direct-access device and on the access method used.

Record type 15 is not written for data sets defined as SYSOUT data sets on DD statements. For accounting purposes, the SYSOUT logical record count is included in record type 6.

The format is:

Decimal Displacement	*Hexadecimal Displacement*	*Field Size*	*Data Format*	*Contents*
0	0	1	binary	Reserved (zero)
1	1	1	binary	Record type 15
2	2	4	binary	Time record was moved to SMF buffer
6	6	4	packed	Date record was moved to SMF buffer
10	A	2	EBCDIC	System identification
12	C	2	EBCDIC	System model identifier
14	E	8	EBCDIC	Job name[1]
22	16	4	binary	Reader start time[1] in hundredths of a second
26	1A	4	packed	Reader start date[1]
30	1E	8	EBCDIC	User identification field
38	26	2	binary	Record indicators *Bit* *Meaning When Set* 0 Reserved (0) 1 Record written by EOV 2 DASD device 3 Temporary data set 4 DCBDSORG = DA 5 DCBDSORG = IS 6 JFCDSORG = IS 7-15 Reserved (0)
40	28	4	binary	Segment sizes *Byte* *Contents* 0 Size of DCB/DEB segment 1 Number of UCB segments[2] 2 Size of each UCB segment 3 Size of extension segment
44	2C	4	binary	Reserved (zeros)
48	30	16		TIOT[3] segment—a portion of the TIOT including: *Byte* *Contents* 0 TIOELNGH 1 TIOESTTA 2 TIOEWTCT 3 TIOELINK 4 TIOEDDNM 12 TIOEJFCB 15 TIOESTTC
64	40	176		JFCB[3] segment—the entire JFCB, not including JFCB extensions

Decimal Displacement	*Hexadecimal Displacement*	*Field Size*	*Data Format*	*Contents*	
240	F0	24		DCB/DEB[3] segment	
				Byte	*Contents*
				0	DCBDSORG
				2	DCBRECFM
				3	DCBMACRF
				5	DCBOFLGS
				6	DCBOPTCD
				7	Reserved (0)
				8	DEBOFLGS
				9	DEBOPATB
				10	DEBVOLSQ
				Tape extension	
				12	DCBBLKCT
				16	Data set serial number
				22	Reserved (0)
				DASD extension	
				12	Relative track (TTR) of the last record processed for a physical sequential data set (left order three bytes followed by a byte of zeros); The value of this field is valid only for writing physical sequential data sets;[5] This field contains zeros for all data set organizations other than physical sequential
				16	Number of tracks released by the DADSM routine
				20	Number of extents released by the DADSM routine
				21	Reserved (0)
264	108	24		UCB[3] segment (24 bytes for each UCB in the data set)	
				Byte	*Contents*
				0	UCBCHA
				1	UCBUA
				2	SRTEVOLI/DCELVOLI
				8	UCBTYP
				12	SRTESTAB/DCELSTAB
				13	Number of extents
				14	Reserved (0)
				16	EXCP count[5]
				Tape extension	
				20	SRTEFSCT
				22	SRTEFSEQ
				DASD extension	
				20	Total number of tracks allocated on the device
		28		ISAM Extension—for DCBDSORG = IS	
				Byte	*Contents*
				0	Reserved (0)
				2	DCBMAC
				3	DCBNLEV
				4	DCBRORG3
				8	DCBNREC
				12	DCBRORG2
				14	DCBNOREC
				16	DCBRORG1
				18	Reserved (0)
				19	DEBNIEE
				20	DEBNPEE
				21	DEBNOEE
				22	Number of cylinders in Independent Index Area
				24	Number of cylinders in Prime Area
				26	Number of cylinders in Independent OVFL Area

[1] The job name, reader start time, and reader start date fields constitute the job log number.

[2] For ISAM data sets, the number of UCB segments in the order stated is one for the index extent, one per volume for primary extents, and one for the overflow extent.

[3] For further information about the contents of the TIOT, JFCB, DCB, DEB, and UCB, refer to *OS System Control Blocks,* GC28-6628.

[4] These conditions can be determined by interrogation of bytes 0 and 3 of the DCB/DEB segment.

[5] The EXCP count accumulates over the entire step. Therefore, if a data set is opened and closed twice during a single step, the count in the second record is the sum of all EXCPs for both uses of the data set. The EXCP count in the last type 15 record for the step is equal to the corresponding entry for the data set in the type 4 record.

Record Type 17 (Scratch Data Set Status)

Record type 17 is written whenever a user data set is scratched. (The REC parameter determines whether record type 17 is created for non-temporary data sets only or for both temporary and non-temporary data sets.) The length varies from 96 to 2,136 bytes, depending upon the number of volumes for the data set.

This record contains the data-set name, number of volumes, and volume serial numbers.

The format is:

Decimal Displacement	*Hexadecimal Displacement*	*Field Size*	*Data Format*	*Contents*
0	0	1	binary	Reserved (zero)
1	1	1	binary	Record type 17
2	2	4	binary	Time record was moved to SMF buffer
6	6	4	packed	Date record was moved to SMF buffer
10	A	2	EBCDIC	System identification
12	C	2	EBCDIC	System model identifier
14	E	8	EBCDIC	Job name[1]
22	16	4	binary	Reader start time[1] in hundredths of a second
26	1A	4	packed	Reader start date[1]
30	1E	8	EBCDIC	User-identification field
38	26	2	binary	Reserved (zero)
40	28	44	EBCDIC	Data-set name
84	54	3	binary	Reserved (zero)
87	57	1	binary	Number of volumes
88	58			Volume serial number (eight bytes for each volume) *Byte* *Contents* 0-1 Reserved (binary 0) 2-7 Volume serial number (EBCDIC)

[1] The job name, reader start time, and reader start date fields constitute the job log number.

Record Type 18 (Rename Data Set Status)

Record type 18 is written whenever any data set is renamed. The length varies from 140 to 2180 bytes, depending upon the number of volumes for the data set.

This record contains the old data-set name, new data-set name, number of volumes, and volume serial numbers.

The format is:

Decimal Displacement	*Hexadecimal Displacement*	*Field Size*	*Data Format*	*Contents*
0	0	1	binary	Reserved (zero)
1	1	1	binary	Record type 18
2	2	4	binary	Time record was moved to SMF buffer
6	6	4	packed	Date record was moved to SMF buffer
10	A	2	EBCDIC	System identification
12	C	2	EBCDIC	System model identifier
14	E	8	EBCDIC	Job name[1]
22	16	4	binary	Reader start time[1] in hundredths of a second
26	1A	4	packed	Reader start date[1]
30	1E	8	EBCDIC	User-identification field
38	26	2	binary	Reserved (zero)
40	28	44	EBCDIC	Old data-set name
84	54	44	EBCDIC	New data-set name
128	80	3	binary	Reserved (zero)
131	83	1	binary	Number of volumes
132	84			Volume serial number (eight bytes for each volume) *Byte* *Contents* 0-1 Reserved (binary 0) 2-7 Volume serial number (EBCDIC)

[1] The job name, reader start time, and reader start date constitute the job log number.

Record Type 19 (Direct-Access Volume)

Record type 19 is written for all direct-access devices that are on line at IPL, when a HALT EOD or SWITCH SMF command is processed, and when a user volume is demounted. The length is 64 bytes.

Note: Record type 19 is not created for DOS volumes used under the operating system.

This record contains the volume serial number, VTOC address, owner identification number, device type, number of unused alternate tracks, number of unallocated cylinders and tracks, number of cylinders and tracks in the largest free extent, number of unallocated extents, channel and unit address, and module identification for the 2314 and 3330.

Note: Synchronization of clocks is essential in a shared file environment.

The format is:

Decimal Displacement	*Hexadecimal Displacement*	*Field Size*	*Data Format*	*Contents*
0	0	1	binary	Reserved (zero)
1	1	1	binary	Record type 19
2	2	4	binary	Time record was moved to SMF buffer
6	6	4	packed	Date record was moved to SMF buffer
10	A	2	EBCDIC	System identification
12	C	2	EBCDIC	System model identifier
14	E	8		Volume serial number *Byte* *Contents* 0,1 Reserved (binary 0) 2-7 Volume serial number (EBCDIC)
22	16	10	EBCDIC	Owner identification
32	20	4	binary	Device type
36	24	5	binary	VTOC address
41	29	1	binary	DS4VTOCI
42	2A	2	binary	Number of DSCBs
44	2C	2	binary	Number of format 0 DSCBs
46	2E	2	binary	Number of unused alternate tracks
48	30	2	binary	Number of unallocated cylinders
50	32	2	binary	Number of unallocated tracks
52	34	2	binary	Number of cylinders in the largest free extent
54	36	2	binary	Number of tracks in the largest free extent
56	38	2	binary	Number of unallocated extents
58	3A	2	binary	Reserved (zero)
60	3C	2	binary	Channel and unit address in the form 0cuu, where c is the channel address and uu is the unit address
62	3E	2	binary	Module identification for the 2314 and 3330, which is taken from bits 2-7 of sense byte 4 for these devices (Refer to the component descriptions of these devices for the meaning of sense byte 4.)

Record Type 20 (Job Commencement)

Record type 20 is written at job initiation when data-set accounting and/or direct-access volume information is specified. The length is variable.

This record contains the record type, time stamp (time and date), CPU identification, job log number (job name, entry time, and entry date), programmer's name, user identification, number of accounting fields on the JOB statement, and accounting fields.

The format is:

Decimal Displacement	*Hexadecimal Displacement*	*Field Size*	*Data Format*	*Contents*
0	0	1	binary	Reserved (zero)
1	1	1	binary	Record type 20
2	2	4	binary	Time record was moved to SMF buffer
6	6	4	packed	Date record was moved to SMF buffer
10	A	2	EBCDIC	System identification
12	C	2	EBCDIC	System model identifier
14	E	8	EBCDIC	Job name[1]
22	16	4	binary	Reader start time[1] in hundredths of a second
26	1A	4	packed	Reader start date[1]
30	1E	8	EBCDIC	User-identification field
38	26	2	binary	Reserved (zero)
40	28	20	EBCDIC	Programmer's name
60	3C	1	binary	Number of accounting fields
61	3D			Accounting fields[2]

[1] The job name, reader start time, and reader start date fields constitute the job log number.

[2] Each entry for an accounting field contains the length of the field (one byte, binary), followed by the field (EBCDIC). An omitted field is represented by a length indicator of 0.

Record Type 21 (ESV)

Record type 21 is written by the Error Statistics by Volume (ESV) option when a user data set on magnetic tape is closed or processed by End-of-Volume. This information can be used by IFHSTATR or by user-written routines that address problems of volume deterioration (defective tracks) and space fragmentation. (See "IFHSTATR" in *OS/VS Utilities*, GC35-0005.) This record is written to the SMF data set only if ESV = SMF is specified in the SCHEDULR macro instruction at system generation. The length is 44 bytes.

This record contains error statistics information about the tape volume.

The format is:

Decimal Displacement	*Hexadecimal Displacement*	*Field Size*	*Data Format*	*Contents*
0	0	1	binary	Reserved (zero)
1	1	1	binary	Record type 21
2	2	4	binary	Time record was moved to buffer
6	6	4	packed	Date record was moved to buffer
10	A	2	EBCDIC	System identification
12	C	2	EBCDIC	System model identifier
14	E	2	binary	Length of this record
16	10	6	EBCDIC	Volume serial number
22	16	2	binary	Channel/unit address
24	18	4	binary	UCB type
28	1C	1	binary	Number of temporary read errors
29	1D	1	binary	Number of temporary write errors
30	1E	2	binary	Number of start I/Os
32	20	1	binary	Number of permanent read errors
33	21	1	binary	Number of permanent write errors
34	22	1	binary	Number of noise blocks
35	23	2	binary	Number of erase gaps
37	25	2	binary	Number of cleaner actions
39	27	1	binary	Tape density (Format of this field is the same as the DCBDEN field)
40	28	2	binary	Block size
42	2A	2		Reserved (not necessarily zero)

Record Type 30 (Start TS)

Record type 30 is written each time the time sharing option is started with a START TS command. The length varies, depending upon the number of time sharing regions.

This record contains the record type, time stamp (time and date), CPU identification, time sharing initiation procedure name, time sharing task identifier, time sharing member name in SYS1.PARMLIB, SMF foreground options, maximum number of TS regions, TSC region size, time sharing driver name, number of terminals allowed, number of time sharing regions, size of each time sharing region, and LSQS size in each time sharing region.

The format is:

Decimal Displacement	*Hexadecimal Displacement*	*Field Size*	*Data Format*	*Contents*
0	0	1	binary	Reserved (zero)
1	1	1	binary	Record type 30
2	2	4	binary	Time record was moved to SMF buffer
6	6	4	packed	Date record was moved to SMF buffer
10	A	2	EBCDIC	System identification
12	C	2	EBCDIC	System model identifier
14	E	8	EBCDIC	Time sharing procedure name
22	16	8	EBCDIC	Time sharing task identifier
30	1E	8	EBCDIC	Member name in SYS1.PARMLIB
38	26	1	binary	SMF foreground options *Bit* *Meaning When Set* 0 System and job data to be collected 1 System, job, and step data to be collected 2 Exits requested 3 Data-set accounting 4 Volume accounting 5 Reserved 6 Temporary data set scratch records 7 Always 1
39	27	1	binary	Maximum number of TS regions
40	28	2	binary	TSC region size (in 2K blocks)
42	2A	8	EBCDIC	Time sharing driver name
50	32	2	binary	Maximum number of TS terminals allowed
52	34	26	binary	Reserved (zero)
78	4E	2	binary	Length of rest of record (including these two bytes)
80	50	1	binary	Number of TS regions

One of the following 5-byte entries for each TS region.

		1	binary	Region number
		2	binary	LSQS size (in 2K blocks)
		2	binary	Region size (in 2K blocks)

Record Type 31 (TIOC Initialization)

Record type 31 is written each time the TIOC initialization routine is entered by the time sharing control (TSC) task as the result of a START TS command. The length is 54 bytes.

This record contains the record type, time stamp (time and date), CPU identification, total number of time sharing buffers, size of time sharing buffers, maximum number of output buffers allowed each terminal before OWAIT (program wait for output buffers), and maximum number of input buffers allowed each terminal before LWAIT (terminal lockup). It also contains the OWAIT threshold (the number of buffers that must be freed in order to be freed from OWAIT), RESTART threshold (the number of buffers that must be freed in order to be freed from LWAIT), number of buffers reserved on the free queue, number of users that constitute slack time, size of one terminal sharing block, and logged-on user change.

The format is:

Decimal Displacement	*Hexadecimal Displacement*	*Field Size*	*Data Format*	*Contents*
0	0	1	binary	Reserved (zero)
1	1	1	binary	Record type 31
2	2	4	binary	Time record was moved to SMF buffer
6	6	4	packed	Date record was moved to SMF buffer
10	A	2	EBCDIC	System identification
12	C	2	EBCDIC	System model identifier
14	E	2	binary	Total number of buffers
16	10	2	binary	Buffer size
18	12	2	binary	Reserved (zero)
20	14	2	binary	Maximum number of output buffers allowed per terminal before OWAIT[1]
22	16	2	binary	Maximum number of input buffers allowed per terminal before LWAIT[2]
24	18	2	binary	OWAIT[1] threshold (The number of buffers that must be freed in order to be freed from OWAIT)
26	1A	2	binary	RESTART threshold (the number of buffers that must be freed in order to be freed from LWAIT[2])
28	1C	2	binary	Number of buffers reserved on the free queue
30	1E	2	binary	Number of users that constitute slack time
32	20	1	binary	Size of one terminal status block
33	21	1	binary	Logged-on user change (indicates when maximum number of output buffers allowed per terminal before OWAIT[1] and LWAIT[2] are to be recalculated; user specifies this in START TS command)
34	22	20	binary	Reserved (zero)

[1] OWAIT is the suspension of the program during input/output to the terminal because no output buffers are available.

[2] LWAIT is the locking up of the terminal user's keyboard because he has filled all the input buffers available to him.

Record Type 32 (Driver)

Record type 32 is written each time the driver initialization routine is entered by the TSC task as a result of a START TS command. The length varies, depending upon the number of subqueues for each region.

This record contains the record type, time stamp (time and date), CPU identification, driver control flags, guaranteed background execution percentage, wait estimate constant, and region estimate constant. For each subqueue the following information is recorded: the region number, number of service cycles to be given to the users on the subqueue, amount of storage allowed to users on the subqueue, maximum main storage occupancy time allowed to users on the subqueue, average service time for users on the subqueue, and the minimum time slice to be given to a user on the subqueue.

The format is:

Decimal Displacement	*Hexadecimal Displacement*	*Field Size*	*Data Format*	*Contents*
0	0	1	binary	Reserved (zero)
1	1	1	binary	Record type 32
2	2	4	binary	Time record was moved to SMF buffer
6	6	4	packed	Date record was moved to SMF buffer
10	A	2	EBCDIC	System identification
12	C	2	EBCDIC	System model identifier
14	E	1	binary	Driver control flags—specify which fields are to be ignored by the driver *Bit* *Ignore When Set* 0 Wait estimate 1 Region activity 2 Main storage occupancy 3 Swap load 4 Average queue service time 5 Background to foreground ratios 6 Scheduling of priority 7 Current RQEL
15	F	1	binary	Percentage of CPU time to be given to background jobs
16	10	4	binary	Constant used to determine wait estimate
20	14	4	binary	Constant used to determine region estimate
24	18	2	binary	Length of rest of record

One of the following 20-byte areas exists for each subqueue for each region.

		1	binary	Reserved
		1	binary	Region number
		2	binary	Number of service cycles to be given to subqueue
		2	binary	Amount of storage in 2K blocks allowed to user on subqueue
		4	binary	Maximum main storage occupancy time (in hundredths of a second) allowed to user on subqueue
		4	binary	Average service time (in hundredths of a second) for user on subqueue
		4	binary	Minimum time slice (in hundredths of a second) for user on subqueue
		2	binary	Reserved (zero)

The remainder of the record contains the following field.

		20	binary	Reserved (zero)

Record Type 33 (Driver Modify)

Record type 33 is written each time the driver modify routine is entered as a result of a modify driver command. The length is 16 bytes.

This record contains the record type, time stamp (time and date), CPU identification, driver control flags, and background execution percentage.

The format is:

Decimal Displacement	*Hexadecimal Displacement*	*Field Size*	*Data Format*	*Contents*
0	0	1	binary	Reserved (zero)
1	1	1	binary	Record type 33
2	2	4	binary	Time record was moved to SMF buffer
6	6	4	packed	Date record was moved to SMF buffer
10	A	2	EBCDIC	System identification
12	C	2	EBCDIC	System model identifier
14	E	1	binary	Driver control flags—specify which fields are to be ignored by the driver *Bit* *Ignore When Set* 0 Wait estimate 1 Region activity 2 Main storage occupancy 3 Swap load 4 Average queue service time 5 Background to foreground ratios 6 Scheduling of priority 7 Current RQEL
15	F	1	binary	Percentage of time to be given to background jobs (specified in MODIFY TS command)

Record Type 34 (TS-Step Termination)

Record type 34 is written each time the LOGOFF function processes a job-step termination. The length varies, depending upon the number of devices and the amount of accounting information.

This record contains the record type, time stamp (time and date), CPU identification, LOGON time, main storage occupancy time, count of TGETs satisfied and TPUTs issued, the time device allocation started, the time the problem program was loaded, job-step CPU time, termination status, TMP name, job-step name, size of region, and main storage used and the storage protect key. Input/output activity is recorded for each data set used by this job step; each entry lists the device class, type, and address, and the EXCP count for the data set. The data-set entry is zeros when the DD entry is TERM, DUMMY, or DYNAM.

The format is:

Decimal Displacement	*Hexadecimal Displacement*	*Field Size*	*Data Format*	*Contents*
0	0	1	binary	Reserved (zero)
1	1	1	binary	Record type 34
2	2	4	binary	Time of step termination
6	6	4	packed	Date of step termination
10	A	2	EBCDIC	System identification
12	C	2	EBCDIC	System model identifier
14	E	8	EBCDIC	User identification field
22	16	8		LOGON time—time of day (in hundredths of a second, four bytes, binary) and date (as 00YYDDDF, four bytes, packed decimal)
30	1E	8	EBCDIC	Reserved for user (blanks)
38	26	1	binary	Step sequence number
39	27	4	binary	Main storage occupancy time[1] in hundredths of a second
43	2B	4	binary	Line-out count, number of TPUTs issued
47	2F	4	binary	Line-in count, number of TGETs satisfied
51	33	2	binary	Step completion code[2]
53	35	1	binary	Step dispatching priority
54	36	8	EBCDIC	Terminal Monitor Program (TMP) name
62	3E	8	EBCDIC	Step name (Procedure)
70	46	2	binary	Size of region (in 1K blocks)
72	48	2	binary	Reserved (zero)
74	4A	2	binary	Main storage used (in 1K blocks)
76	4C	6	binary	Reserved
82	52	1	binary	Storage protected key (See TCBPKF in *OS System Control Blocks*, GC28-6628)
83	53	3	binary	Reserved
86	56	4	binary	Device allocation time in hundredths of a second
90	5A	4	binary	Problem program load time of day in hundredths of a second
94	5E	8	binary	Reserved
102	66	2	binary	Length of device entry portion, including this field, which is calculated: (8 x d) + 2, where d = number of devices

Decimal Displacement	*Hexadecimal Displacement*	*Field Size*	*Data Format*	*Contents*
One of the following 8-byte areas exists for each device.				
		1	binary	Device class from UCBTYP field of unit control block
		1	binary	Unit type from UCBTYP field of unit control block
		2	binary	Channel/unit address
		4	binary	EXCP count
The remainder of the record contains the following fields.				
		1	binary	Length of rest of record, excluding this field
		3	binary	Step CPU time in hundredths of a second[3]
		1	binary	Number of accounting fields
				Accounting fields[4]

[1] The main storage occupancy time may be invalid if the terminal monitor program supplied by IBM is not used. (Refer to *IBM System/360 Operating System: Time Sharing Option; Guide to Writing a Terminal Monitor Program or a Command Processor*, GC28-6764.)

[2] The contents of the completion code field varies according to the condition of termination, as follows:

X'0ccc', which indicates system ABEND; ccc is the ABEND code (see *OS Messages & Codes*, GC28-6631).
X'8ccc', which indicates user ABEND; ccc is the user ABEND code.
X'nnnn', which indicates normal completion; nnnn is the contents of the two low-order bytes in register 15 at termination.
X'0000', which indicates the last job step was not executed because of an error in the preceding job step.

Abnormal or normal termination can be determined from the job-termination indicator starting at byte 62 of record type 35.

[3] CPU time is not expected to be constant between runs of the same job. One or more of the following factors may cause small variations in CPU times: channel program retries, CPU architecture (such as core buffering), cycle stealing with integrated channels, queue searching (such as task switching), and pending interruptions.

[4] Each entry in an accounting field contains the length of the field (one byte, binary) followed by accounting information (EBCDIC). An omitted field is requested by a length indicator of 0.

Record Type 35 (Logoff)

Record type 35 is written each time a LOGOFF process has been completed. The length varies, depending upon the amount of accounting information.

This record contains the record type, time stamp (time and date), CPU identification, number of TGETs satisfied and TPUTs, storage protect key, session termination status, LOGON priority, LOGON sequence time, termination indicator, SYSOUT classes for session, and session CPU time.

The format is:

Decimal Displacement	*Hexadecimal Displacement*	*Field Size*	*Data Format*	*Contents*
0	0	1	binary	Reserved (zero)
1	1	1	binary	Record type 35
2	2	4	binary	Time of LOGOFF
6	6	4	packed	Date of LOGOFF
10	A	2	EBCDIC	System identification
12	C	2	EBCDIC	System model identifier
14	E	8	EBCDIC	User identification field
22	16	8		Logon time—time of day (in hundredths of a second, four bytes, binary) and date (as 00YYDDDF, four bytes, packed decimal)
30	1E	8	EBCDIC	Reserved for user (blanks)
38	26	1	binary	Number of steps in session
39	27	4	binary	Reserved (zero)
43	2B	4	binary	Line-out count, number of TPUTs issued
47	2F	4	binary	Line-in count, number of TGETs satisfied
51	33	2	binary	Job completion code[1]
53	35	1	binary	LOGON priority
54	36	8		LOGON enqueue time—time of day (in hundredths of a second, four bytes, binary) and date (as 00YYDDDF four bytes, packed decimal)
62	3E	1	binary	Termination indicators *Bit* *Meaning When Set* 0 Reserved (0) 1 Reserved (0) 2 Canceled at exit IEFUJI 3 Canceled at exit IEFUSI 4 Canceled at exit IEFACTRT 5 Reserved (0) 6 0—Normal completion 1—ABEND 7 Reserved
63	3F	5	binary	SYSOUT classes for session[2] (see record type 5 for explanation of field)
68	44	4	binary	Reserved (zero)
72	48	1	binary	Storage protect key (xxxx0000, where xxxx is the key which is described under TCBPKF in *OS System Control Blocks,* GC28-6628)
73	49	19	binary	Reserved
92	5C	1	binary	Length of rest of record (next four fields)
93	5D	20	EBCDIC	Reserved (blanks)
113	71	3	binary	Session CPU time (in hundredths of a second)[3]
116	74	1	binary	Number of accounting fields
117	75			Accounting fields[4]

[1] The contents of the completion code field varies according to the condition of the condition of termination, as follows:

X'0ccc', which indicates system ABEND; ccc is the system ABEND code (see *OS Messages & Codes*, GC28-6631).
X'8ccc', which indicates user ABEND; ccc is the user ABEND code.
X'nnnn', which indicates normal completion; nnnn is the contents of the two low-order types in register 15 at termination.
X'0000', which indicates the last job step was not executed because of an error in the preceding job step.

Abnormal or normal termination can be determined from the job termination indicator starting at byte 62 of this record.

[2] Each bit of the indicator represents the following classes:

Byte 0	*Byte 1*	*Byte 2*	*Byte 3*	*Byte 4*
Bit–Class	*Bit–Class*	*Bit–Class*	*Bit–Class*	*Bit–Class*
0–A	0–I	0–Q	0–Y	0–6
1–B	1–J	1–R	1–Z	1–7
2–C	2–K	2–S	2–0	2–8
3–D	3–L	3–T	3–1	3–9
4–E	4–M	4–U	4–2	
5–F	5–N	5–V	5–3	
6–G	6–O	6–W	6–4	
7–H	7–P	7–X	7–5	

[3] CPU time is not expected to be constant between different runs of the same step. One or more of the following factors may cause small variations in CPU times: Channel program retries, CPU architecture (such as core buffering), cycle stealing with integrated channels, queue searching (such as task switching), and pending interruptions.

[4] Each entry in an accounting field contains the length of the field (one byte, binary) followed by accounting information (EBCDIC). An omitted field is requested by a length indicator of 0.

Record Type 38 (Initial TS Configuration)

Record type 38 is written each time the time sharing option is started and after the TSC task has determined the time sharing configuration. The length varies, depending upon the number of swap devices.

This record contains the record type, time stamp (time and date), CPU identification, number of swap devices, and a device entry for each swap device. Each device entry contains the device class, unit type, and channel/unit address.

The format is:

Decimal Displacement	*Hexadecimal Displacement*	*Field Size*	*Data Format*	*Contents*
0	0	1	binary	Reserved (zero)
1	1	1	binary	Record type 38
2	2	4	binary	Time record was moved to SMF buffer
6	6	4	packed	Date record was moved to SMF buffer
10	A	2	EBCDIC	System identification
12	C	2	EBCDIC	System model identifier
14	E	22	binary	Reserved
36	24	2	binary	Length of rest of record (including this field)
38	26	1	binary	Number of swap devices

One of the following four-byte areas exists for each swap device.

		1	binary	Device class of swap devices from UCBTYP field of unit control block
		1	binary	Unit type of swap device from UCBTYP field of unit control block
		2	binary	Channel/unit address of swap device

Record Type 40 (Dynamic DD)

Record type 40 is written when the dynamic allocation function processes a deallocation, concatenation, or deconcatenation request. The length varies, depending upon the number of devices.

This record contains the record type, time stamp (time and date), CPU identification, logon time, dynamic allocation function indicators, and a data-set entry. Each data-set entry consists of the device class, unit type, channel/unit address, and EXCP count. The data set entry is zeros when the DD entry is TERM, DUMMY, or DYNAM.

The format is:

Decimal Displacement	*Hexadecimal Displacement*	*Field Size*	*Data Format*	*Contents*
0	0	1	binary	Reserved (zero)
1	1	1	binary	Record type 40
2	2	4	binary	Time record was moved to SMF buffer
6	6	4	packed	Date record was moved to SMF buffer
10	A	2	EBCDIC	System identification
12	C	2	EBCDIC	System model identifier
14	E	8	EBCDIC	User identification field
22	16	8		LOGON time—time of day (in hundredths of a second, four bytes, binary) and date (as 00YYDDDF, four bytes, packed decimal).
30	1E	8	EBCDIC	Reserved for user (blanks)
38	26	1	binary	Step sequence number
39	27	1	binary	Functional indicators 02—Deallocate 03—Concatenate 04—Deconcatenate
40	28	20	binary	Reserved (zero)
60	3C	2	binary	Length of rest of record (including these two bytes)

One of the following 8-byte areas exists for each device.

		1	binary	Device class from UCBTYP field of unit control block
		1	binary	Unit type from UCBTYP field of unit control block
		2	binary	Channel/unit address
		4	binary	EXCP count

Record Type 41 (Modify TS)

Record type 41 is written each time a MODIFY TS command is issued. The length varies, depending upon the number of time sharing regions that have been modified.

This record contains the record type, time stamp (time and date), CPU identification, time sharing initiation procedure name, time sharing task identifier, SMF foreground options, and maximum number of terminals allowed. For each time sharing region that is modified, there is an entry consisting of the total number of regions, region number for each modified region, LSQS in the region, and size of the region.

The format is:

Decimal Displacement	*Hexadecimal Displacement*	*Field Size*	*Data Format*	*Contents*
0	0	1	binary	Reserved (zero)
1	1	1	binary	Record type 41
2	2	4	binary	Time record was moved to SMF buffer
6	6	4	packed	Date record was moved to SMF buffer
10	A	2	EBCDIC	System identification
12	C	2	EBCDIC	System model identifier
14	E	8	EBCDIC	Time sharing initiation procedure name
22	16	8	EBCDIC	Time sharing task identifier
30	1E	8	binary	Reserved (zero)
38	26	1	binary	SMF foreground options *Bit* *Meaning When Set* 0 System and job data to be collected 1 System, job, and step data to be collected 2 Exits requested 3 Data-set accounting 4 Volume accounting 5 Reserved 6 Temporary data set scratch records 7 Always zero
39	27	11	binary	Reserved (zero)
50	32	2	binary	Maximum number of users allowed time sharing
52	34	26	binary	Reserved (zero)
78	4E	2	binary	Length of rest of record (including this field)
80	50	1	binary	Number of regions

One of the following 5-byte areas exists for each time sharing region that has been modified.

		1	binary	Region number
		2	binary	LSQS in region (in 2K blocks)
		2	binary	Size of region (in 2K blocks)

Record Type 42 (Stop TS)

Record type 42 is written by the terminate function of the TSC during the termination of time sharing options. It is also written when the TSC abnormally terminates one or more time sharing regions and none is restarted. The length is 50 bytes.

This record consists of the record type, time stamp (time and date), CPU identification, time sharing initiation procedure name, and time sharing task identifier.

The format is:

Decimal Displacement	*Hexadecimal Displacement*	*Field Size*	*Data Format*	*Contents*
0	0	1	binary	Reserved (zero)
1	1	1	binary	Record type 42
2	2	4	binary	Time record was moved to SMF buffer
6	6	4	packed	Date record was moved to SMF buffer
10	A	2	EBCDIC	System identification
12	C	2	EBCDIC	System model identifier
14	E	8	EBCDIC	Time sharing initiation procedure name
22	16	8	EBCDIC	Time sharing task identifier
30	1E	20	binary	Reserved (zero)

Index

Indexes to systems reference library manuals are consolidated in *IBM System/360 Operating System: Systems Reference Library Master Index*, GC28-6644. For additional information about any subject listed below, refer to other publications listed for the same subject in the *Master Index*.

Note: If more than one page number is given, the primary discussion is listed first. The entries in the index appear the same way they appear in the body of the book, which means that entries with bold type or italic type in the body of the book appear in bold type or italic type in the index.

Your Comments, Please ...

IBM System/360 Operating System
OS SMF
System Reference Library

Order Number GC28-6712-7

Your comments and constructive criticism regarding this publication will help us to improve it so that you may be better served. Each comment will be reviewed carefully by those responsible for the publication. Comments and suggestions become the property of IBM.

Requests for copies of publications or for assistance in using IBM systems and programs should not be offered as comments but should be directed to your local IBM representative.

Thank you for your cooperation.

Comments:

No postage necessary if mailed in the U.S.A.

GC28-6712-7

International Business Machines Corporation
Data Processing Division
1133 Westchester Avenue, White Plains, New York 10604
(U. S. A. only)

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

Your Comments, Please ...

IBM System/360 Operating System
OS SMF
System Reference Library

Order Number GC28-6712-7

Your comments and constructive criticism regarding this publication will help us to improve it so that you may be better served. Each comment will be reviewed carefully by those responsible for the publication. Comments and suggestions become the property of IBM.

Requests for copies of publications or for assistance in using IBM systems and programs should not be offered as comments but should be directed to your local IBM representative.

Thank you for your cooperation.

Comments:

No postage necessary if mailed in the U.S.A.

International Business Machines Corporation
Data Processing Division
1133 Westchester Avenue, White Plains, New York 10604
(U. S. A. only)

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

GC28-6712-7

Printed in U.S.A. GC28-6712-7

IBM®

International Business Machines Corporation
Data Processing Division
1133 Westchester Avenue, White Plains, New York 10604
(U.S.A. only)

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