NCE & TECHNOLOGY:

GUIDELINE ON MAJOR JOB ACCOUNTING SYSTEMS:

The System Management Facilities (SMF) for IBM Systems under OS/MVT

NBS Special Publication 500-40
U.S. DEPARTMENT OF COMMERCE
National Bureau of Standards

NATIONAL BUREAU OF STANDARDS

The National Bureau of Standards[1] was established by an act of Congress March 3, 1901. The Bureau's overall goal is to strengthen and advance the Nation's science and technology and facilitate their effective application for public benefit. To this end, the Bureau conducts research and provides: (1) a basis for the Nation's physical measurement system, (2) scientific and technological services for industry and government, (3) a technical basis for equity in trade, and (4) technical services to promote public safety. The Bureau's technical work is performed by the National Measurement Laboratory, the National Engineering Laboratory, and the Institute for Computer Sciences and Technology.

THE NATIONAL MEASUREMENT LABORATORY provides the national system of physical and chemical and materials measurement; coordinates the system with measurement systems of other nations and furnishes essential services leading to accurate and uniform physical and chemical measurement throughout the Nation's scientific community, industry, and commerce; conducts materials research leading to improved methods of measurement, standards, and data on the properties of materials needed by industry, commerce, educational institutions, and Government; provides advisory and research services to other Government Agencies; develops, produces, and distributes Standard Reference Materials; and provides calibration services. The Laboratory consists of the following centers:

Absolute Physical Quantities[2] — Radiation Research — Thermodynamics and Molecular Science — Analytical Chemistry — Materials Science.

THE NATIONAL ENGINEERING LABORATORY provides technology and technical services to users in the public and private sectors to address national needs and to solve national problems in the public interest; conducts research in engineering and applied science in support of objectives in these efforts; builds and maintains competence in the necessary disciplines required to carry out this research and technical service; develops engineering data and measurement capabilities; provides engineering measurement traceability services; develops test methods and proposes engineering standards and code changes; develops and proposes new engineering practices; and develops and improves mechanisms to transfer results of its research to the utlimate user. The Laboratory consists of the following centers:

Applied Mathematics — Electronics and Electrical Engineering[2] — Mechanical Engineering and Process Technology[2] — Building Technology — Fire Research — Consumer Product Technology — Field Methods.

THE INSTITUTE FOR COMPUTER SCIENCES AND TECHNOLOGY conducts research and provides scientific and technical services to aid Federal Agencies in the selection, acquisition, application, and use of computer technology to improve effectiveness and economy in Government operations in accordance with Public Law 89-306 (40 U.S.C. 759), relevant Executive Orders, and other directives; carries out this mission by managing the Federal Information Processing Standards Program, developing Federal ADP standards guidelines, and managing Federal participation in ADP voluntary standardization activities; provides scientific and technological advisory services and assistance to Federal Agencies; and provides the technical foundation for computer-related policies of the Federal Government. The Institute consists of the following divisions:

Systems and Software — Computer Systems Engineering — Information Technology.

[1]Headquarters and Laboratories at Gaithersburg, Maryland, unless otherwise noted; mailing address Washington,D.C. 20234.
[2]Some divisions within the center are located at Boulder, Colorado, 80303.

The National Bureau of Standards was reorganized, effective April 9, 1978.

COMPUTER SCIENCE & TECHNOLOGY:

GUIDELINE ON MAJOR JOB ACCOUNTING SYSTEMS:

The System Management Facilities (SMF) for IBM Systems under OS/MVT

Gary Durbin, Todd Kinney,
Peter Lamasney, Edward Newman, and
Edward Syrett

Tesseract Corporation
101 Howard Street
San Francisco, California 94120

Sponsored by the
Institute for Computer Sciences and Technology
National Bureau of Standards
Washington, D.C. 20234

U.S. DEPARTMENT OF COMMERCE, Juanita M. Kreps, Secretary
Dr. Sidney Harman, Under Secretary
Jordan J. Baruch, Assistant Secretary for Science and Technology
NATIONAL BUREAU OF STANDARDS, Ernest Ambler, Director

Issued October 1978

Reports on Computer Science and Technology

The National Bureau of Standards has a special responsibility within the Federal Government for computer science and technology activities. The programs of the NBS Institute for Computer Sciences and Technology are designed to provide ADP standards, guidelines, and technical advisory services to improve the effectiveness of computer utilization in the Federal sector, and to perform appropriate research and development efforts as foundation for such activities and programs. This publication series will report these NBS efforts to the Federal computer community as well as to interested specialists in the academic and private sectors. Those wishing to receive notices of publications in this series should complete and return the form at the end of this publication.

National Bureau of Standards Special Publication 500-40
Nat. Bur. Stand. (U.S.) Spec. Publ. 500-40, 170 pages (Oct. 1978)
CODEN: XNBSAV

Library of Congress Catalog Card Number 78-600113

U.S. GOVERNMENT PRINTING OFFICE
WASHINGTON: 1978

For sale by the Superintendent of Documents, U.S. Government Printing Office, Washington, D.C. 20402
Stock No. 003-003-01989-5 Price $4.00
(Add 25 percent additional for other than U.S. mailing)

GUIDELINE ON MAJOR JOB ACCOUNTING SYSTEMS: THE SYSTEM MANAGEMENT FACILITIES (SMF) FOR IBM SYSTEMS UNDER OS/MVT

by

Gary Durbin, Todd Kinney, Peter Lamasney
Edward Newman, Edward Syrett

ABSTRACT

This document reports the results of a study which was commissioned in response to the need for a better understanding of how job accounting systems work, what they measure, and how accurately they measure. The accounting system described is IBM's System Management Facilities (SMF) for 360/370 environments operating under OS/MVT. Considerable detail is provided on SMF's activity in collecting the data necessary to account for resources utilized by the individual jobs in a multiprogramming system and to provide some indicators of the performance of the system itself. Included within the scope of this study was an investigation of the accuracy, both absolute and relative, of SMF as a measurement tool and the costs entailed in the use of SMF. The experimental methodology used to explore these questions is summarized in the report, as are the conclusions reached. Performance analysts, system programmers, and managers interested in optimal use of their computer resources will find much help here in the application of a valuable measurement tool.

Keywords: Computer performance analysis; computer performance measurement; CPU time; data accuracy; data validation; EXCP counts; IBM OS/MVT; job accounting systems; resource utilization measurement; System Management Facilities; system wait time.

NBS FOREWORD

The usefulness of vendor-supplied job accounting systems in improving the performance of computer installations has long been recognized. Originally designed for user chargeback, these supervisor-resident programs produce data on workload and resource utilization that are essential to the estimation of installation efficiency. They are comprehensive, convenient to use and--being bundled with standard-release operating system software--essentially free of charge. They have been used to forecast workload, drive prediction models, estimate capacity, reconfigure equipment, and set priorities for software optimization. They are arguably the most widely available and most widely used of performance measurement tools.

Taken as a group, job accounting systems have one additional characteristic that makes them particularly interesting to auditing and standards activities: nearly all contemporary medium-to-large scale computer installations have them. Some collect accounting data but make no practical use of it, while others use it only to charge customers for the computing services they receive. Yet nearly all have the capability of collecting this data and the potential for making productive use of it. It is not surprising, therefore, that some federal managers have begun to turn to this tool as the basis for describing the performance of whole classes of computer installations, and even comparing performance from one site to another.

Several technical obstacles stand in the way of these expanded uses of accounting data, however. Different job accounting systems reflect unique vendor ideas of hardware and software architecture and attitudes toward the accounting function. Consequently, they measure different things, measure the same things differently, and record resource demands that may not be duplicated for the same workload on any other system. It is not always clear what specific system event a measurement represents, how precisely it is being measured, or what calibration it requires. Any extended use of job accounting data--either at the local installation level or for complex performance evaluation purposes--will require an exact understanding of the accounting system as an instrumentation device.

The study on which the following report is based was commissioned in response to the need for better understanding of how accounting systems work, what they measure and how well they measure it. It attempts to apply rigorous criteria of description, analysis and validation to an already valuable and potentially influential method of computer instrumentation. The report deals with the job accounting system used by most federal installations equipped with third-generation IBM mainframes: the System Management Facilities (SMF) for OS/MVT sytems. More than 200 medium-to-large scale federal installations are capable of generating SMF data under the OS-series of operating systems.

To managers and technicians at these installations and others like them in the private sector, this report is recommended reading before attempting to use job accounting data for performance improvement, reporting, prediction or control. It is intended neither as an endorsement of specific job accounting systems and the equipment that supports them, nor as a criticism of those systems for falling short of objectives they were never intended to meet. On the contrary, it is offered in recognition of the fact that accounting data already gives us an invaluable window on computer performance, but that its application must be qualified by an understanding of its limitations.

Seymour Jeffery, Chief
Systems and Software Division
Institute for Computer Sciences
and Technology
National Bureau of Standards

Authors' Preface

The purpose of this report is to describe SMF, to suggest what it is good for, and to caution against what it is not good for. "SMF" refers to the System Management Facilities of IBM OS/360 MVT. As part of Tesseract's investigation of SMF, a series of controlled tests were performed to measure the accuracy and reliability of SMF data, and the overhead incurred in collecting and recording the data. The results of this validation experiment are summarized in Section VI.

The intended readership of this report consists of three major groups: installation managers, performance analysts, and system programmers. Since the detailed analysis of SMF presented here requires extensive use of specialized terminology and acronyms, a glossary of abbreviations and an introductory section have been included to assist those readers not in daily contact with OS internals. Obviously, system programmers could bypass these sections, especially on first reading this report. Similarly, some of the most detailed analyses of the structure and function of SMF might well be skipped by readers concerned with the overall evaluation of its suitability for their sites' needs. This level of detail was included in the hope that this report could serve the system programmers among its readership as a Program Logic Manual (PLM) for SMF, concentrating in one short document material scattered throughout the existing PLM's (<4>, <5>, <10>, <11>, <12>, <13>), or in some cases available only in microfiche form. It should be noted, however, that this report deals only with SMF on the MVT version of the IBM Operating System. The Time-Sharing Option (TSO), Virtual Storage (VS), and Multiprogramming with a Fixed number of Tasks (MFT) are not covered in this report, though occasional references to them occur.

This report assumes that its readers have ready access to the OS SMF manual (<2>) for elaboration of background material which is only outlined here. Although the SMF manual is not a prerequisite, this report is intended to be read in conjunction with it. For those readers who are not familiar with OS/360 from either the user's or the operator's viewpoint, the manual entitled IBM System/360 Operating System: Introduction (Order Number GC28-6534) is highly recommended.

The organization and readability of this report have benefitted immensely from the constructive criticism, guidance and suggestions provided by Richard Dunlavey and Dennis Conti of the National Bureau of Standards. The authors gratefully acknowledge the wholehearted cooperation of everyone on the staff of Greyhound Computer Corporation's San Francisco Data Center, especially the invaluable assistance of the customer engineers in the hardware monitoring phase of the project. Several obscure technical points were clarified in enlightening discussions with Michael W. Haven of the G.C.C. systems programming department. Naturally, any deficiencies of the report are the responsibility of the authors.

Table of Contents

I. Introduction to SMF Concepts

A. General Overview

1. Purpose and background of SMF

The purpose of SMF, as stated in the IBM manual (<3>), is to "provide the means for gathering and recording information that can be used for billing customers or evaluating system usage." The need for such data-gathering facilities arises directly from the fundamental concept of third-generation computer architecture, namely, the sharing and concurrent use of diverse resources by several jobs running independently under the control of an operating system. During one second of operation, the central processing unit (CPU) of a large-scale, modern computer may perform calculations for several jobs and also spend some time idle, while the channels simultaneously move data to and from disks, tapes, card readers and printers on behalf of several other jobs. The only way to account for the costs of individual jobs in such a system is to have the supervisory program keep track of how much of each resource it dispenses to each job, and for how long. Actually, since the natural unit of program execution is the job step, most resources are accounted for on a job step basis.

The accounting capabilities of the IBM 360 Operating System have evolved from release to release along with most other OS functions. Initially, all that was provided was an exit to an installation-written accounting routine at the end of each job step and at the end of each job. This routine was given relatively little information about the resources requested by the job or job step, and almost none about the amounts actually used. Many large-scale 360 users found the accounting exit inadequate, but their discontent with this and other aspects of OS was alleviated somewhat with the appearance of HASP. This IBM field-developed spooling package supplemented the operating system in the area of job queuing as well as in operator communication and storage of output awaiting printing or punching. Because of the way that HASP insinuated itself into control of functions normally reserved for the operating system, it provided a convenient "window" on resource allocation at a detailed level. Modifying HASP or adding code to it was easier than interfacing directly with OS, so many installations came to rely on HASP as their major source of accounting data. Still, there were many areas in which HASP could supply little information. Perhaps the most important of these was the use of space, both in main memory and on direct access storage devices.

As installations accumulated experience with third-generation systems, they became increasingly interested in improving throughput and turnaround by using the many control options available to operators and system programmers. But before they could improve their performance, they first had to be able to measure it. Thus there arose another more stringent requirement for the system to make available quantitative

information, not only on the use of resources by jobs, but also on its own behavior. Ideally, such data could be displayed dynamically in a form enabling the operator to eliminate bottlenecks and maximize the system's performance.

In response to these different but related demands, IBM made the SMF option available in Release 18 of OS. Many of the properties of SMF which are discussed in detail later in this report, properties observed in shops using SMF and confirmed by the analysis and validation experiments, were predictable from the history of its development. It was added to the operating system as an afterthought, not designed into it from the start. It had to be compatible with the old step-end accounting exit. For economic reasons, it could not involve any major redesign of OS, nor add significantly to the system's overhead. Under these circumstances, it is remarkable that SMF comes as close as it does to satisfying the requirement for collection and recording of data on which a reasonably fair and repeatable assignment of costs to jobs can be based.

As a case in point, the amounts of CPU time reported by SMF as being used by two job steps performing identical calculations on the same hardware may differ by more than 25%. The main reason for this is that the overhead of supervising input/output operations is distributed randomly over all jobs in the system rather than being charged to the jobs which originate the I/O. Technically, this happens because the I/O Supervisor fails to turn off the task timer immediately after an I/O interruption. Historically, it happens because SMF uses the job step timing facility of OS, which was originally designed to detect and automatically cancel any program which took more CPU time than its user estimated it should need. For that purpose, a 25% variability is obviously acceptable. For performance measurement, something better is needed; this often turns out to be a hardware monitor (<16>, <18>).

As a tool for overall system tuning, SMF is even more severely limited, although some use can still be made of it (<14>, <15>). The major limitation of SMF for performance evaluation is that it provides next to no information on OS overhead. Another limitation is its failure to record significant operator actions such as mounting disks and tapes or resetting the time-of-day clock. Again taking CPU time as an example, the OS SMF manual suggests using the relationship:

System time = Elapsed time - (Wait time + Total job time)

to compute the CPU time spent performing system tasks. The trouble with this method is that both elapsed time (determined from time stamps on the SMF records) and wait time can be rendered meaningless by operator actions like resetting the clock or stopping the system temporarily, actions which are not recorded by SMF.

In general, if SMF data is accumulated over a long enough time span to average out anomalies due to operator actions and system failures, it can provide a qualitative picture of performance and reveal chronic weaknesses in the configuration of the system. The shorter the time span, the less reliable SMF data is for performance evaluation. For locating transient bottlenecks or tuning the system dynamically to its workload, SMF is essentially useless.

2. Outline of SMF capabilities

The principal functions provided by SMF are:

* collection of data on system performance;
* collection of data on use of resources by jobs and job steps;
* collection of data on creation and use of data sets;
* organization of such data into records of various types;
* writing of these records onto data sets on disk or tape;
* backing up ("dumping") of these data sets to archival storage;
* execution of installation-supplied routines as components of the operating system at specific times during processing of each job;
* communication with the operator concerning these activities.

The principal function not provided by SMF is the production of reports based on analysis of the data which it collects and records. Each installation must separately acquire or devise its own analysis and reporting tools.

Whether SMF is included in the system at all is the installation's option at the time the system is generated from the materials distributed by IBM. The first stage of this system generation (Sysgen) process consists of the specification of a large number of symbolic parameters in macroinstructions which are expanded by the 360 assembler into control statements; in the second stage, these control statements determine the selection of optional OS components from IBM's menu. The specific parameters pertaining to SMF are discussed below in section II.A.3.

Once SMF is included in the operating system, the installation has many options at its disposal, such as classes of data to be collected and recorded, size of SMF data sets and buffers, and execution of user-written SMF exit routines. SMF exits are discussed further in section IV.A.1 below. Various SMF options are controlled by specification of parameters in the member SMFDEFLT of the system library SYS1.PARMLIB. If the parameter OPI=YES is specified in SMFDEFLT, the operator is allowed to modify the other SMF parameters each time the system is loaded. The major options available here are to limit or suppress the recording of SMF data.

SMF collects performance data at system, job, and job step levels. The system performance data collected by SMF includes the time of each IPL, a list of devices online at IPL time, the address of each device varied online or offline, the amount of free space on each direct-access volume online and ready at IPL time, and magnetic tape error statistics. The time spent by the CPU in the wait state is accumulated over 10-minute intervals and recorded in the SMF data set somewhat less often. Not included in the SMF system performance data are such desirable items as CPU time used by specific system tasks, usage counts for OS modules, lengths of internal system queues, I/O counts for system data sets such as SYS1.LINKLIB and SYS1.SVCLIB, the reason for each IPL, and a record of the request for and performance of each volume (tape or disk) mount.

The data collected by SMF on the use of resources by jobs and job steps includes the elapsed time, CPU time, main storage utilization and counts of I/O accesses by device and data set for each job step. SMF records the time each job is read into the system, the time it is enqueued for execution, the time it is selected for execution, and the total CPU time it uses. Whenever a device is brought online specifically to satisfy the allocation requirements of one particular job step, a special SMF record is written identifying the device and job step involved. In systems without HASP, where spooling of output to be printed or punched is handled by OS itself, SMF provides the number of records written per form number and output class for each job.

The CPU time figures recorded by SMF are subject to variability, as mentioned above. The usefulness of the I/O counts ranges from near zero for graphics terminals and teleprocessing lines to a high level for sequential files with fixed block sizes. Under the newer MVS operating system, even this limited usefulness is severely impaired by the introduction of variability into the I/O counts for most sequential processing. The main storage figures reported are satisfactory for cost accounting but inadequate for performance analysis, since storage fragmentation or "checkerboarding" is not revealed by the SMF data.

Complementary to the dynamic use of resources by jobs is the static use of disk and tape storage by data sets. To account for this service, SMF writes a record every time a user data set is closed, re-named or scratched. While these records provide voluminous quantities of information about each data set, they lack certain features essential to any practicable tape and disk space accounting system. The most obvious loophole is that a job can create a data set and reserve any available amount of disk storage for it with no indication of this in the SMF records until the first time something is actually written into the data set. In the meantime the space is unavailable to other users. Because of this and other deficiencies in the SMF data set records, few installations have tried to use them as a source of data for tape and disk space accounting.

One positive feature of SMF is its open-endedness. An installation may design its own SMF records to supplement or supplant the standard ones, and use the IBM-supplied routines for recording them via a convenient well-documented interface. A few simple fixes to OS and judicious use of exit routines could make up for many of SMF's shortcomings.

The most common use of SMF exit routines is to enforce installation standards. Since the user's job control statements are accessible to certain exit routines, they can check for validity of account numbers or any other significant parameters. SMF exit routines have the power to cancel unacceptable jobs, or to override their parameters to force compliance with standards. They can also collect additional data on job and system status, construct supplementary records and have them written to the SMF data set, and cause messages to appear on the user's printout or the operator's console. Thus the SMF exit points are convenient places to "hook" OS to tailor it to an installation's requirements. Even here, however, something needed for performance analysis is lacking, namely, an exit taken at regular time intervals to permit sampling of system status. All of the SMF exits in OS/MVT are triggered by events in the lives of jobs moving through the system, and thus occur at unpredictable times.

The cost in additional overhead incurred by including SMF in an MVT system is low compared to the cost of the programming effort (or software package purchase) necessary to derive significant benefits from SMF. Besides the data reduction and reporting functions required for job accounting, exit routines should be supplied and some changes to OS itself might be advisable for purposes of operational control and performance analysis. Major improvements to SMF should not be expected from IBM, either in MVT or in the virtual-storage (VS) versions of the system. The later releases of VS2 offer a system activity measurement facility, MF/1, which serves as a software monitor. Also available in OS/VS is the Generalized Trace Facility (GTF), whose use in performance analysis is covered in <40>.

B. Methods of Data Collection

1. Internal data collection

Within OS/360, resource-related control information resides in queues representing unsatisfied resource requirements of tasks, and tables identifying available resources. These queues and tables are typically composed of control blocks chained together by pointers. A control block is a contiguous area of main memory or auxiliary storage containing control information and pointers to other control blocks. A pointer is simply the address of some location in main memory or auxiliary storage. During operation of the system, control blocks are created as needed by various components of OS, or in some cases by user-written programs. Other modules maintain, update and refer to the information they contain. When control blocks are no longer needed, pointers to them are erased and the storage they used is made available for reallocation.

As each job moves through the system, control blocks containing information on its resource usage are thus created, maintained, and eventually deleted. Such information tends to migrate from one control block to another as time goes by. The basic method used by SMF to collect its data is to construct records in a buffer in main memory by piecing together fields derived from various system control blocks. These records are then written onto the SMF data set on disk or tape, provided that their writing has not been suppressed by the installation and that data set space is available. In the MVT version of OS/360, there are 20 types of basic SMF records, containing different collections of data elements derived from a wide variety of sources. The record types are discussed below in section III.B. The data elements are tabulated in Appendix D,followed by more detail on some of the most significant data elements. A discussion of the variability of CPU time and EXCP counts is presented in the summary section, Section VI.

2. External data collection

The external sources of SMF data are:

* JCL and commands from incoming job streams;
* commands and replies entered through the operator's console(s);
* member SMFDEFLT of the system parameter library SYS1.PARMLIB;
* data set and volume labels; and
* the system control panel, especially the LOAD and STOP keys.

Most of the information derived from these sources is converted by the operating system to an internal representation as part of some control block. It is then captured by internal SMF data collection routines and scheduled for recording as outlined in the previous section. Only two of the above sources are specific to SMF; all the others are converted to internal form as part of the normal operation of OS, e.g., reading and interpretation of incoming job streams.

The two SMF-specific external data sources are the SMF options and the SWITCH command. The SMF options are initially read from SMFDEFLT at IPL time, and may be modified or corrected by operator responses to messages from OS. In this case, the conversion to internal form is performed by components of SMF itself. While the SWITCH command does not have any parameters to be converted and collected as data, it does cause a special SMF record to be written, as well as a switching of SMF data sets.

II. Installation and Operation of SMF

A. Installation and Maintenance of Software

1. Hardware requirements

Since SMF is a component of OS/360, it requires at least the hardware needed to support that system: specifically, an IBM 360 or 370 mainframe, having enough processor storage to accommodate the OS nucleus, System Queue Area (SQA), and link pack area, with storage left over for concurrently executing job steps; the mainframe must be connected through its channels with at least one direct-access storage device. In the special case of Model 65 Multiprocessing, the M65MP option of OS/MVT, two mainframes can be used with SMF. For details see <9>, <22>, <23> pp. 91-93, <24> Appendix D, and primarily <25>.

The OS SMF manual (<2>) states that the additional main storage required for SMF in the resident nucleus of an MVT system is 1700 bytes. The amount of storage used by SMF in the SQA varies dynamically with the workload. Of this amount, the part consisting of the SMF buffer and the System Management Control Area (SMCA) is fixed in size at IPL time. The SMCA occupies 124 bytes, and the SMF buffer takes at least 400 bytes; values like 1400 or 2800 bytes are more representative of the buffer sizes likely to be used to avoid spanning of records and to optimize use of direct-access space on various devices. The variable part of the SQA requirement is the space for a Timing Control Table (TCT) for each active job step in the system. If only job accounting is requested, by specifying OPT=1 in SMFDEFLT, each TCT takes 96 bytes of SQA space. If OPT=2 is specified to request job-step accounting, then each TCT is augmented by a Timing Control Task Input/Output Table (TCTIOT) whose size depends on the number of DD statements and the number of units allocated for the job step. The formula given by <2> for TCT size with OPT=2 is

$$TCT = 112 + 12*a + 8*b$$

where "a" is the number of DD statements and "b" is the number of units, counting each unit as many times as it was allocated to data sets in the job step.

Further requirements for SQA space depend on the presence and characteristics of user-written exit routines. When such routines are included in the system, the region sizes for the Reader/Interpreter and Initiator/Terminator and the size of the link pack area must be increased to accommodate them.

The OS SMF manual would seem to imply that a system with SMF must have at least one tape drive, since the SMF data set(s) must either be allocated on tape or dumped from disk to tape periodically. This is not strictly true, however, since private, removable disk packs can be used in place of tapes as the media to which the SMF data is dumped. What is required, assuming

SYS1.MANX and SYS1.MANY to be on a Direct Access Storage Device (DASD), is that these data sets be allocated sufficient space on a volume or volumes mounted prior to IPL on a device or devices which must be online and ready during IPL. If two devices are used, they must be of the same type. Any device used for SMF data recording becomes permanently resident at IPL time; this means that the volume on that device cannot be dismounted.

The amount of DASD space to be allocated to the SMF data sets depends so completely on the installation's workload characteristics and archiving strategy that no meaningful estimate can be given here. However, a detailed example of how to arrive at an estimate, given realistic assumptions about the frequencies of such events as IPL's, VARY commands, job submissions, and step terminations, can be found in <2>, Table 12. The additional space required for system libraries amounts to six 2314 tracks, a quite negligible fraction of the thousands of tracks commonly devoted to these libraries (SYS1.LINKLIB, SYS1.SVCLIB, SYS1.NUCLEUS and SYS1.PARMLIB).

Altogether, the hardware requirements of SMF are modest and should not strain the resources of any OS/MVT installation. About the only way to create a noticeable impact would be to allocate the SMF data set on tape, thereby in effect reducing by one the number of tape drives available on the system's configuration.

2. System generation options

To include SMF in the operating system, certain parameters must be specified at the time the system is generated. Also, some optional features of SMF require action by the system programmer at this time if they are to be usable later.

In the SCHEDULR macroinstruction, coded as part of the first stage of system generation, the prospective SMF user must:

* specify ACCTRTN=SMF;

* specify ESV=SMF if error statistics by volume are desired, i.e., if a Type 21 SMF record is to be written each time a user data set on magnetic tape is closed or processed by end-of-volume;

* increase the value of the JOBQLMT parameter by 2 to allow work queue space for two extra SMF messages;

* increase the value of the MINPART parameter to provide sufficient main storage space for as many of the user-written exit routines (IEFUJI, IEFUSI and IEFACTRT) as are supplied.

In the SUPRVSOR macroinstruction, also part of stage I Sysgen, the SMF user must:

* specify TIMER=JOBSTEP; and

* specify RESIDNT=ACSMETH to make certain access method modules resident in main storage.

The BSAM modules IGG019BA and IGG019BB must always be resident for SMF. If SMF data is to be recorded on magnetic tape, modules IGG019BD and IGG019CC must also be resident. If the SMF data sets are allocated on DASD, the module IGG019CD must be resident; if the device is one having the Rotational Position Sensing (RPS) feature, e.g., an IBM 3330, the modules IGG019CJ and IGG019FN must also be resident. To improve performance of systems with SMF, module IGC0008C (the first load of SVC 83) should be made resident. If data-set information is requested (DSV=2 or 3) then the modules IFG0202H and IFG0202I should also be made resident; they are called to build a Type 14 or 15 SMF record whenever a user data set is closed or processed by EOV.

User-written routines such as SMF exits should be link-edited into the distribution library SYS1.CI505 before the system generation process is begun. For detailed information on system generation, refer to <9>.

3. System programming and vendor support requirements

a. System programming requirements

SMF requires very little direct maintenance in that there are few problems with the components and few new features. The system programming time required to implement SMF is almost entirely spent either setting up operating procedures for archiving the SMF data or developing programs for the analysis of the data. The operating procedures are a one-time expenditure that, if conscientiously done, will need very little maintenance. The analysis programs, on the other hand, can require as much time as an installation is willing to commit. Several software packages that provide reports from SMF data are currently on the market.

b. Vendor support

SMF is fully integrated into the operating system and supported as Type I programs. IBM-developed PTF's (Program Temporary Fixes) are available in SMP format (<34>) starting with release 21.8. Outstanding SMF APAR's (suspected bugs) are listed in the OS Early Warning microfiche TNL S2C0-0149.

B. Operating Considerations

1. Normal startup and data collection procedures

a. General discussion

SMF records can be written on magnetic tape or on disk. If tape is chosen as the first storage medium for SMF records, the file simply remains there for further processing. If the SMF records are to be written on disk, however, two data sets must be provided: SYS1.MANX, the primary SMF data set, and SYS1.MANY, the alternate SMF data set. SMF adds records to one of these files (beginning with SYS1.MANX) until the file is full. At this point, SMF notifies the operator that the currently active data set should be dumped, and switches automatically to the other one. If the reserve data set is also full, recording of SMF data is suspended until an empty data set is available to receive it. Dumping an SMF data set to tape requires submitting a job to execute the IBM-supplied SMF dump routine (IFASMFDP); the operator could do this via a START command. Rather than waiting for the active SMF data set to fill up and cause an automatic switch to the reserve data set, the operator can force switching to the reserve by means of the SWITCH command.

Magnetic disk is the storage medium most frequently used for the purpose ascribed to it above; but, in fact, any DASD can serve as the first repository for SMF records. Furthermore, when the occasion of dumping the direct-access file arises, the file may be dumped not only to tape, as described above, but, additionally or alternatively, to another DASD or almost any other medium. For further discussion of the post-dump storage of SMF records, see III.A.2 "Dumped log files" below.

Frequent use of the switch-and-dump procedure described above is liable to create operational problems, since the operator must submit (or start from the console) the proper one of two possible jobs: one that dumps SYS1.MANX and one that dumps SYS1.MANY. To simplify the switch-and-dump procedure and reduce the likelihood of human error, some installations (<8>) have modified SMF to start the proper dump job automatically whenever a switch takes place, rather than asking the operator to do so. Another approach is to have the dump job first dump the primary, then simulate console entry of a SWITCH command to resume recording in the now-empty primary, and finally dump the alternate. Using this scheme, very little space would be needed for the alternate data set.

b. Normal IPL procedure

At startup (IPL) time, the operator may receive a console message asking him to correct or redefine the SMF parameters read by the system from member SMFDEFLT of SYS1.PARMLIB. This will happen if one or more parameters are improperly specified, if a required parameter is omitted, or if the parameter OPI=YES is present. If the unit and volume to be used for the SMF data

set(s) are not defined in SMFDEFLT, if the data set is not allocated on the specified device, or if the device is unavailable, a message is sent to the operator warning him that the SMF function is disabled. Otherwise, the system opens the SMF data set(s); this action may cause one or two expiration date override messages, to which the operator should reply 'U'. Expiry date protection of the SMF data sets (and any other valuable files in the system) is recommended to prevent inadvertent or malicious destruction of their contents.

If both SMF data sets on direct-access storage contain data, the operator will receive a message requesting him to dump the one with more data in it. Meanwhile, SMF recording will begin on the other data set. If only one data set is non-empty, it is selected for recording; if both data sets are empty, the primary (SYS1.MANX) is selected. In any case, a console message tells the operator where the SMF data is being recorded.

Since speed is usually of the essence during IPL, the interaction of the operator with SMF should be minimized by the following procedures:

(1) Allocate the SMF data sets on drum or permanently resident disk.

(2) Once the SMF IPL parameters have been debugged, shut the operator communication option off by setting OPI=NO in SMFDEFLT. This procedure also insures uniform collection of data.

c. Normal data collection

In order to protect against accidental loss of data due to full SMF data sets and decrease exposure to hardware failures, the following procedures may be used:

(1) Regularly perform the SMF dump function as a part of normal operations, either at the beginning of each shift or at a particular time each day. Performing this function on a regularly scheduled rather than a demand basis will improve the reliability of data collection.

(2) Assign the responsibility for SMF data collection to someone other than a computer operator. This insures that the process will not be overlooked during a period of heavy system activity.

(3) Produce printed reports from the collected data for delivery to installation management or for use in daily operations.

A data collection procedure that minimizes both disk storage requirements and exposure to hardware errors is:

(1) Allocate enough space in the MANX data set to collect records for the collection period plus a substantial margin for growth. This space estimate can be checked from time to time by listing the VTOC for the data set immediately prior to dumping and then examining the last TTR pointer to determine the high water mark. Only a minimum of space is allocated to the MANY data set, to allow for a few hours' data collection.

(2) At the beginning (or end) of each shift (or each day), the operator issues the SWITCH command and starts the MANX dump from the console. When that dump completes, the operator issues another SWITCH command and starts the MANY dump.

(3) The operators are instructed to consider any forced switch as a serious problem and immediately start the MANX dump and report the occurrence of the event to the systems maintenance staff.

(4) The MANX and MANY dumps run from disk to disk with each one allocating an output data set on resident work volumes. Following successful emptying of the collection data sets, the intermediate data sets can be copied to backup and archive files. Care must be exercised when setting up the JCL for these processes, since there is no way to recapture data that is accidentally scratched or is written on a bad device.

One method of decreasing the risk of I/O errors is to create two or more copies of the collected data or the merged archive file. This is particularly desirable when the data is moved to a more volatile medium, such as tape. Before scratching the intermediate disk data set, then, two copies of the data should be made--for example, one tape copy of the two intermediate files concatenated together, and a week-to-date merged file. Once all copies are complete, the intermediate data sets can be deleted for the next dump process.

2. System failure and lost-data procedures

Some provision has been made in SMF for recovery from errors in other components of OS, both hardware and software, and even from a catastrophic failure of the entire system. For example, after each block is written to the SMF data set, a temporary CLOSE is performed to place a recognizable end-of-file marker in the data set immediately following the last block. This precaution keeps the loss of data due to most system crashes to at most one block of records. Without careful planning by the user, however, significant losses are still possible. The worst

case is obviously that of a mechanical failure causing physical damage to the storage medium on which the SMF data is being recorded. Loss due to this type of accident can be minimized by using fairly small SMF data sets on disk and dumping them frequently. However, as discussed previously in section II.B.1, this strategy increases the risk of operational errors.

If it becomes impossible to record SMF data because the currently active data set is full and the reserve data set is not empty, SMF suspends writing of records but continues to generate them internally and tallies the number not written. When the reserve data set becomes available, SMF resumes writing, beginning with a Type 7 (Data Lost) record that tells how many records were generated but not written during the hiatus. This situation may arise because of some difficulty in dumping the reserve data set, e.g., the dump job terminates abnormally and the operator fails to notice it. It should be observed that if an IPL were to occur just before the active data set overflowed, and if the alternate were only, say, half full but was scheduled to be dumped because of a SWITCH command, SMF data would start being recorded in the alternate after the IPL. This would definitely cause records to appear out of sequence in the dumped SMF data; it could also lead to the attempted dumping of an SMF data set while records were being written into it by the system. The IFASMFDP program, however, refuses to dump the active data set.

The situation just outlined in which a disk-to-tape SMF dump job survives an IPL and attempts to dump the wrong data set is only one of a variety of pathological cases that can be avoided entirely by writing the SMF records directly on tape in the first place. Proper chronological sequencing of the records is then guaranteed. There are, however, several drawbacks to this approach. First, it requires dedicating a tape drive full-time to a function which is not directly productive. Second, its future potential is limited, since the OS/VS version of SMF requires the use of direct-access storage for the SMF data sets. Third, it is quite vulnerable to I/O errors on the tape, since the response of the SMF writer to a permanent I/O error is to act as though no more space is available for writing SMF records.

Another potential trouble area is failure of the access method routines which are used to write records to the SMF data set. In case of an ABEND (software failure) in one of these routines, the SMF writer suspends further movement of jobs through the system (see section IV.B.2 below) while it sends a message to the operator asking his permission to continue with SMF recording disabled. (The alternative is for the operator to take a system dump to aid in analysis of the failure.)

b. Types of failure

Several classes of failures affect SMF data and its collection; they fall into three general categories: software failures, hardware failures, and operational errors.

i. Software failures

The failure of the system software components responsible for collection of SMF data will generally cause failure of the operating system itself and require an IPL. The net result of such a failure is incomplete data since many job steps in progress will not complete and termination records will not be written. It is possible with some failures to bring the system to an orderly shutdown and thereby collect some of the SMF data using the procedures outlined in section II.B.1 above.

ii. Hardware failures

These failures, if not correctable by the operating system, are either CPU or channel failures that bring the operating system down or device failures affecting the SMF data sets. If the operating system is stopped due to hardware failure, the data on steps in progress is lost. Simple I/O errors on the SMF data sets will normally result in the loss of SMF records, but loss of the SMF data sets may be considered severe enough by the installation to force shutdown of the operating system.

iii. Operational errors

Whenever there are hardware or software failures, a few SMF records may be lost, but operational errors can destroy or invalidate a whole day or shift worth of data. Proper procedures and operator training are essential if the SMF data set is to be consistently usable.

A common operator error is improper system shutdown. Whenever the operation of the system is terminated, whether due to some type of failure or simply a weekend, all never-ending tasks such as HASP should be stopped and a HALT command issued to complete writing of records.

The careful use of the set clock command must be enforced, since much analysis of SMF data is heavily dependent upon the accuracy of the system clock.

III. Description of SMF Files

A. Structure of SMF Files

1. Online log collection files

SMF records are variable-length, blocked, and spanned. In this format, data is stored in physical records called "blocks", each of which may contain one, more than one, or part of one logical record. Each block begins with a Block Descriptor Word, in which is stored the length of the block. The rest of the block consists of one or more segments. Each segment starts with a Segment Descriptor Word, which contains the length of the segment and a code indicating whether the segment is a complete logical record or is at the beginning, middle, or end of one.

The types of records, if any, to be written in the SMF data sets are controlled by the values of the parameters in member SMFDEFLT of data set SYS1.PARMLIB. If SMF as well as user records are to be written (parameter MAN=ALL), the first record in the original (pre-dump) SMF file for each run (where "run" denotes a period between consecutive IPL's) will be a Type 0 (IPL) record.

2. Dumped log files

Dumped log files are much the same as the original online files, except that each file is preceded by a Type 2 (Dump Header) record and followed by a Type 3 (Dump Trailer) record.

There are many ways to provide for the long-term storage of SMF data. Among these are:

a. generation data group on disk or tape (<1>, <6>, <7>);

b. consecutive files on unlabelled tape (<7>);

c. addition of data to the end of a disk or tape data set by use of a DISP=(MOD,KEEP) parameter in the JCL for the dump (<7>). Note that in this case, the Type 2 and 3 records actually serve a purpose: they delimit consecutive dumps.

B. Structure of Records

1. General discussion

Each SMF record generated by the IBM-supplied routines begins with a standard 18-byte header, including the four-byte Segment Descriptor Word mentioned in section III.A.1 above. The OS SMF manual (<2>) recommends that records produced by user-written routines should also include this standard header. Its format is as follows:

Decimal Offset	Hex Offset	Field Size	Data Format	Description of Contents
0	0	2	Binary	Record length in bytes
2	2	2	Binary	Segment control code
4	4	1	Binary	Header flag byte
5	5	1	Binary	Record type
6	6	4	Binary	Time in 100ths of seconds
10	A	4	Packed	Date as 00yydddF
14	E	4	EBCDIC	System & model identifier

The segment control code has the following possible values and meanings (values given in hexadecimal):

Value	Meaning
0000	Complete logical record, not segmented
0100	First segment of a multisegment record
0200	Last segment of a multisegment record
0300	Intermediate segment of a multisegment record

The header flag byte is always binary zeroes for OS/MVT; it is reserved for use with VS. The record type is an eight-bit, unsigned binary number; values 0 through 127 are reserved for records generated by IBM-supplied routines, while numbers 128 through 255 are available for identification of any additional record types created by a particular installation.

OS SMF writes 31 types of records, with numbers ranging from 0 to 42. Type numbers 16, 22 through 29, 36, 37 and 39 are unassigned. Records numbered 30 and above are produced only in systems with the Time-Sharing Option, and will not be discussed in this report. Record Type 13 is peculiar to OS/MFT or OS/VS1; since this report is confined to OS/MVT systems, discussion of record Type 13 is also omitted. The remaining 20 types, which are referred to in this report as "basic SMF records," can be grouped into system records, job step records and data set records. These are discussed in turn in the following sections.

2. System records

The system records are those SMF records containing only data elements describing some change of status in the operating system or the underlying hardware. Each of these will be mentioned briefly in this section; for a detailed presentation of the data elements contained in each record, refer to Appendix D.

The Type 0, or IPL, record is produced at IPL (Initial Program Loading) time, i.e., each time the system is started or restarted. It contains the maximum job wait time value and the SMF options derived from member SMFDEFLT of SYS1.PARMLIB, as well as the hardware-defined main storage size in kilobytes. Its time stamp serves to identify the start of a sequence of SMF records, and the beginning of a period of availability of the system to its users.

The Type 1, or Wait Time, record contains only one four-byte field beyond the 18-byte standard header. It gives the amount of time the CPU spent in the wait state during one or more ten-minute intervals of elapsed time.

As mentioned previously the Type 2 (Dump Header) and Type 3 (Dump Trailer) records are produced by the SMF Dump program and serve merely to delimit the output of the dump. These records each consist only of the standard header, with no additional data.

The Type 7, or Data Lost, record is produced when SMF recording begins again following a period of time during which no output data set was available. This record tells when the hiatus in SMF recording began and how many records were lost; through the date/time stamp in its standard header it also tells when the hiatus ended.

The Type 8, or I/O Configuration, record consists of the standard header plus one four-byte entry for every device on line at IPL time. The entry gives the device class, unit type, channel address and unit address.

The Type 9, or VARY ONLINE, record identifies the system resources being added to the configuration by a VARY ONLINE command. In addition to the standard header, it contains one four-byte entry for each CPU, channel, I/O device or core box brought online.

The Type 11, or VARY OFFLINE, record is identical to the Type 9 except that the resources it describes are being removed from the configuration rather than added to it.

The Type 12, or End-of-Day, record is written whenever a HALT EOD or SWITCH SMF command puts an end to recording of data on the currently active SMF data set. Apart from the type number, this record is identical to the Wait Time record.

The Type 19, or Direct-Access Volume, record provides space information for a direct-access volume which is mounted on a unit that is online and ready. It contains the volume serial number, VTOC address, owner identification, device type, number of unused alternate tracks, number of unallocated cylinders and tracks, size of the largest free extent, number of free extents, channel and unit address, and module identification for the 2314 and 3330. At IPL time and at the time a HALT or SWITCH command is processed, one Type 19 record is produced for each online, ready direct-access unit. When a user volume is dismounted, a Type 19 record is written for that volume only.

The Type 21, or Error Statistics by Volume, record is written whenever a user data set is processed by End-of-Volume, provided the ESV=SMF option was selected during system generation (see II.A.3 above). It provides statistics on temporary read and write errors, start I/O's, permanent read and write errors, noise blocks, erase gaps and cleaner actions. It also gives the channel and unit address and the four-byte UCB type field for the device, the recording density and serial number of the volume, and the block size of the data set being processed.

3. Job step records

Under this heading are listed the SMF records which describe the usage of resources by a particular job or job step.

The Type 4, or Step Termination, record is written at the end of each job step to record such resource-related data as elapsed time, CPU time, main storage allocated and used, number of Sysin records, I/O devices and EXCP counts. The job is identified by the job log number, which consists of the eight-byte job name taken from the JOB card and the eight-byte date/time stamp provided by the OS reader when it recognized the JOB card. The step within the job is identified by the eight-byte user identification field from the exit routines' common parameter area, the step number, and the eight-byte step name taken from the EXEC card. Also to be found in the Type 4 record are the step completion code, the dispatching priority, the name of the program specified in the EXEC statement, the device allocation time, and the accounting data, if any, from the EXEC card.

The Type 5, or Job Termination, record is written at the end of each job to summarize its resource usage. This record includes the start and stop time for processing of the job by the Reader/Interpreter, the job initiation time and date, the number of steps in the job, the number of card-image records in "DD *" or "DD DATA" data sets, an indicator of which Sysout classes were used by the job, a checkpoint/restart indicator, the device class and unit type of the reader device, the storage protect key, and the total CPU time used by the job. To identify the job, the job log number is accompanied by the user identification from the common parameter area, as well as the priority, input class, programmer name and accounting fields from the JOB statement.

The completion code for the last step of the job which was actually executed is augmented by a job termination indicator byte, which reveals whether one or more steps of the job terminated abnormally, and whether the job was cancelled by one of the user-written SMF exit routines.

The Type 6, or Output Writer, record is written when a writer has finished processing a Sysout class or a form within a class, for each job in the system. The job is identified by job log number and user identification field, the writer by Sysout class and form number. The Type 6 record includes the date and time the writer began processing this job, the number of records written per form number and class, an I/O error flag, and the total number of data sets processed by the writer for this job.

The Type 10, or Allocation Recovery, record is written whenever special action involving operator intervention succeeds in making a previously offline device available for allocation to a job step that needs it. A typical example of allocation recovery would be a sort/merge step requesting several large work files on separate direct-access devices, at a time when only a couple of drives have packs up containing much free space. The system sends a message to the operator asking for one of four replies:

* "WAIT" - this tells OS to suspend initiation of the sort/merge step until some other job step terminates, possibly freeing enough disk space to permit allocation.

* "NOSEP" - this tells OS to try to satisfy the sort step's space needs, ignoring the requirement that the work files be on separate devices.

* "CANCEL" - this tells OS to cancel the sort/merge job.

* A device name - this tells OS to bring the device specified by the three-character channel and unit address online (if it was offline originally), and then retry the allocation. To facilitate this response, the system accompanies the original message with a menu of devices that are currently offline.

If the last of these four possibilities results in successful allocation, a Type 10 record is written, identifying the device brought online by device class, unit type and device address. The job log number for the job requesting allocation, and the user identification field are also given in the Type 10 record. If allocation recovery was for a system task rather than a job step, the job name and user identification fields contain blanks and the job entry time and date fields contain binary zeros. Type 10 records also have the usual date/time stamp and CPU identification.

4. Data set records

These records, containing information on all tape or DASD data sets that are explicitly created, read, updated, extended, rewritten, scratched, or renamed in a user job step, are produced by SMF, provided DSV=2 or DSV=3 is specified in SMFDEFLT. While the EXCP counts in the step termination records reflect I/O activity per device for each job step, the data set records go beyond this to permit analysis of I/O activity by data set as well. Like the job step records, the data set records identify the job to which they refer by the job log number, consisting of the jobname, entry time and entry date. Most of them also contain the 44-character symbolic name (DSname) of a specific data set.

The principal data set records are the Type 14, or Close/EOV Input, record and the Type 15, or Close/EOV Output, record. These records are identical in format, structure, and function, differing only in the type number, which is 14 if the data set being closed or reaching end-of-volume was opened for input or read-backwards, or 15 if the data set was opened for output, update, input followed by output, or output followed by input. For brevity's sake, these records will be referred to collectively as "the Close/EOV records" in further discussion. They are the largest SMF records, ranging in length from a minimum of 288 bytes (316 for indexed sequential data sets) up to a maximum of 6412 bytes. For each unit beyond the first allocated to the data set, 24 bytes are added to the Close/EOV record; thus, the maximum length corresponds to the highly unlikely case of an indexed sequential data set occupying 255 volumes. Note that Close/EOV records are not written for "DD *" or "DD DATA" data sets, nor for Sysout data sets. JOBLIB and STEPLIB data sets also escape observation through Close/EOV records since they are opened and closed by the Initiator/Terminator, not by the job step itself.

The Close/EOV records contain a wide variety of data elements taken from various control blocks describing the data set (DCB, DEB, JFCB, TIOT, and UCB); in fact, the entire JFCB, comprising all the information from the DD statement for the data set, is included in the record. (In the case of a data set occupying more than five volumes, only the first five volume serial numbers are contained in the JFCB. The JFCB extension, containing the additional volume serial numbers, is not included in the Close/EOV record.) The records also contain the EXCP counts for each unit allocated to the data set. Two points are worth noting regarding the EXCP counts: first, they are accumulated on a job step basis, so that if a data set is closed, reopened and closed again in one step, the EXCP counts in the second Close/EOV record will include those in the first. The second point is that a similar situation can arise for multi-reel tape data sets which are processed at least once by End-of-Volume as well as by Close. If multiple reels are mounted on the same unit, the EXCP counts for that unit in successive Close/EOV records will be cumulative.

The Type 17, or Scratch, record is written whenever a data set is scratched. Depending on the number of volumes for the data set, the length of the record varies from 100 to 2140 bytes. Besides the job log and data set name, the Type 17 record contains the number of volumes and the volume serial numbers.

The Type 18, or Rename, record is written whenever a data set is renamed. It varies in length from 144 to 2184 bytes, again depending on the number of volumes. In addition to the volume serial numbers, it contains both the old and new data set names.

Although the Type 20, or Job Commencement, record is classified as a data set record, it does not refer to any data set, but resembles the job step records in that it contains information about a particular job. Its purpose is to correlate the other data set records with the job accounting fields so that I/O charges can be distributed to the jobs incurring them even if a system failure prevents the Job Termination record from being written. The presence of the Type 20 record also facilitates the analysis of SMF data when records for one job are split across two SMF data sets. The Type 20 record is variable length, and contains the job log, programmer name, user identification, system and model identification, and the job accounting fields.

C. Data Elements

1. Symbolic names and formats

Appendix D to this report consists of a tabulation of the individual data elements contained in the basic SMF records. For each record, this table tells which components of OS construct it and write it, i.e., issue SVC 83 to have it moved to the SMF buffer. For each field in the record, other than the standard record type and length fields, the table gives the symbolic name assigned by expansion of the IFASMFR macroinstruction, the format and length (e.g., PL4 for a four-byte packed field), a capsule description, the external source (e.g., the OS operator), and the source control block. It should be noted that the macro definition for SMF record layouts is not included in the OS/MVT system, but is distributed with OS/VS as member IFASMFR of data set SYS1.AMODGEN on volume DLIBA2.

The table also identifies the OS components which create the source control block, initialize the field and maintain the field. These last two columns (the fifth and sixth in the table) tell which components initialize and maintain the field in the source control block from which the data element is derived, unless the sixth column contains the notation 'N/A'; in that case, the fifth column tells which OS component initializes the field in the SMF record itself. This arrangement provides for a precise definition of most of the data elements supplied by SMF. However, some data elements require more discussion; this follows the Table of Symbolic Names as a series of paragraphs that are keyed to the appropriate data elements in the Table.

Appendix E to this report is an edited assembly listing of the expansion of the IFASMFR macroinstruction, including the cross-reference. In this listing the fields appear in the same order as they physically occur in the records, while in Appendix D the fields are listed alphabetically by symbolic name within each record. By using these Appendices together, the reader can find the format and source(s) of any standard SMF data element by searching either for its symbolic name or the record type in which it appears.

2. Sources and precision

The sources of most of the SMF data elements are tabulated in Appendix D, as noted above. For a few of these elements, further discussion of their derivation is warranted. Also in this section some of the abbreviations used in the "External Source" columns of Appendix D are explained in more detail.

Every field whose contents are a date or a time-of-day is shown in Appendix D as having "OS Operator or TOD Clock" as its external source. On a 360 system, the date and time-of-day are maintained by the Timer Second-Level Interruption Handler, using the hardware interval timer to measure the elapsed time from the reference value supplied by the operator as part of the IPL procedure. The precision of the interval timer will be discussed in the next section; at this point it need only be observed that the timer is sufficiently precise to make the operator the primary source of date and time-of-day values. On a 370 system, the built-in time-of-day clock is used to provide consistent timing across IPL's and periods of system downtime. This clock runs when the CPU is in the wait or stopped state or in the instruction-step, single-cycle, or test mode; in particular, it need not be reset after each IPL.

The interval timer on an IBM System/360 works by decrementing bit positions 21 and 23 of the fullword at location 80 (decimal) in low memory sixty times a second, based on the frequency of the AC power lines. Hence measurements made using this timer have an inherent error of plus or minus 17 milliseconds, approximately, per sample. In addition to this minimum error, other sources of imprecision may arise depending on the load on the system. The IBM System/360 Principles of Operation manual (<21>) states: "Timer updating may be omitted when I/O data transmission approaches the limit of storage capability, when a channel sharing CPU equipment and operating in burst mode causes CPU activity to be locked out, or when the instruction time for READ DIRECT is excessive. The program is not alerted when omission of updating causes the real-time count to be lost."

In normal daily operating practice, however, the most significant source of imprecision in timing values collected by SMF is operator action. While the CPU is in the stopped state, the interval timer is not decremented. The SMF records give no indication of when or for how long the operator stopped the

system to change disk packs or deal with other routine operating exigencies. Another problem is the fact that SMF keeps no record of the operator's SET command, which can be used to reset the value of the software-maintained time-of-day clock on the System/360. This can make steps appear to terminate before the jobs containing them were initiated; such anomalies complicate the analysis of SMF data.

In Appendix D, the reference to "hardware" as the source of the real storage size (SMFORST) means that this number is determined by the Nucleus Initialization Program at IPL by the use of the hardware feature that checks for use of non-existent addresses in instructions. By testing progressively higher memory locations and handling the program check interruption that eventually occurs, NIP can find the top of main storage and thus its size.

The contents of the user identification field in the JMR, which appears in the SMF Job/Step records, are shown in Appendix D as having "installation standard" as their external source. This is because that JMR field is intended to be filled in by user-written exit routines, the most likely candidate being IEFUJI, the job-initiation exit.

In the Type 19 record, the SMF19IND field contains a DASD module identification or drive number obtained from the hardware as follows: the SVC 78 routine executes a channel program containing a special sense command, which causes the 2314 or 3330 disk unit to return the desired value.

IV. Design of SMF

A. Components of SMF

SMF is implemented by a mixture of batch programs, user-written exit routines, OS modules devoted exclusively to SMF recording or data collection, data-collection code within other OS modules, and routines that initialize SMF functions at IPL time. These six types of SMF components will now be discussed in turn.

1. Batch programs

As part of the distributed material for OS, IBM supplies three batch programs specific to SMF, namely the SMF dump program, the ESV utility, and the user exit test driver. The SMF dump program consists of a single load module named IFASMFDP which may be executed as the main program in a job step or invoked as a subroutine of a user-written main program. It uses the Basic Sequential Access Method (BSAM) of OS Data Management to copy an SMF data set from direct-access storage (DDname DUMPIN) to a backup ("dump") data set (DDname DUMPOUT). The DUMPOUT data set is usually on magnetic tape. If no errors occur during copying, IFASMFDP resets the DUMPIN data set to an empty condition. Errors encountered while opening the data sets or reading and writing records cause a diagnostic message to be written to DDname SYSPRINT; then all files are closed and a code of 16 is returned to the caller.

Before opening the DUMPIN data set, IFASMFDP uses the RDJFCB macroinstruction to read the Job File Control Block, which contains information derived from the DD statement. The JFCB is used to determine whether DUMPIN specifies the currently active SMF data set; if so, the program writes a message to the operator's console informing him of the error and requesting a reply of 'CANCEL'. When this reply is entered, the program acknowledges its receipt via another console message and terminates with a return code of 16.

IFASMFDP determines that DUMPIN is the active data set whenever (1) the first eight characters of the data set name are 'SYS1.MAN'; (2) the ninth character agrees with the SMCAXORY field in the SMCA; (3) the first volume serial number in the JFCB agrees with the volume serial number of the active data set as recorded in the SMCA; and (4) it is not the case that both SMF data sets are full.

When IFASMFDP succeeds in opening the DUMPIN and DUMPOUT data sets, it writes a Type 2 (Dump Header) SMF record to DUMPOUT. Next, it issues a series of READ, WRITE, and CHECK macros to copy one block at a time from DUMPIN to DUMPOUT, until end-of-file is reached on DUMPIN. Then a Type 3 (Dump Trailer) record is written to DUMPOUT and both files are closed. Finally, DUMPIN is opened for output and immediately closed, so that its final status is empty.

The ESV utility program, IFHSTATR (<6>), given SMF data as input, produces a report on tape error statistics by formatting Type 21 records into 120-column print lines. The user exit test driver, TESTEXIT (<2>), is supplied in assembly-language source form, would normally reside in a user library, and is intended for use in debugging user-written SMF exit routines. With the aid of this program, user exit routines can be tested in a batch setting even though they will eventually run as part of the operating system. The test facility provided by TESTEXIT does not allow convenient simulation of a real OS environment, however; the main objection to it raised in <39> is that calls to various exits cannot be interspersed as they would be in normal operation.

Two PL/I programs are available from IBM for the analysis and graphic presentation of SMF data. One is SMFPOST (<2>, pp. 35-37), distributed as a member of the source library SYS1.SAMPLIB; it merely formats and prints each SMF record whose type number is less than 14. It is not intended for production use, but rather as an example of how to begin constructing an SMF analysis program in PL/I. The second is the SMF Selectable Analyzer (<28>), a field-developed program not distributed as part of OS but available on an "as is" basis by special request. This program handles all basic SMF records and produces 13 varieties of reports. These include detailed and summary reports, histograms and a generalized accounting routine. However, the vendor makes no warranty as to correctness of this program, nor does the vendor accept any maintenance responsibility. Thus it is realistic to assume, as the SMF manual does, that each installation using SMF must determine what reports it wants and where the programs to produce these reports will be obtained.

In addition to these SMF-specific modules, IBM supplies a variety of utility programs that could be used in moving, copying, or restructuring the SMF files (<6>). TESTEXIT uses one of these utilities (IEBDG) as a sub-program to generate simulated test data to be passed to user exits.

2. User-written exit routines

Interfaces to six user-written routines are provided in the OS control program as part of SMF. Whether these exit routines are actually invoked during operation of the system is determined by the EXT and OPT parameters in the SMFDEFLT member of SYS1.PARMLIB, which may optionally be overridden by the operator at IPL time. If EXT=NO, no user exits are taken. If EXT=YES and OPT=1, only the exit routines named IEFUJV, IEFUJI, IEFUTL and IEFACTRT are taken. If EXT=YES and OPT=2, exit routines IEFUSI and IEFUSO are used, as well as those named above; in addition, IEFUTL and IEFACTRT are invoked on a step-by-step rather than a job-by-job basis.

In general, these exit routines make it possible to modify or suppress standard SMF records, write installation-defined records to the SMF data set, or scan each user's JCL for adherence to local conventions and enforce these by penalizing non-complying jobs.

Vendor support for these routines includes documentation of the interfaces to them (<2>) and provision of a macroinstruction (SMFWTM) to facilitate writing of user-defined SMF records. Sample versions of four of the six exit routines, namely IEFUJV, IEFUJI, IEFACTRT and IEFUTL, are supplied as members of the distribution source library SYS1.SAMPLIB. As mentioned above, IBM also provides facilities for testing user-written exit routines in a batch environment.

a. IEFUJV - JCL Validation

This exit in the OS Reader/Interpreter is taken each time a non-null JCL record (card image) is read into the system, and also just before a job is placed in the work queue to await execution. The routine can inspect and modify each JCL record before it is interpreted, and can cause a job to be cancelled before it is enqueued for execution.

b. IEFUJI - Job Initiation

This exit in the OS Initiator/Terminator is taken whenever a job is selected for initiation. It has access to the programmer name, accounting and priority fields from the JOB statement in the JCL for the job. By returning a code of 4 to its caller, it can have the job cancelled.

c. IEFUSI - Step Initiation

This exit in the Initiator/Terminator is taken just before each job step is initiated. Among the parameters it is passed are the step name, the program name, and the accounting information from the EXEC statement in the JCL for the job step. It has the option of cancelling the remainder of the job.

d. IEFUSO - Sysout Limit

This exit is scheduled by the I/O Supervisor whenever the limit on the number of records to be written to a Sysout data set is exceeded. Such a limit exists only when the OUTLIM=nnn parameter is specified on the DD statement defining the data set; this should not be confused with the practical limit imposed by the finite amount of auxiliary storage space allocated to the data set. The IEFUSO routine may cause the output limit to be extended or may allow the standard system action, which is to terminate the job step abnormally.

e. IEFACTRT - Termination

This exit in the Initiator/Terminator is taken whenever a job or a job step is terminated. Its interface to the system is more elaborate than those for the other user exits. One of the many parameters passed to it by the Initiator/Terminator is the SMF Type 4 (step termination) or 5 (job termination) record, which it may inspect and alter. The IEFACTRT routine may also:

* prevent the SMF Type 4 or 5 record from being written;
* write a user-defined record to the SMF data set;
* write a message to the user-specified system message class (this would cause it to be printed along with the JCL and Initiator/Terminator messages for the job);
* cancel the job.

f. IEFUTL - Time Limit

This exit from the Timer Second-Level Interrupt Handler is taken whenever a job, step or wait time limit expires; a code is passed in a register to tell the user's routine which of these limits was exceeded. IEFUTL may grant an extension of the time limit, or it may allow the standard system action (abnormal termination of the job step).

3. SMF recording modules

The SMF recording function is performed by two routines, the SMF writer and the SVC 83 routine, which communicate with each other via WAIT and POST macroinstructions. When an OS component or user-written exit routine has constructed an SMF record, it places a pointer to the record area in general-purpose register 1 and issues SVC 83; this linkage is provided by the SMFWTM macroinstruction. The SVC 83 routine copies the record into the SMF buffer, and when the buffer is half full it signals the SMF writer to output one block to the SMF data set. The operation of these two routines will now be outlined in turn; for more detailed descriptions, the reader is referred to the Job Management Program Reference Manual (<5>) or the microfiche listings of the modules.

a. The SMF Writer

The SMF writer task, which runs as a subtask of the Master Scheduler, performs the functions of record transfer (from the SMF buffer to the SMF data set), data set switching, space analysis, and initialization of the SMF data sets. This task is executed by the SMF writer routine, module name IEESMFWR. It communicates with the SVC 83 routine via the SMCA, primarily through two ECB's: the writer ECB and the buffer ECB. The writer routine waits on the writer ECB; when that ECB is posted,

the writer tests flag bits in the SMCA to determine what function is required, performs it, posts the buffer ECB to signal completion of that function and goes back to waiting on the writer ECB again.

The space analysis function of the writer routine serves to detrmine whether the currently active SMF data set contains enough space for a record that must be written in segments. This guards against the possibility of a record being split across data sets with the eventual result being the occurrence of a Type 3 and a Type 2 record between successive segments of a longer record.

The SMF writer uses BSAM WRITE and CHECK macroinstructions to transfer records from the buffer to the active data set. When the CHECK macro determines that the WRITE was successful, IEESMFWR issues a CLOSE macroinstruction specifying the TYPE=T parameter. This temporarily closes the data set by writing an end-of-file mark following the last record and updating the last-track-written and track-balance fields in the DCB and the DSCB. Note that a temporary CLOSE (with TYPE=T), unlike a standard CLOSE, does not cause a Type 14/15 SMF record to be written; consequently, there is no recursion between Open/Close/EOV and the SMF writer.

An I/O error return from the CHECK routine is treated the same as running out of space on the active data set, namely, by switching to the reserve data set and requesting the operator to dump the full data set. Abnormal termination of the BSAM routines is handled by means of a STAE retry routine, which quiesces activity in the system by requesting exclusive control of the work queue, and then issues a console message informing the operator of the ABEND. When the operator replies to this message, the SMF writer task then goes into a permanent wait.

b. The SVC 83 routine

The SVC 83 instruction is issued (as part of the expansion of an SMFWTM macroinstruction, or directly) whenever an OS module or user exit routine wishes to move an SMF record from the work area in which it was built to the SMF buffer. In addition to servicing these requests, the SVC 83 routine performs initialization and housekeeping functions. The SVC 83 routine consists of three modules: IEESMF8C (the record transfer routine), IEESMFOP (the Open routine), and IEESMFAL (the allocation routine).

The record transfer routine is the first load of SVC 83, and as such is also referred to as IGC0008C. If register 1 contains a pointer to the SMCA, control is transferred to IEESMFOP; this happens at initialization time, when the SMF writer requests opening of the SMF data sets. The Open routine in turn passes control to the allocation routine, which finds the UCB for the first data set, marks it as allocated and permanently resident, and transfers control back to IEESMFOP again. At this time

IEESMFOP places a pointer to the UCB in the Master Scheduler TIOT and opens the data set. If the SMF data sets are on DASD, the allocation module is invoked a second time. After opening the second data set to be allocated, the Open routine must determine which of the data sets SYS1.MANX and SYS1.MANY is to be used first. If both already contain data, the one containing less (in absolute terms, not proportionally to the size of the data set) is chosen and the operator is requested to dump the other one. If just one data set is empty, the non-empty one is selected; if both are empty, SMF recording begins on SYS1.MANX. Once this determination has been made, the Open routine notifies the operator of its choice and exits to its caller.

When register 1 does not point to the SMCA at entry to the record transfer module, that module tests the miscellaneous indicators field in the SMCA to determine whether both SMF and user records, user records only, or no records at all are to be written to the SMF data set. On this basis, it either exits immediately to the Supervisor or attempts to move the record to the SMF buffer. At this point, the data and time stamp (acquired via the TIME macroinstruction) and the system ID (from the SMCA) are filled into the record, unless it is a job or step termination or a user record.

The attempt to move the SMF record to the buffer begins with the record transfer routine issuing an ENQ macroinstruction. The major and minor queue names (SYSSMF01 and BUF) represent the SMF buffer; the ENQ prevents two concurrently executing tasks from attempting to move records into the buffer simultaneously. If the record fits in the space remaining in the buffer, it is moved there; otherwise, the record transfer module posts the SMF writer ECB in the SMCA to request that the current contents of the buffer be written out, and waits on the buffer ECB for a signal that the buffer is free. If the record will now fit in the empty buffer, it is moved there. Once the record has been moved and the available-space counter updated, the record transfer module issues the DEQ macroinstruction to permit other tasks to use the buffer; then it exits to the Supervisor.

The description of the SVC 83 routine in the OS/360 MVT Job Management PLM (<5>) disagrees with the microfiche assembly listing in one noteworthy respect. The discrepancy involves the testing of the contents of register 1 at entry to the record transfer routine. This test determines whether a record transfer actually is performed or whether, instead, control is transferred to the SMF Open routine.

According to <5>, the record transfer is performed if register 1 is positive at entry to IGC0008C. The test actually made is a logical comparison of the absolute value of the contents of register 1 with the contents of the SMCA pointer field in the CVT. If these values differ, the record transfer is performed; if they agree,IEESMFOP is given control to carry out initialization or data set switching. The rationale for this appears to be that the address of the record to be transferred is

normally placed in register 1 just before SVC 83 is issued to move the record to the buffer. Such an address might or might not appear negative, but it would never equal the address of the SMCA.

The reader is referred to <5> for a description of the processing performed by the SVC 83 routine and the SMF writer when the record to be moved must be segmented because it will not fit in the empty buffer.

4. SMF data collection modules

Several modules of OS are entered only when SMF is being used; they serve to build specific types of SMF records, provide linkage to user-written exit routines, and collect data for later recording. Most of these modules are discussed briefly in subsection a. below; the EXCP counting routine, the SMF storage routines and the system wait time collection routine are covered in greater depth in subsections b., c., and d.

a. Record construction and exit interface routines

The SMF VARY record handler, IGC2303D, which builds the Type 9 (VARY Online) and Type 11 (VARY Offline) records, is executed as part of the processing of the VARY command by the Master Scheduler.

Two modules of Open/Close/EOV are dedicated to SMF data collection functions. Construction of the Type 14/15 record during closing or end-of-volume processing of a data set is begun by module IFG0202H; it is also completed by that module unless the data set resides on a direct-access device, in which case module IFG0202I is invoked to finish building the record.

As part of the servicing of RENAME (SVC 30) requests by the DADSM routines, the SMF record Type 18 processing module, IGC03003, is entered to construct the record and have it moved to the buffer. The Type 18 record contains the old and new data set names, and the serial number of each volume on which the data set resides.

When SMFDEFLT specifies DSV=2 or 3, and the job was not cancelled by the user-written initiation exit, a job start record (Type 20) is constructed for each job by the user exit initialization routine, IEFSMFIE. This module performs several other SMF-specified functions each time a job step is selected for initiation. First, it determines whether a Timing Control Table (TCT) must be built, i.e., whether this is the first step of the job. If so, IEFSMFIE acquires main storage for the TCT and the first forty bytes of the Job Management Record (JMR). After initializing the TCT, IEFSMFIE reads the JMR from the work queue data set and copies its first forty bytes for use as the SMF Common Exit Table, which is passed as the first parameter to every user exit routine. If exits were specified, IEFSMFIE reads the job's ACT into main storage and passes control to the job

initiation exit routine, IEFUJI; otherwise, it builds the Type 20 record and has it moved to the SMF buffer.

When IEFSMFIE determines that the TCT is already in existence, it updates the time stamp in the JMR; if user exits are to be taken, it then reads the step's ACT into main storage, and passes control to the step initiation exit routine, IEFUSI. If the return code from IEFUJI or IEFUSI indicates that the job is to be cancelled, IEFSMFIE sets the job-failed bit in the JCT.

Another module of the initiator which is bypassed when SMF is omitted from the system is the TCTIOT construction routine, IEFSMFAT. As its name implies, this module obtains main storage for the Timing Control Task Input/Output Table, which is used to collect EXCP counts and other data set statistics. It is invoked by the initiator's attach routine (module IEFSD263) just before a job step task is attached. By that time, devices have been allocated and a region acquired for the step. IEFSMFAT finishes filling in the TCT with the following information:

* lowest address allocated at the high end of the region;
* highest address allocated at the low end of the region;
* the amount of unused storage in the region;
* the wait time limit;
* the address of the user time-limit exit routine, IEFUTL;
* the address of the Sysout limit exit routine, IEFUSO;
* the address of the TCTIOT; and
* the current time and date, to record when loading of the problem program was requested.

Several SMF functins are performed at step and job termination time by the terminator's user accounting linkage routine. The source module for this routine is named IEFSMFLK, but its entry point is IEFACTLK. Before SMF was added to OS, this module was the interface to the original user accounting exit. The first operation attempted by IEFSMFLK when it gets control is the construction of the Type 1, or Wait Time, record. The routine checks a field in the SMCA (field name SMCAWAIT) into which cumulative wait time values are added every ten minutes by the Timing Supervisor. If SMCAWAIT contains zero, no Type 1 record is written at this time; otherwise, IEFSMFLK builds the Type 1 record, moves the wait time into it, and zeroes out the SMCAWAIT field before issuing SVC 83 to write the Type 1 record. It next reads the proper Accounting Control Table (ACT) into main storage, determines the maximum amount of core used, and then invokes the SMF termination record construction routine, IEFSMFWI, to build the Type 4 or Type 5 record. After IEFSMFWI returns control, IEFSMFLK puts together the parameter list for the user's termination exit routine, IEFACTRT, and then calls it. Depending on the return code from IEFACTRT, IEFSMFLK may cancel the job, omit writing of the Type 4 or 5 record, or issue SVC 83 to have the record moved to the SMF buffer.

The SMF time/output limit expiration routine, IEATLEXT, resides in the nucleus and serves as an interface to both the user's time limit exit routine, IEFUTL, and the user's Sysout limit exit routine, IEFUSO. It runs under the control of an IRB/IQE created by either the timer Second-Level Interrupt Handler (SLIH) or the EXCP counting routine, IEASMFEX, depending on which limit expired. The SMF-related activities of the timer SLIH are described in section IV.A.5, while IEASMFEX is discussed in subsection b., immediately below.

In the case of time limit expiration, IEATLEXT sets up a register save area and the proper parameter list and calls IEFUTL, with register 0 set to 0, 4, or 8 to indicate which limit was exceeded: job CPU time, step CPU time, or job wait time. The exit routine may return a code of 4 in register 15 and a number of timer units in register 1; if so, IEATLEXT does the bookkeeping necessary to extend the expired limit by the given number of timer units. If IEFUTL returns a code of 0, IEATLEXT schedules the job step to be abnormally terminated with a code of 322 for too much CPU time or 522 for too long in the wait state.

In the case of Sysout limit expiration, IEATLEXT again sets up a save area and a parameter list and calls IEFUSO, which may return a code of 4 in register 15 and a count of additional blocks permitted in register 1. If it does, IEATLEXT adds the specified value to the output limit field in the appropriate TCTIOT entry. Otherwise, IEATLEXT schedules the job step for abnormal termination with a code of 722.

b. The EXCP counting routine

This routine, whose entry point name is IEASMFEX, resides in the nucleus in module IEAQNU00. It is called by the EXCP validity checking routine of the I/O Supervisor whenever the control blocks governing an EXCP (SVC 0) or an error EXCP (SVC 15) appear acceptable to the IOS. It is also called by the I/O Interruption Supervisor immediately upon return from the PCI Appendage routine in the event of a Program-Controlled Interruption (PCI). This fact is significant for I/O operations that use chained scheduling, since under OS such operations cause one PCI to occur for each block read or written. Chained scheduling and its significance for I/O accounting and performance measurement under both OS and MVS (VS2 Release 2 and later releases) are discussed in Appendix G to this report; the effects of chained scheduling on EXCP counts and CPU time variability are discussed in section VI.

When IEASMFEX is entered, it performs a complex series of tests to determine whether it should count the EXCP (or PCI). The first few of these deal with the situation where I/O was requested by an SVC routine: their effect is to count the EXCP only if the issuing routine is servicing an OPEN, CLOSE, OPEN TYPE=J or CLOSE TYPE=T request made by a problem program. When the issuing routine is itself a problem program, IEASMFEX will check the DCBOFLGS field of the DCB to ensure that the data set

is either open or in the process of being opened or closed. If the corresponding flag bits in this field are both zero, the EXCP is not counted. Since these flags are accessible to the problem program (the DCB is not copied into key 0 under OS) and since resetting the "open" flag has no effect on normal I/O operations other than inhibiting EXCP counting, this test constitutes an open invitation to cheat any accounting system which charges for I/O on the basis of EXCP counts.

Following these tests, IEASMFEX checks a field in the current task's TCB for the presence of a pointer to a Timing Control Table (TCT). (See sections IV.C.5, IV.C.6 and IV.C.7 below.) If the current task is a system task, this field will contain binary zeroes, and the EXCP will not be counted. Otherwise, the field in the TCT that should point to a TCTIOT is tested; if it is zero, EXCP counting is not in effect, and IEASMFEX returns control to the I/O Supervisor.

At this point, the decision to count the EXCP has been made, and the TCTIOT is searched for the entry containing the counter to be incremented. By referring to the description of the TCTIOT in section IV.C.6 below, the reader can see that a two-level associative search is required. First, the DD entry must be located by matching the TIOT offset in the DCB against a list of halfword fields in the TCTIOT; then the DD entry itself must be searched for a subentry whose UCB address field correspondonds to the unit on which I/O was requested. The time required to perform these searches is the sum of three terms. The first is constant, the second is proportional to the number of DD statements preceding the one for this data set, and the third is proportional to the number of devices for this data set preceding the one on which I/O is being requested.

After incrementing the proper EXCP counter, IEASMFEX rescans the entire DD entry summing the counts for all units to get a total EXCP count for the data set. Then it tests the output limit field, and if it is non-zero, compares it with the total EXCP count. Exceeding the Sysout limit causes IEASMFEX to create an IRB/IQE for the time/output limit expiration routine (IEATLEXT) and schedule it for execution. However, if SMF exits are not to be taken, exceeding the Sysout limit causes IEASMFEX to terminate the job step abnormally with a code of 722.

The cost of this Sysout limit checking could be considerably reduced by testing for a limit of zero, i.e., a non-Sysout data set, before computing the total EXCP count. There is no need for this total unless the data set is Sysout.

c. The SMF storage routines

The two SMF storage routines comprise a module named IEASMFGF which maintains core usage information in the TCT (timing control table). When SMF is active in the system, they are called as subroutines by the resident SVC routines which service GETMAIN and FREEMAIN requests. If no TCT pointer is

present in the TCB for the requesting task, the SMF storage routines return control immediately to their caller.

The function of these routines is determined by the structure of the TCT, which in turn depends on the way GETMAIN/FREEMAIN works. Storage within a region is allocated from the top down and from the bottom up, so that the largest contiguous area of free storage is typically (but not necessarily) in the middle of the region. Furthermore, the 360/370 machine architecture forces storage to be converted from key 0 to the requestor's protect key in multiples of 2048 bytes. Thus the unallocated key 0 core in the middle of the region is always an exact number of 2K blocks in size, as is the region itself. The low and high ends of this central free area are referred to in IBM documentation as the "low water mark" (LWM) and the "high water mark" (HWM) respectively. Their difference, a multiple of 2K, gives the size of the central free area; SMF maintains the minimum value of this difference in the TCT. At step termination, this minimum value is subtracted from the region size to derive the core-used figure, SMF4HOST.

When core is requested within a region, the GETMAIN routine first inspects its storage control queues to determine whether the amount wanted is available and, if so, where it should be allocated. Then it calls its SMF storage subroutine at entry point GMSMFCRE within IEASMFGF. If a TCT is present, GMSMFCRE checks for a new LWM or HWM and updates the proper TCT entry as needed. It then takes the diference between the HWM and the LWM in units of 2K blocks, and updates the minimum difference in the TCT if the new value is lower. Likewise, when core within a region is freed, the FREEMAIN routine calls IEASMFGF at entry point FMSMFCRE to update the HWM or LWM if necessary.

d. The system wait time collection routine

This nucleus-resident routine, named IEAQWAIT, works in conjunction with the OS Dispactcher and the Timer SLIH to measure and accumulate the intervals of time that the CPU spends in the wait state. This process is entirely distinct from job wait timing which uses the separate mechanisms described in the next section.

When no task in the system is ready to use the CPU, the Dispatcher sets up the system wait pseudo-task as the current TCB and "dispatches" it by loading a PSW with the wait bit on. If SMF is in effect, the Dispatcher saves the value of the interval timer in the first word of a special two-word save area, SYSWSAVE, just before loading the wait-state PSW. When an I/O or external interruption reactivates the CPU, the first-level interruption handler calls IEAQWAIT. If the current TCB is not the system wait pseudo-task, IEAQWAIT returns control immediately; if it is, IEAQWAIT reads the interval timer and subtracts the first word of SYSWSAVE from it to determine the number of timer units spent in the wait state. This number is then added to the value in the second word of SYSWSAVE.

When a supervisor 10-minute time interval expires, the timer SLIH adds the accumulated wait time in the second word of SYSWSAVE into a field in the SMCA (System Management Control Area) and resets the second word of SYSWSAVE to zero. The system wait time thus passes through two intermediate areas, which may be thought of as accumulators or buffers, on their way to the Type 1 or Type 12 record. SYSWSAVE is "filled" by IEAQWAIT and "emptied" by the timer SLIH; SMCAWAIT is "filled" by the timer SLIH and "emptied" by the user accounting linkage routine, IEFSMFLK, of the Terminator when it builds a Type 1 record. Both "buffers" are "emptied" by the Halt/Switch command processor, IEE1403D, when it builds a Type 12 record.

5. Data-collection code within OS modules

Six of the basic types of SMF records are constructed by code within OS modules that also serve other purposes. Besides these modules, several others are involved in the creation and maintenance of data items that eventually find their way into SMF records. One of the OS components most intimately involved with SMF data collection is the Timing Supervisor, also known as the Timer SLIH. It and the OS Dispatcher are discussed below in subsection a. The construction of the Type 6 record by various modules of the OS output writer is described in subsection b, while the other data-collection functions are lumped together in subsection c.

a. Job step timing

SMF job step timing uses the STIMER/TTIMER facility provided by the Timing Supervisor to measure the CPU time used by all tasks in the job step. The Timer SLIH is also concerned with preventing non-system tasks from remaining in wait status for longer than the maximum time specified by the JWT parameter in SMFDEFLT. Before the initiator attaches a job step, it issues an STIMER request specifying the amount of CPU time allowable for the job step. The Timing Supervisor services this request by creating a timer queue element (TQE) of task type, with a special flag indicating that an initiator issued the STIMER.

The Timer SLIH uses TQE's to perform a wide variety of timing functions by means of the single hardware interval timer built into the IBM 360 CPU. When a TQE is placed on the timer queue (this may be thought of as starting its clock running), the interval specified in the STIMER macroinstruction is converted to an absolute time-of-expiration, which determines the TQE's position on the queue. The value in the interval timer is always the difference between the current time and the nearest time-of-expiration. When this goes to zero, an external interruption occurs; the external SLIH recognizes its source as timer expiration and gives control to the Timer SLIH. The SLIH removes from the timer queue its topmost element, which represents the interval just expired, and takes appropriate action depending on its type (and on whether it belongs to an initiator). Meanwhile, the interval from the current time to the

time-of-expiration of the next TQE on the queue is placed into the timer, which continues to run.

For CPU timing, the TQE is task-type, and such TQE's require the cooperation of the OS Dispatcher to ensure that their "clocks" run only when the corresponding tasks have control of the CPU. When a task switch occurs, the Disptacher must deactivate any task-type TQE's for the old task and activate any that belong to the new task. A system task will generally have no TQE's; an ordinary job step task or subtask will have one; and a job step task or subtask that has issued a task-type STIMER of its own will have two, including the one belonging to its initiator.

Deactivating a task-type TQE involves converting its time-of-expiration to a time-remaining interval and removing the TQE from the timer queue. Reactivating it is just the reverse. Both actions are accomplished by a call from the Dispatcher to a routine within the Timing Supervisor which subtracts or adds the current time-of-day to convert expiration times to intervals or vice versa. Note that the time-of-day is derived from the current value of the interval timer, which may change between successive calls by the Dispatcher. This can lead to some peculiar results when task times obtained by programs that measure their own CPU usage via STIMER are compared with SMF-reported step CPU times.

When a TQE belonging to an initiator is deactivated, it is not just set aside, but rather is coverted to a specially flagged TQE with a real (not task) time interval of JWT. Thus the same TQE is used to monitor cumulative CPU time and maximum wait interval for the job step. If the wait or CPU interval expires, the Timer SLIH sets up linkage to the SMF time/Output Limit Expiration routine by converting the TQE into an IRB/IQE for IEATLEXT.

Finally, as mentioned in the previous section, the Timer SLIH maintains the supervisor 10-minute interval which governs transfer of accumulated system wait time from the special save area SYSWSAVE to the SMCA field where the terminator looks for it. The TQE corresponding to this interval is assembled into the nucleus of a system with SMF. When it expires, the Timer SLIH performs the above-mentioned transfer, resets the time-of-expiration to ten minutes hence, and puts the TQE back on the timer queue.

b. Data collection in the output writer

The Type 6, or Output Writer, record is built by the DSB handler, module IEFSD085; in a system with HASP, the Type 6 record would usually be written by the HASP print processor. Module IEFSD085 is the component of the system output writer which receives control when a Data Set Block (DSB) is read from the output queue entry for a job. The DSB describes a system output (Sysout) data set which has been written on direct-access

storage to await transcription to an output device. In a system with SMF, the last 28 bytes of the DSB contain information similar to that contained in the first 28 bytes of the Job Management Record. This "job log" portion of the DSB forms the basis for the Type 6 record.

When the OS output writer is preparing to transcribe an outpput data set, the DSB handler determines whether the form number has changed from that for the previous data set and whether the Sysout record count is nonzero. If both these conditions are satisfied, it copies the job log from the previous DSB into the Type 6 record area, fills in the Sysout class and form number, uses the TIME macroinstruction to supply the writer stop time and date, and issues SVC 83 to have the record moved to the SMF buffer. Then the DSB handler moves the stop date/time stamp to the writer start date/time field in the record area and zeroes out the Sysout record count and data set count fields. This initializes the Type 6 record area for the next form number to be processed.

A Type 6 record is always written by the output writer's job delete routine, module IEFSD079, after the last Sysout queue entry for a job has been processed. The job delete routine is also entered, and also writes a Type 6 record, when Sysout processing for a job is suspended because of an I/O error on the output data set. In this case, the data set delete routine, module IEFSD171, puts the job back in the queue, with a special System Message Block (SMB) inserted in front of its queue entry. This SMB points to the DSB for the data set that was being processed when the error occurred. When the next attempt is made to write the output for that job, it will begin with that data set and skip over all SMB's and DSB's already processed. Note that in such cases there will be more than one Type 6 SMF record for the same form number, Sysout class, and job log.

Several modules of the output writer contribute data elements to the Type 6 record. The data set delete routine (IEFSD171) adds 1 to the data set count just after scratching a Sysout data set that has been successfully transcribed. The put routine, module IEFSD089, increments a counter in the work area containing the Type 6 record every time it sends a Sysout record to the output data set. The job delete routine (IEFSD079) consolidates the I/O error indicators supplied by other routines into a one-byte field. The standard writer routine, module IEFSD087 does its SMF processing just before closing the input data set; this consists of flagging any input errors, updating the Sysout record count from the counter used by the out routine, and resetting that counter to zero.

If a user-specified alternative program is used to write system output, as requested by the second positional parameter on the DD statement (<7>), the contents of the SMF Type 6 record obviously depend on what that program does to the work area containing the Type 6 record image. The OS SMF manual (<2>) states that an incomplete Type 6 record is written for each

output class, but not for form changes within an output class. It further states that the numbers of logical records, I/O status indicators, and form number fields are not provided.

c. Other data-collection code

The Type 10, or Allocation Recovery, record is built by the I/O device allocation recovery routine, module IEFXJIMP, whose main purpose is to inform the operator of the recovery options available and give control to the appropriate routine to process his response.

The Type 12, or End-of-Day, record is built by the HALT command processor, module IEE1403D; its format is identical, apart from record type, with the Wait Time (Type 1) record. In response to a "HALT EOD" or "SWITCH SMF" command from the console, the HALT commmand processor issues SVC 78 to create a Type 19 record, and later builds the Type 12 record and issues SVC 83 to have it moved to the SMF buffer.

The Type 17, or Scratch Data Set Status, record is built by the Format 4 DSCB updater (module IGG0290D of the SVC 29 routine) as part of the processing of a SCRATCH request for a user data set.

The Type 19, or Direct Access Volume, record is built by the second load (module IGC0107H) of the LSPACE (SVC 78) routine if the contents of register 1 at the time SVC 78 was issued indicate a request for SMF information. This SVC is also issued as part of the processing of the "DISPLAY SPACE" console command; in this case, the data gathered by the LSPACE routine is used in constructing the reply to the operator, and no SMF record is created. The SMF version of SVC 78 is issued by the SMF initialization program, by the HALT command processor and by the disposition and unallocation message routine of the terminator (IEFZHMSG).

There is SMF-related code in the Attach routine (IGC042) to copy the TCT address from the attacher's TCB to the created TCB. This ensures that all subtasks of a job step task will share its TCT and its TCTIOT, and hence all I/O and main storage requests they make will be charged to the job step.

There is also SMF-related code in the Wait routine (IGC001) which performs a minor but necessary part of the job step timing function. If SMF is in the system, this routine places the wait time limit taken from the TCT (originally derived from the JWT parameter in SMFDEFLT), rather than the 30-minute default, into the TQE for the iniitiator's STIMER.

6. The SMF initialization program

When the operator presses the LOAD button on the control panel, the IPL (Initial Program Load) and NIP (Nucleus Initialization Program) procedures respectively load into main

memory and initialize the core-resident portions of OS. NIP passes control to the Master Scheduler, one function of which is to initialize the SMF data collection and recording process. The part of the Master Scheduler which performs this function is called the SMF initialization program. It consists of four modules: IEESMFIT, IEESMFI3, IEESMFI2 and IEESMFOI.

This program first scans the directory of the partitioned data set SYS1.PARMLIB, looking for a member named SMFDEFLT. If it fails to find this member, the SMF initialization program writes a message to the operator requesting entry of the SMF parameters from the console. If SMFDEFLT is present, it is read into main memory and checked for validity. If any of the parameters is invalid, or if SMFDEFLT specifies OPI=YES, the operator is invited to redefine the parameters. On the basis of parameters so determined, the SMF initialization program constructs the System Management Control Area (SMCA) from which many SMF modules will later extract information.

Provided that recording of SMF data has not been suppressed by use of the MAN=NONE parameter, the initialization program next checks the specification of the SMF data sets in the PRM and ALT parameters. If PRM is omitted, or if ALT is omitted when the primary data set is on direct-access storage, an error message is written to the operator and recording of SMF data is suppressed. This procedure is also followed if a specified I/O device is not availabe, or if a data set specified on a direct-access device has no space allocated for it. If none of these conditions arise, the initialization program creates the SMF writer task and activates its data set initialization functions. It, in turn, issues SVC 83 with register 1 pointing to the SMCA, to request opening of the SMF data set(s).

B. Interfaces between SMF and OS

The diagram on the foldout sheet at the back of this volume illustrates the interfaces between SMF and OS components in a hierarchical fashion. The diagram represents the tree structure of OS/MVT with all the branches having nothing to do with SMF pruned away. The only exceptions to this are Catalog Management and Access Methods, which are included primarily to emphasize that their activities are not directly measured by SMF.

The tables used to collect SMF data and the construction of SMF records are not shown on the diagram; they are illustrated and described in section IV.C. The diagram on the foldout does show double arrows leading from modules that write SMF records (i.e., issue SVC 83) to the records that they write. Note that these are not necessarily the same modules that construct the records.

C. Internal flow of data within SMF

SMF-related data is found in a wide variety of control blocks and tables throughout OS. Some of these are well-known, at least to systems programmers, and are described in detail in the System Control Blocks manual (<26>); these include the TCB, the DCB, the UCB, the DEB, the CVT, the JFCB, and the DSCB's that comprise the VTOC. A few SMF data elements are derived from fairly obscure corners of the system; these are grouped together at the end of this section under the heading "Other SMF data sources". The most important repositories of SMF data are the six tables discussed in the next six subsections below. The accompanying figures illustrate the flow of data through these tables, while subsection 7 shows how they are linked to each other and to standard OS/MVT control blocks by means of pointers. For a more detailed attribution of data element sources, see Appendix D.

1. Job Accounting Control Table

The Job ACT is created, partially filled in, and written out to SYS1.SYSJOBQE by the JOB statement processing routine of the Reader/Interpreter. It is read back in by the Initiator's Task Delete routine at the end of each step, updated, and written back to the Job Queue. It is also read in during initiation of the first step for inspection by user exit routines and for use in building the Type 20 record.

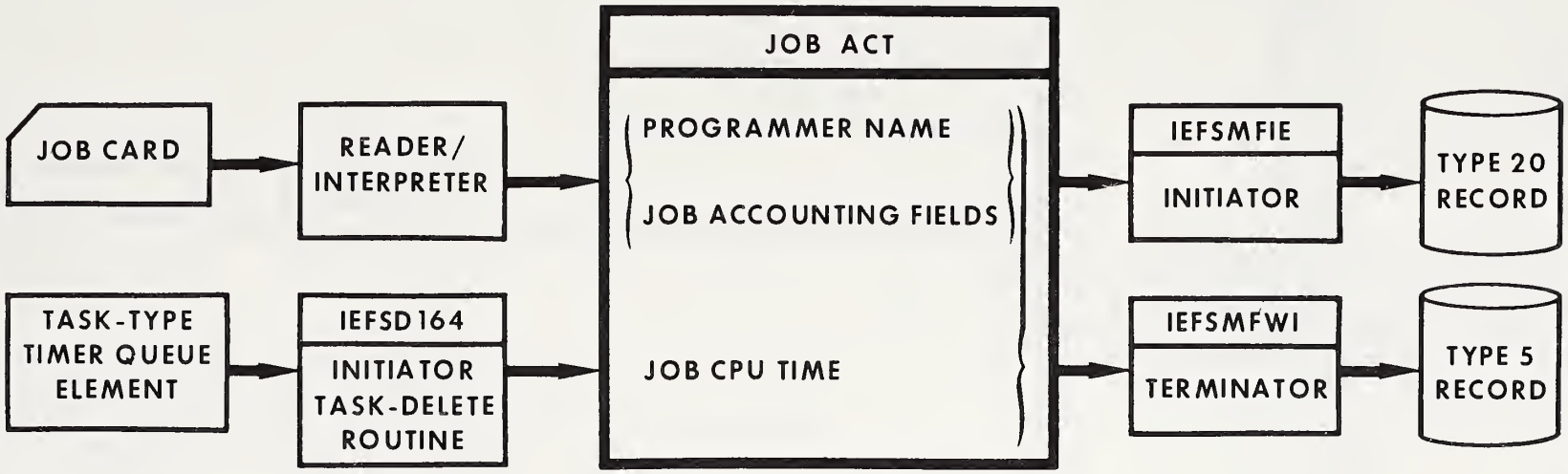

Figure 1. Job Accounting Control Table.

2. Step Accounting Control Table

The Step ACT is created, partially filled in, and written out to SYS1.SYSJOBQE by the EXEC statement processing routine of the Reader/Interpreter. It is read back in by the Initiator's Task Delete routine at the end of each step, updated, and written back to the job queue.

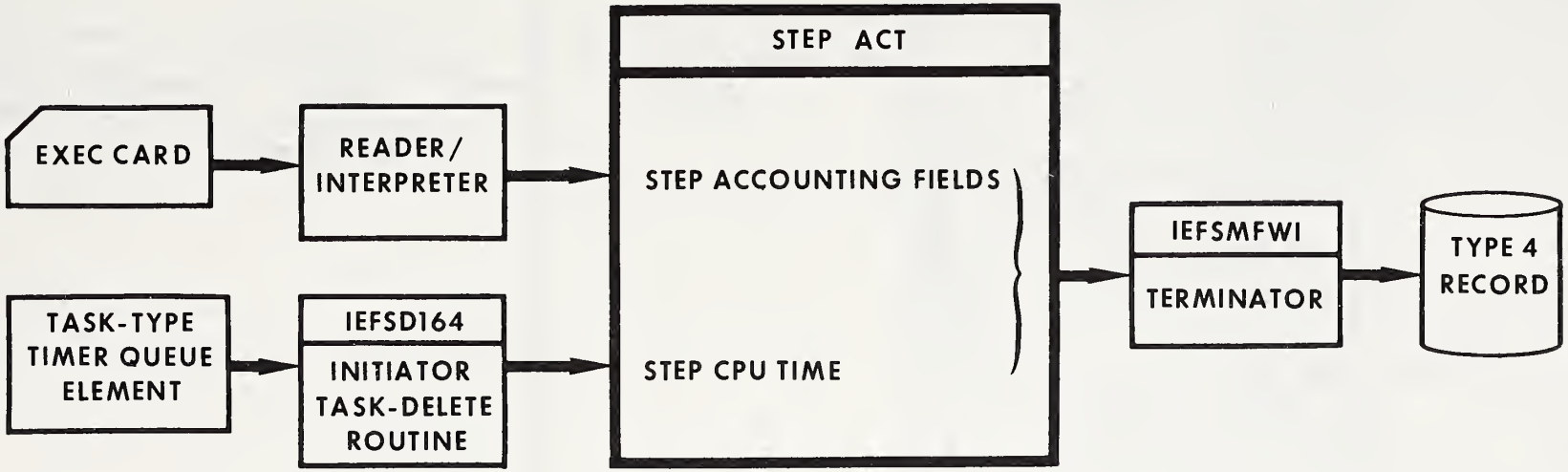

Figure 2. Step Accounting Control Table.

3. System Management Control Area

The SMCA is created in the System Queue Area of main memory by the SMF Initialization Routine of the Master Scheduler, and remains there until the system is reloaded. It is specific to SMF and does not exist in systems without SMF. Its address can be found in location CVTSMCA of the Communication Vector Table; this pointer is zero in systems without SMF.

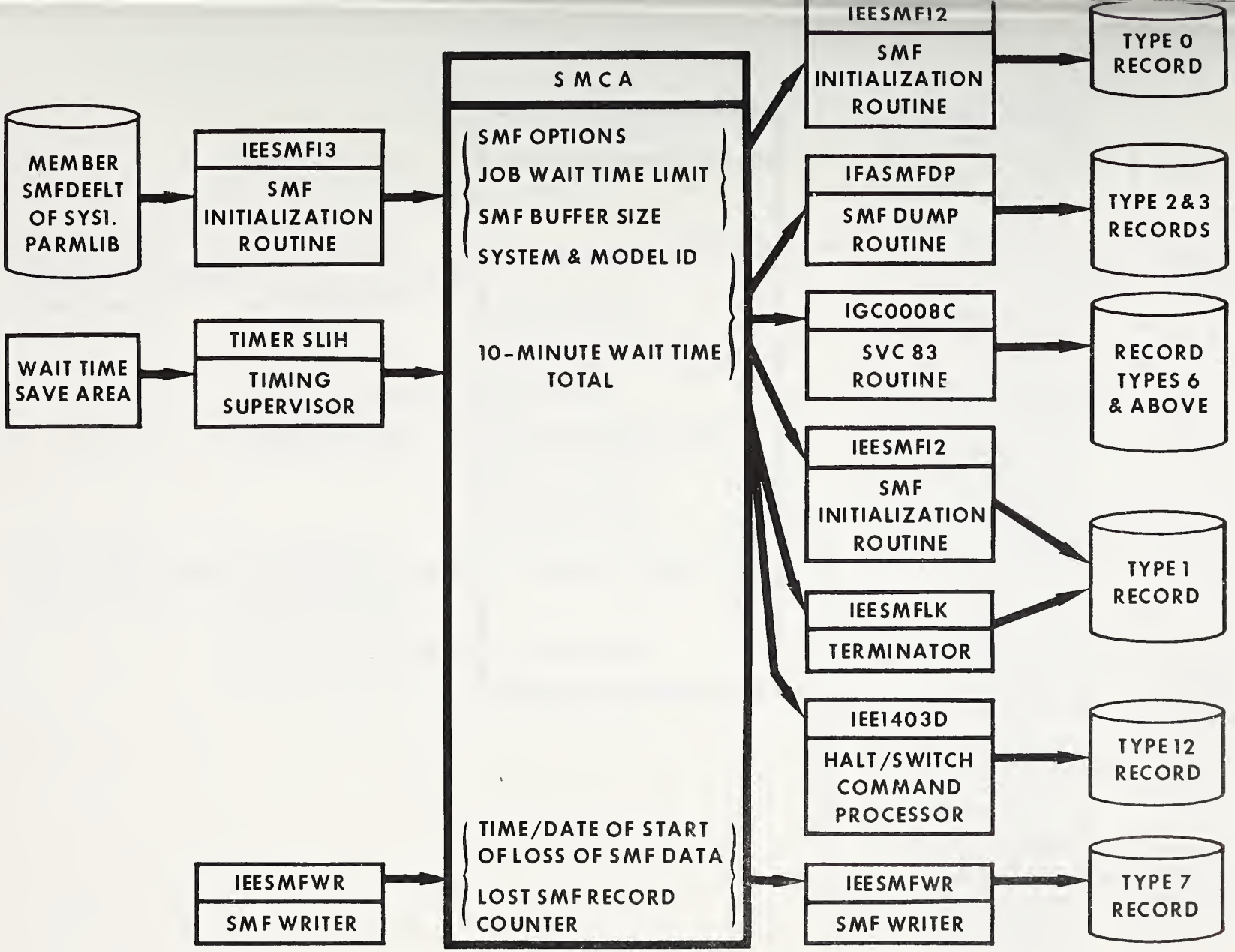

Figure 3. System Management Control Area.

4. Job Management Record

The JMR is created, partially filled in, and written to the OS work queue data set by the JOB statement processing routine of the Reader/Interpreter. In systems with SMF, it is read back in and completed by the User Exit Initialization Routine of the Initiator at the start of the first job step; from then on, it is maintained in main memory and updated at the start of each subsequent step.

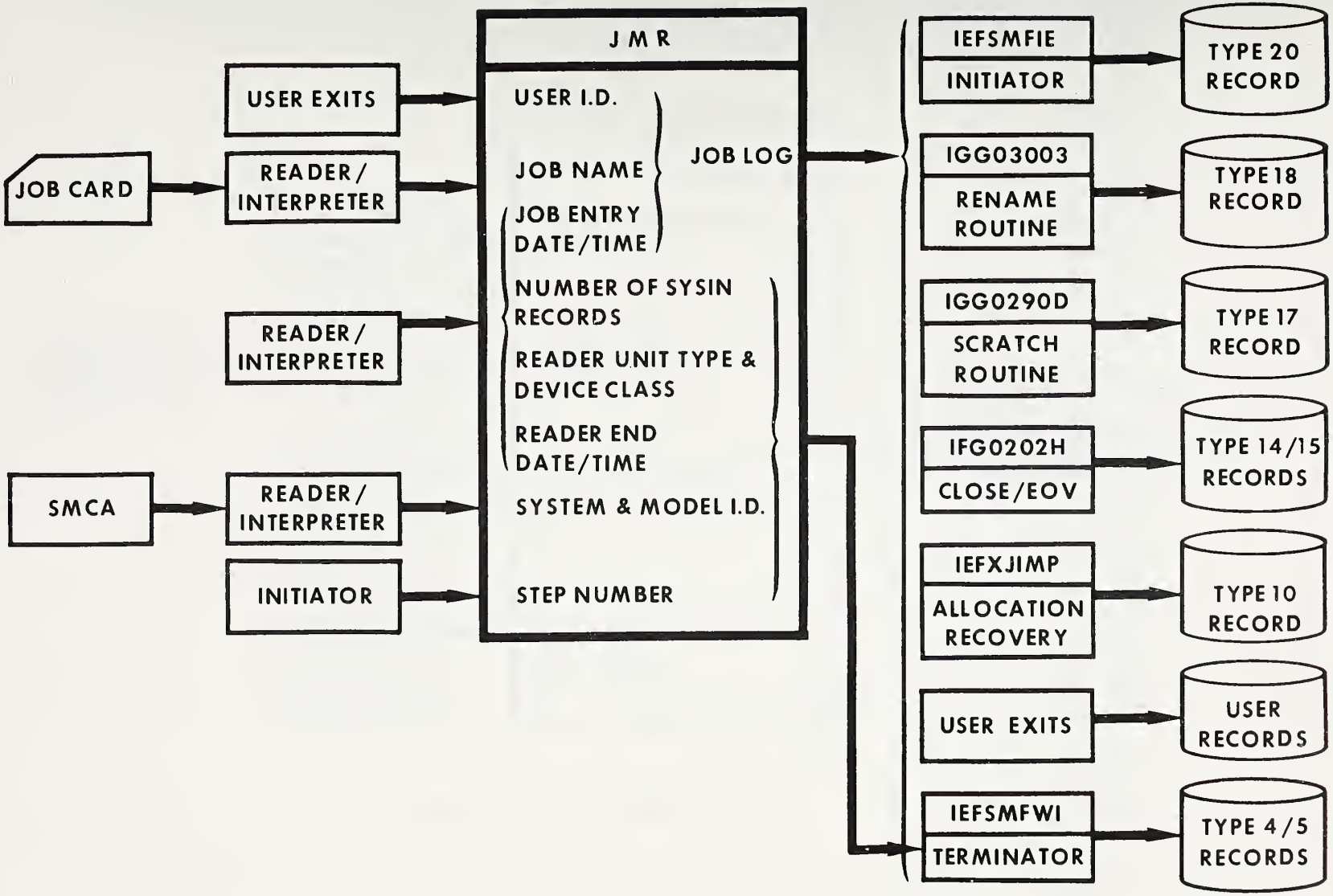

Figure 4. Job Management Record.

5. Timing Control Table

The TCT is created in main memory by the Initiator's User Exit Initialization Routine at the start of the first step of each job, and updated by that routine and by the TCTIOT Construction Routine at the start of each subsequent step. It is specific to systems with SMF.

The TCT is pointed to by a field in the Task Control Block (TCB) for the job step task. Any subtasks of the job step task contain pointers to the same TCT in their TCB's; thus all main storage used within the region allocated to the job step is accounted for in the same place.

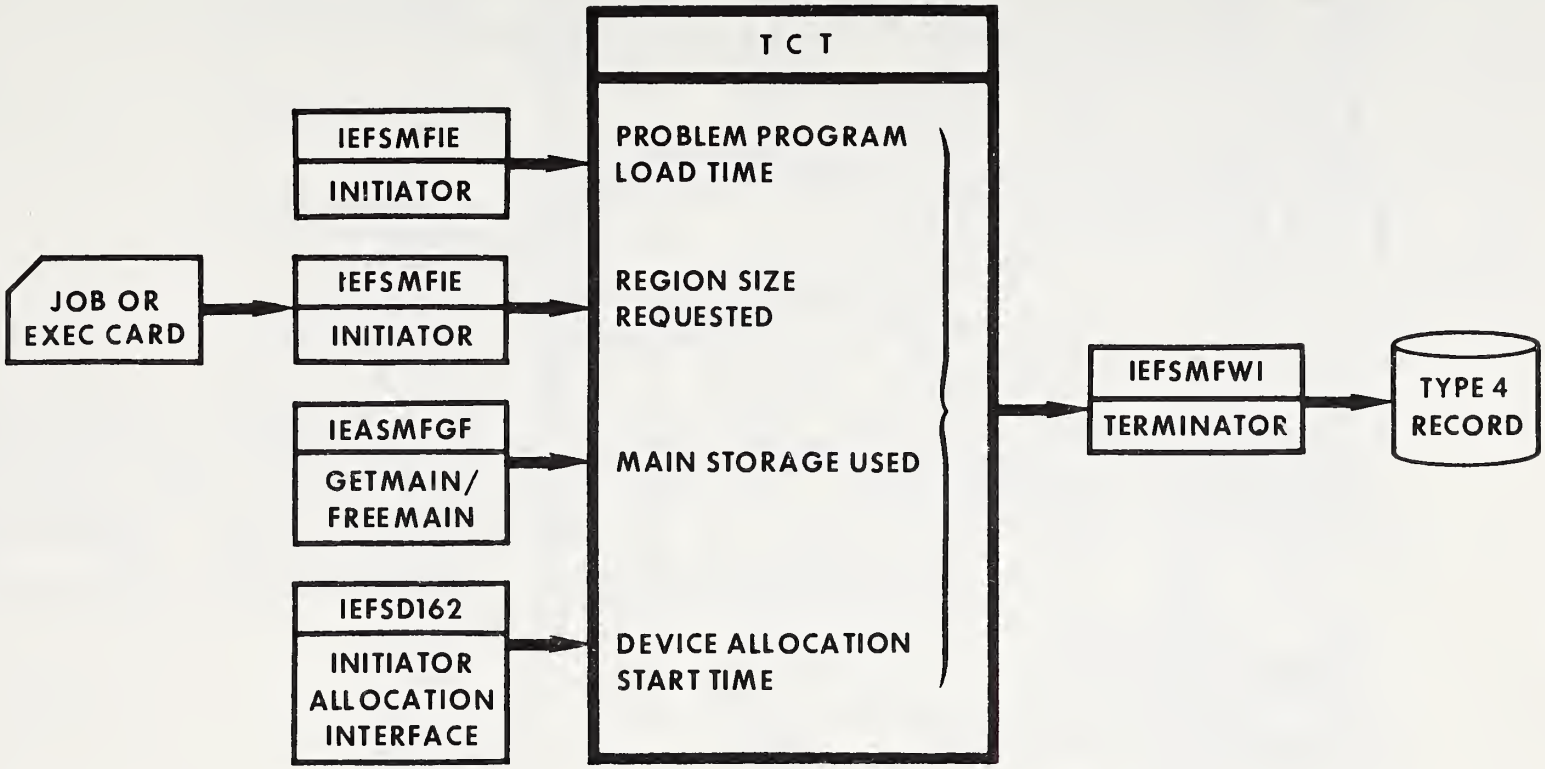

Figure 5. Timing Control Table.

6. Timing Control Task Input/Output Table

The TCTIOT is created in main memory by the TCTIOT Construction Routine of the Initiator at the beginning of each step, following device allocation and just before the problem program is loaded. Like the TCT, it is SMF-specific; in fact, the routines that use it pick up its address from the TCT. Its principal purpose is to maintain EXCP counts by data set and device, but its structure introduces inefficiencies into the SMF EXCP-counting function.

Each DD statement in the JCL for the job step gives rise to a DD entry in the TCTIOT as well as an entry in the Task Input/Output Table (TIOT). The Data Control Block (DCB) for the data set contains a halfword offset giving the location of the corresponding TIOT entry relative to the beginning of the TIOT. Near the start of the TCTIOT is an associative lookup section consisting of pairs of halfword offsets, the first of each pair being the same TIOT offset found in the DCB, and the second being the offset to the corresponding DD entry within the TCTIOT itself. Each DD entry contains as many eight-byte subentries as there are devices allocated to the data set, plus one more eight-byte field controlling output limits for Sysout data sets. This output limit field is present even for non-Sysout data sets, but contains a limit of zero in that case.

It should be noted that EXCP counts are maintained in the TCTIOT on a device-within-data-set basis, and no overall EXCP count for the entire data set is present. Thus testing whether the Sysout limit has been exceeded requires summing the EXCP counts for all devices allocated to the data set, every time an EXCP is issued. It should also be noted that the TCTIOT is designed to be searched associatively on the basis of the TIOT offset in the DCB to locate the DD entry; the length of this search will depend on the number of DD statements preceding the one for the data set currently requesting I/O.

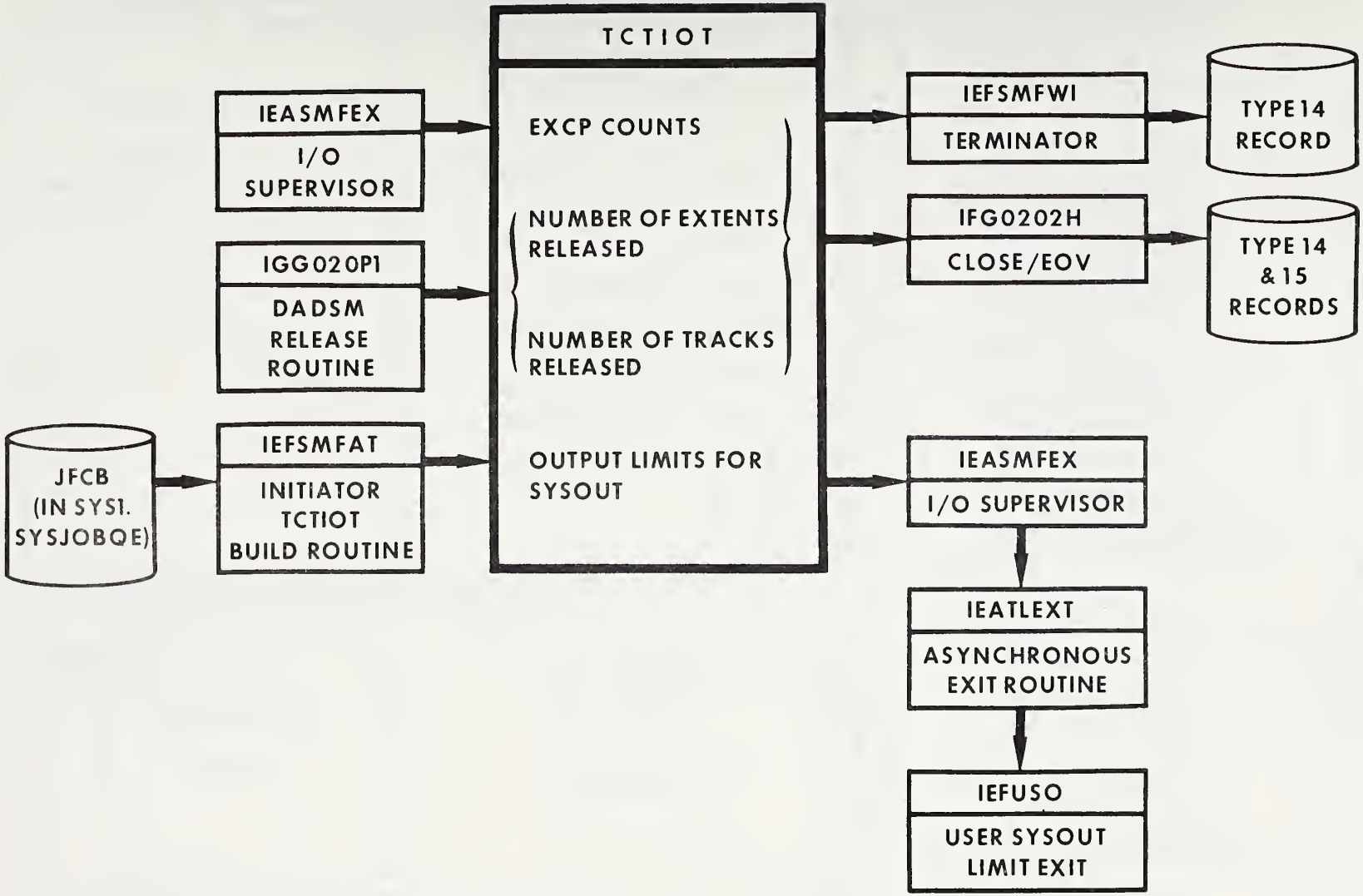

Figure 6. Timing Control Task Input/Output Table.

7. Table Interconnections During Job Step Execution

The accompanying diagram shows the interconnections between the TCT, the TCTIOT, and related control blocks and routines during execution of a problem program. Since the user time limit and Sysout limit exit routines run under the TCB for the current task within the job step, the pointer relationships illustrated hold true during execution of those routines also. By exploiting these relationships, a problem program (or one of the two exit routines mentioned) can gain access to the data collected by SMF at any point in its processing. The TCT and TCTIOT contain the region high and low water marks and the EXCP counts, respectively.

It should be noted that the accompanying diagram is not valid for other SMF exits, since they run as subroutines of system tasks (readers or initiators).

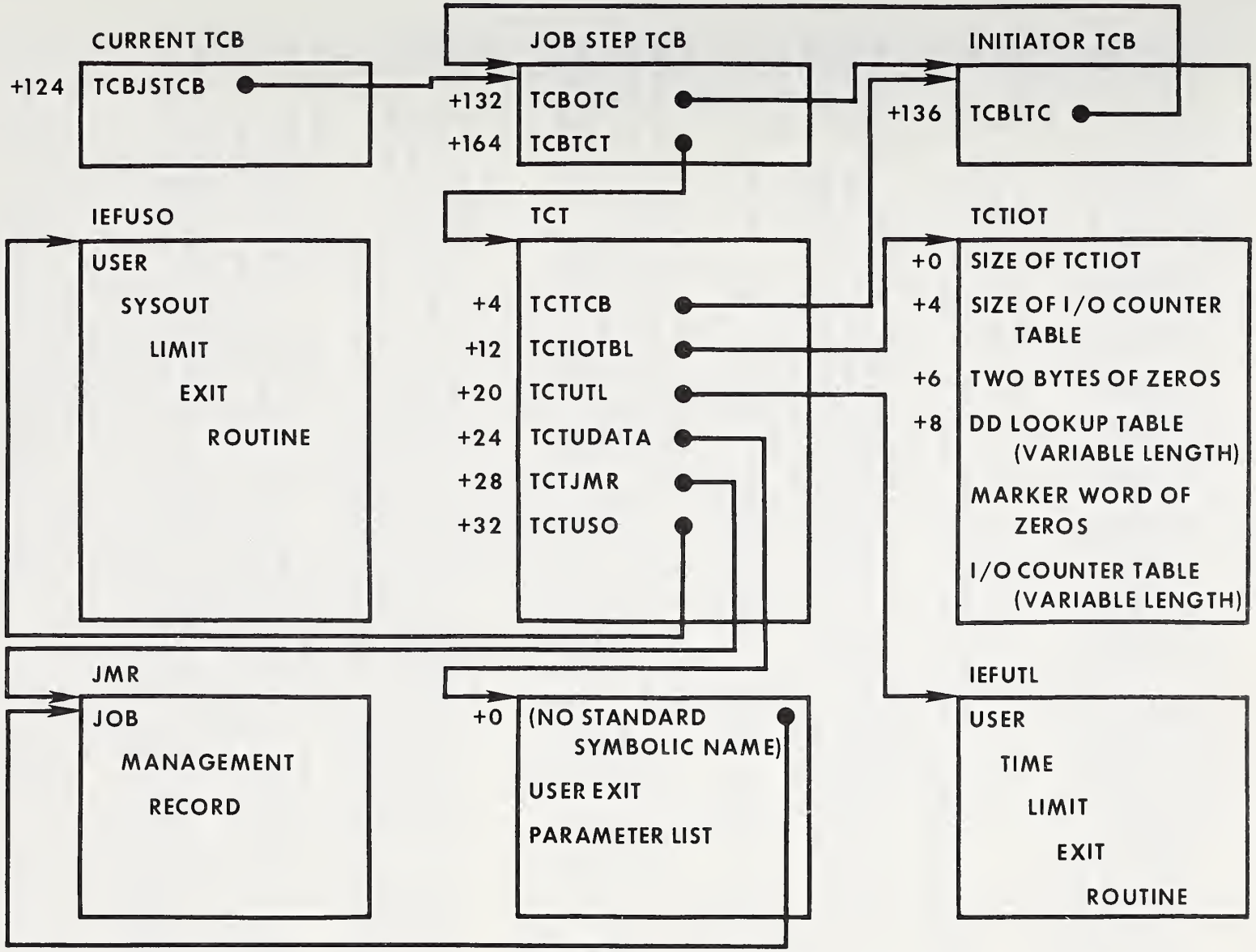

Figure 7. Table Interconnections During Job Step Execution.

8. Other SMF data sources

The major repositories of SMF data are the tables used by the Job Scheduler and the Master Scheduler; these are documented in detail in the Job Management Program Logic Manual (<5>). They include the Linkage Control Table (LCT), Job Control Table (JCT), Step Control Table (SCT), and the Master Scheduler resident data area.

The LCT, built in core by the initiator initialization routine, serves as a communications area among components of the initiator. From it, SMF extracts the job and step completion codes and the job class.

The JCT contains parameters from the JOB statement and other job status information. It is created and written to the system work queue by the JOB statement processing routine of the Reader/Interpreter, and read back into main storage by the initiator's Job Selection and Job Delete routines. From it, SMF extracts the step initiation date and time, job initiation date and time, job priority, the Sysout classes and message class indicator, and the job termination indicator byte.

The SCT (one per job step) is created and written to the OS work queue data set by the EXEC statement processor of the Reader/Interpreter. The initiator's Job Selection routine reads the first SCT back into main memory before the start of the first step. From the SCT, SMF extracts the step name, the program name, and the number of card images in "DD DATA" or "DD *" data sets.

The Master Scheduler resident data area is located in the nucleus, and contains control information common to all initiators and command processors. It is pointed to by the CVTMSER and CVTMSLT fields of the Communication Vector Table. It is of interest as the ultimate source of the storage protection key extracted by SMF from the initiator's TCB.

One further source of SMF data is the Volume Statistics Table, located in the nucleus, which holds error statistics and other information pertaining to magnetic tape volumes. It contains one entry for each 2400 series magnetic tape device on the system. The error statistics in this table are maintained by the I/O Supervisor's error recovery routines for magnetic tape, and are formatted as part of the Type 21 record by the SVC 91 (VOLSTAT) routine.

V. SMF and Performance Measurement

A. General

The resources of interest to the management of an installation running OS/360 are quite diverse. In addition to the commonly recognized ones such as CPU time, main storage, auxiliary storage, print lines, and I/O devices, there are other resources whose usage has a strong influence on performance or the turnaround of a particular job through a particular system at a particular time of day. These include operator interventions, channel loading, and several items that can be lumped together under the general heading of software resources.

A thorough approach to resource measurement and management must include identification of all resources in the system and the collection of data on their use. The subsequent paragraphs of this section define those resources that should be measured and evaluated to determine system performance and resource utilization. They also indicate the specific resources measured by SMF, the measures used, SMF sources of the usage measures, and possible views of the usage.

In general, all resource measurements can be categorized as applying to either the quantity of usage or the productivity of usage. The first category refers to the amount of the resource made unavailable to other contenders for that resource, and is usually expressed as a numerical quantity, possibly related to a time base. Thus we say a job used 30 seconds of CPU time or that a CPU is in use an average of 48 seconds per minute or 80% of capacity. The objectives of quantitative measurement of resources are to allocate costs of resources to jobs, determine resources that are under- or over-used, and to tune a system by balancing loads among various resources.

While quantitative measures define how much a resource is used, they do not say anything about how well a resource is used. For example, if a program being tested goes into a one-instruction loop and runs for an hour of CPU time, from the standpoint of quantitative measures the CPU is 100% in use, but from the standpoint of productive work, usage is 0% after the loop begins. The objective of productivity measures is to produce the most work from the available resources. SMF provides mainly quantitative measures of resources, but in the discussion that follows, both types will be covered where applicable.

Another goal of the subsequent sections is to describe, for each resource, how its use can be charged back to individual jobs or job steps. Such allocation of system usage to jobs is, in fact, the primary objective of the design of SMF; consequently, many of its usage measures sacrifice precision in order to provide better repeatability from run to run of the same job. Some resources are ignored altogether since they cannot be directly linked to a specific job. Part of the following discussion is based on material previously published in (<27>).

Job step and data set records carry a job log number, composed of the job name and reader time and date stamp, that can be used to associate all the records with a specific job. Each record bears a time stamp that makes it possible to organize the records into the sequence in which they were originally written. This last procedure is necessary in order to associate the data set records with their respective steps, since the data set, allocation recovery, scratch, and rename records do not contain step number or name. The normal order of records for a job is:

* job initiation record;

* allocation recovery records - step 1 (from initiator);

* rename records - step 1 (from utilities);

* scratch records - step 1 (from utilities);

* data set records - step 1;

* step termination record - step 1 (from terminator);

* allocation recovery, scratch/rename, and data set records - step 2;

* step termination record - step 2;

.
.
.

* allocation recovery, scratch/rename, and data set records - last step;

* step termination record - last step;

* job termination record;

* Sysout record;

.
.
.

* Sysout record.

Although the record time stamp provides a convenient means of sequencing records for a job, it is prone to serious difficulty if the system time is changed during execution of a job. (In fact, the SMF records can indicate that a job finished before it began if the time is reset while a job is executing.) An improvement on this method can be had by using the step initiation exit (IEFUSI) to set the step number in the user-i.d. field which is part of all job and data set records.

Usage of some resources cannot be directly allocated to jobs by means of SMF data; but for many resources, an approximation of the relative load placed upon the resource can be determined from the SMF data. The following discussion draws on a variety

of algorithms presented in the literature or in common practice to assign resource usage to jobs. Where possible, the assignment of resources is at the step level, where the programmer has the greatest control of resource allocation.

B. Human resources

Although the operator may not be the most expensive item in the machine room, his time is often a scarce resource. In OS/360, it is easy to multiply the number of operator interventions significantly through carelessness or ignorance in setting up JCL. Such commonplace utility operations as copying a series of data sets from disk to tape or scratching a few dozen unwanted members from an expiry-date-protected library contain the potential for enormous waste of the operator's time by requiring repeated remounts of the same tape or replying to multitudinous console messages. Being human, an operator subjected to such harassment is liable to suffer a loss of effectiveness in performing his regular duties. It is desirable both to account for the operator interventions necessitated by each job, and to measure operator-related delays so that installation management can recognize when operator time is the limiting factor in system throughput.

The direct measures of operator actions provided by SMF are:

* IPL's--from the IPL record can be obtained the frequency of IPL's and the duration between IPL's. While this will usually show the frequency of system problems rather than operator performance, many IPL's will impact operator performance.

* Allocation Recoveries--from the Allocation Recovery record can be obtained the frequency of allocation recoveries and the type of device for which they occurred. The frequency of allocation recoveries gives a measure of the demand upon the system operator for this type of decision making. An excessively high number of allocation recoveries may indicate either poor system planning or poor system operation.

* Vary Online/Offline--the Vary records provide another indication of decision-making on the part of the operator, as well as a measure of device availability. The Vary records, in conjunction with the IPL record, provide measures of device up-time, and frequency of Vary commands initiated by operators. Since Vary Online commands may be issued to simulate the device-ready interrupt for device mounting, those Vary Online records that do not reflect a change in device status must be discarded. The Vary records, combined with the I/O Configuration record, can provide the frequency of changes in device status.

Among the measures of operator actions not provided by SMF are:

* Volume Mounts--although SMF does not provide a record or a count of volume mounts, an approximation of the number of mounts can be obtained by careful examination of the CLOSE/EOV records. Such considerations as the presence or absence of Automatic Volume Recognition (AVR) in the system, resident and reserved volumes, presence of the RETAIN or PASS parameter, multi-volume data sets, temporary data set allocation, and device class will affect the number of volume mounts. SMF does not record the time taken to perform a volume mount.

* Operator Action Delay--SMF under MVT provides no measure of time spent awaiting operator action. There is a major improvement to SMF in the VS versions of OS where the device allocation end time has been added to the step termination records. This time does not give a precise measure of operator action times, since device allocation may be held up for a wide variety of causes. Neither version of OS records time spent by the operator responding to program or system messages requiring a console reply, nor the number of such actions.

Accounting for operator activity on the basis of SMF data is difficult enough to discourage any attempt to distribute the costs of such activity over individual jobs or job steps. However, by analyzing the data set records, a count of external actions can be developed that reflects the relative load a job places on the operator, rather than specific mount commands, which may be affected by pre-mounting, device conflicts with other jobs, and changes in configurations or operator practices. The next two paragraphs review the reasoning behind tape and disk mount charge assignment.

1. Tape mounts

Generally, every tape EOV implies an operator action to mount the tape at the start of the step and demount it when it is no longer needed, unless the tape is used in multiple steps, RETAIN or PASS was specified in an earlier step, and the volume sequence count is less than the number of devices allocated. A mount should be charged for each tape device allocated to steps that abend, since there may be no Close/EOV records. The Close/EOV records for the same data set/DD statement can be associated by using the TIOEJFCB field of the data set records. The presence of the PASS or RETAIN parameter on the DD statement can be inferred from the SMFTIOE field in the Close/EOV record.

2. Disk mounts

In most systems, very few allocations to DASD involve operator interventions. Two approaches can provide measures of operator actions, depending upon installation practice:

1) If specific devices are reserved for removable packs, then the first allocation to one of those devices in a job and any change in volume serial for one of those devices can be charged a mount.

2) If specific volumes are normally resident, then the first request for a non-resident volume causes a mount charge to the job.

While both of these methods generally accumulate more mount counts than actually occur, they have the advantage of fairly high repeatability, the second method more so than the first.

C. Hardware Resources

1. CPU time

This is the time used fetching and executing instructions; in those 360's with integrated channels, it may also include channel command execution and main storage access time. In any 360, the CPU may have to wait for access to main storage which is being used by a channel; such delays are considered "CPU time" since they occur during instruction fetch or execution cycles.

On a system-wide basis, i.e., total time spent executing instructions, SMF does not measure CPU time. It does collect the inverse of CPU time--wait time. Subject to certain imprecisions and irregularities in the collection and reallocation of the wait time values, wait time can be used to determine the total load on the CPU. In order for this value to be useful for tuning of a system, it would have to be further broken down by protect key and system state--this information is not presently available to any system function. Since the data collection interval is so long (ten minutes minimum) and may be broken if there is a system failure, the data on CPU usage must be accompanied by an estimate of its reliability.

On a job step basis CPU time is collected from the time a program begins execution until it terminates. Initiator/Terminator CPU time is not collected. This CPU time figure, however, may contain a substantial amount of time that is not actually related to the execution of the program, primarily I/O interrupt overhead for other tasks including time spent in appendages. There are many tasks for which no CPU time is recorded--system tasks and never-ending job steps such as HASP. Such tasks terminate only when the system crashes,so no SMF records are ever written for them.

In MVT systems, from 85% to 90% of the CPU time used by problem programs is reported in the step termination records. Since the CPU time that is not separately accumulated and not lost (e.g., I/O interrupt servicing time) is distributed over the tasks in execution in proportion to how much CPU time was used by each one, the reported CPU time in the step termination records is a relative measure of the CPU time used by a step. If the job

mix varies widely from run to run, there may be variations as large as 36% in the CPU time allocated to a job step for the execution of the same instructions.

2. Main storage

Main storage space, or "core" as it is usually called, is both a fixed and a dynamic resource. Allocation of main storage is controlled by the Main Storage Supervisor (GETMAIN/FREEMAIN) routines.

Main storage has three primary characteristics: speed, address, and protect key. The operating system provides for control of two speeds of core: high-speed or processor core, and slower Large Core Storage or LCS; the control of two types of main storage is an operating system feature called hierarchy support. Address is a characteristic of main storage that is significant for performance analysis, since the requirement for contiguous real storage addresses is the major difference between MVT and later versions of the operating system. This requirement for contiguous address intervals or "regions" is of concern when evaluating core allocation and usage. Storage protect keys are the hardware feature used by MVT to protect one task from another. The hardware restriction of 16 possible keys limits this resource and consequently the number of protected tasks in the system. Since the 360 architecture provides only one protect key for every 2048 bytes of core, the 2K block becomes the smallest unit that the Main Storage Supervisor can use in allocating regions. In fact, the structure and schema of the Main Storage Supervisor are based on 2K units.

The method of allocating regions is also the major difference between MVT and MFT. In MFT, partitions are allocated by a routine (GETPART) at initialization time and whenever a DEFINE command is processed. In MVT, the GETPART routine is invoked dynamically by the initiators: consequently, the problem program area of main storage can be used more flexibly.

Usage of each type of main storage can be measured in terms of quantity (bytes or K), frequency of reference, duration of occupancy, and whether the usage is fixed or dynamic.

Fixed main storage consists of (1) the Nucleus which contains resident routines, tables, and control areas; and (2) the Link Pack Area which contains optionally resident routines and may be distributed across processor storage and in LCS. The only measure of fixed main storage provided by SMF is the maximum hardware size at IPL time in the Type 0 record. The sizes of the nucleus and link pack area are not available from SMF.

Dynamic main storage consists of a System Queue Area which is used for control blocks for system functions and job steps and the contents of which may be modified by the Supervisor only, and a Dynamic Area which is used for region space for executing job steps. SMF provides no measure of the amount of System Queue

Area used by a particular step. Various algorithms have been developed to infer the nature of usage of the dynamic main storage area from an analysis of the Type 4 records. However, since no Type 4 records are written for system tasks such as mount and start commands and no measures of any kind are provided for dynamic System Queue Area, all such algorithms have a high degree of error.

By analyzing the Type 4 records (region size, protect key, execution start and stop times), the amount of region space in use for job processing can be approximated. This measure provides an indication of the amount of dynamic area in use for productive work. The difficulty with this measure is that it is unclear whether the remaining space was free, fragmented, or in use by tasks not accounted for.

On a step-by-step basis, the usage of the region requested for a step can be analyzed by comparing the values of core requested versus core used. While the core used figure may be imprecise for many jobs, those jobs that request significantly more region size than they need can be isolated using these values. Any differences greater than 20% between core used and core requested generally indicate wasted region space. Because programs such as Sort/Merge and certain compilers adjust their core requirements to use whatever space is available to them, job steps executing them will always show 100% region utilization. Region sizes for such steps can be chosen to improve storage utilization of non-self-adjusting steps in the same job. Since MVT tends to perform better if region sizes on the successive steps of a job decrease rather than an increase, the region-requested value can be analyzed to locate jobs that may be contributing to poor core utilization.

The IBM 360 and 370 hardware does not provide any measure of frequency-of-reference usage of core and hence no working set or productivity measures of core on either a system or job step basis are available from SMF.

The core-allocated values for each main storage hierarchy are present in the step termination record. These values are only the amounts of core and LCS in the region and do not include System Queue Area. The measure of the main storage resource must include a time-in-use quantity to be meaningful. Two approaches are possible using the SMF data, as discussed in the next two subsections.

a. Job step duration

Using the start time from the step or job initiation record and the termination time from the job termination record, the job duration can be computed. This value accurately reflects the time the job resided in core, but since, in a multi-programming environment, there is high variability in job durations due to the interference of other jobs, this value does not accurately reflect the demand placed on this resource by the job.

b. Effective duration

Based on the idea that, when a task is ready to compute or do I/O but cannot because another task has control of the CPU or channels, that task's core is not effectively "in use", a number of algorithms have been developed to compute an effective duration, and hence a more repeatable core usage value. The idea is to add to the measured CPU time an "effective I/O time" based on I/O usage. How to arrive at an effective I/O time is explained below under "I/O devices".

Whether actual or effective duration is used to measure occupancy time, the main storage usage charged to a job step is normally taken to be the product of core allocated and residency time, expressed in such units as kilobyte-minutes.

3. Direct-access storage

Direct-access storage is storage on any of the variety of magnetic disk or drum devices available to IBM 360-370's. Devices can be classified by transfer rate, capacity, track size, and performance characteristics. Devices with similar characteristics have the same "device type" assigned to them for identification by the Operating System. In any configuration, there is a finite quantity of DASD storage. Some jobs may have to await the freeing of DASD space before they can execute if the resource is saturated; many will terminate abnormally if the resource is exhausted.

DASD storage may be subdivided into spool space, work space, online storage, and offline storage; it may be further classified by device type and device performance. Space may also be valued more highly if it is on a dedicated or resident volume, becoming in effect a part of the operating system. SMF provides very few measures of DASD space usage, and some of the measures are better provided by other OS functions.

a. Space available

While the space available record reports the available space on a DASD volume and various volume characteristics, its usefulness for space management or analysis is limited. Multiple volumes with the same serial number or the actions of utilities can radically change the values reported by SMF. The Type 19 record is only written at IPL, HALT, SWITCH, or volume removal time. Thus, in a stable OS system, the Type 19 information may be very stale. An IEHLIST of the volume provides the same information as the Type 19 record. Additionally, the IEHLIST utility can show data set allocation by data set name so that maintenance action can be taken.

There is no way permanent or semi-permanent DASD space can be allocated to a single job. This is more appropriately handled by a DASD space management system that can allocate track days to specific users.

b. Space used

The data set records (Types 14 and 15) provide a partial measure of the amount of temporary disk space used by each step. This includes time in use, tracks allocated and used, volume serial number, device type, and primary device address. This data does not include data sets allocated but not opened, data sets allocated in one step and passed around the step being measured, or work data sets allocated but not classified as temporary by the system. A more precise picture of work space can be obtained by combining the data set records with the scratch records. This information, combined and graphed over time, can provide a map of the usage of PUBLIC and STORAGE volumes for work data sets.

DASD work space for temporary data sets can be determined for a job by combining the data set records and the scratch records. By making the assumption that a data set is allocated in the first step that uses it, the high water mark for each step can be determined and then, using either effective duration or actual duration, the work space use can be expressed in track-seconds or track-minutes.

c. Productive use of DASD space

Productivity measures of DASD space are generally confined to analyzing the density of data within a data set. SMF provides two measures of density: the first is the tracks used versus allocated figure from the data set records--this can provide an analysis highlighting the worst cases. The second measure uses the device type, access method, and logical record length, also from the data set records, to determine the best block size and compare it with the actual block size--this can show where space is being wasted by poor choice of blocking factors.

4. Channels

When none of the tasks in a multiprogramming system is highly computational in nature, overall performance depends heavily on channel loading and interference among users of direct-access storage devices.

Channels are in use for a task from the time the I/O is started until the channel-end interrupt has been serviced and the I/O operation is posted complete. This may be a simple operation like a card read that performs a single buffer transfer; a more complex operation like a disk write involving command chaining and branching in the channel program; or a sophisticated operation like a teleprocessing poll that relies on intricate channel programs and appendage routines. While a channel (or multiplexor sub-channel) is in use, a number of other tasks may be waiting for the resource. The operating system provides channel queues to pend these requests until the channel is available.

Channel utilization can be measured from a variety of viewpoints: channel busy time, depth of channel queues, channel data transfer, and channel storage access. Separate measures are required for each channel, channel type, or sub-channel in the configuration.

SMF does not measure channel usage. A large number of channel transfers go unrecorded, e.g., system tasks, errors, and actions by appendages; and the units counted (EXCP's) do not reflect the usage of the channel for an I/O operation or even the specific channel involved (in the case of a multi-channel device). The analysis of EXCP counts from Type 4 or Type 14/15 records can provide an overall measure of job-related I/O activity in the system, but the usefulness of such a measure for analyzing system performance is limited. Only in systems with single channel paths, a stable configuration and mounting characteristics, as well as a similar workload from day to day, can the SMF EXCP counts by channel give some indications of channel balance, but no determination of total channel load can be made.

Although channel usage cannot be precisely determined from SMF data, it is possible, by assigning standard values to variable device characteristics such as tape start time, rotational delay and buffer transfer time, to approximate the relative load placed on the channels by particular jobs. In addition to these standard values that can be determined from device class alone, if the block size value from the data set records is used, the average data transfer time can be computed and added to device latency. Since the characteristics of channel usage vary significantly by access method, the data set organization and access type (sequential or direct, input or output) from the data set record can be used to estimate rotational delay and arm movement latency values. Since the EXCP access method does not provide block size values, transfer time averages have to be used for this access type. Channel time for HASP pseudo-devices can be assigned as a fraction of the implied real disk I/O.

Channel utilization can be categorized by selector or block-multiplexor time and byte-multiplexor time based on the high-order four bits of the device address.

5. I/O devices

I/O devices fall into two categories, (1) serially usable, e.g., tapes and printers, and (2) sharable, e.g. disks and drums.

Serially usable devices are in use by a task or the system on an exclusive basis from the time they are allocated until the time they are deallocated. Such a device is completely unavailable to other users during the time it is allocated, even if there is no data transferred to or from it. For example, a tape drive may be allocated in a daily run to a file that is only

created once a month. Serially usable device utilization can be measured by the time devices are allocated to tasks. Some devices which are sharable may be used in a serial fashion; for instance, very large disk data sets may require wholly dedicated drives.

Sharable devices are only in use by tasks while they are actually performing I/O operations on those devices. A single I/O operation on a sharable device cannot be considered alone. Each such operation has an associated overhead that affects all other tasks using that device, since increased arm movement or rotational delay causes more I/O wait time and consequently greater core residency time. When considering the usage of sharable devices, it is best to subdivide the devices based on utilization. Hence, classes of such devices can be created for:

(1) system residence volumes;
(2) resident private data sets or library volumes;
(3) public work space.

In many cases, devices are dedicated to a specific function by the operations management group. Devices might be dedicated for security reasons, to provide fast job turnaround or for the ease of the operators. Such dedication must be considered when discussing resource usage, even though it may not be reflected in any quantitative measurement.

SMF provides a relatively good measure of the usage of serial devices. Since such devices are in use from the time they are allocated until they are deallocated, the step termination record can provide a measure of device usage. The device address combined with step start and end times gives the time each device was allocated. Comparing this time with the time online derived from an analysis of the I/O configuration, Vary Online, and Vary Offline records gives a measure of the usage of available time. A comparison of time online to time available gives a measure of usable time. When considering usage of serial devices, such devices as unit record devices allocated to HASP, and HASP pseudo-devices should be excluded from the analysis. For a serial device that is switchable from one CPU to another, the usage figure must include contributions from all CPU's having access to it.

An analysis of the productive use of serial devices can be made using the step termination record and the data set records. The step termination record gives the time the device was allocated while the data set records give the number of EXCP's and (for standard sequential data sets) the amount of data transferred. Such an analysis can lead to the reconfiguration of a step's devices for better resource utilization.

The analysis of the usage of sharable devices is more difficult than that for serial devices. The usage of a sharable device is primarily the activity level of the volume on that

device. That is, given a number of devices of the same type, the most heavily used device will be the one on which the most heavily used volume is presently located. Heavy concentration of device usage does not necessarily mean poor performance; the opposite may be the case. Since analysis of channel utilization has the objective of improving throughput by spreading the I/O load among the channels, device usage analysis can be confined to those elements concerned with devices alone. Such elements as arm movement and rotational delay affect device usage.

The data set records give the EXCP count for each volume serial and device type, as well as the access method, block size, and data set name. A period of usage is the step duration from the appropriate step termination record. This data can be used to identify the high-usage volumes, determine the relative usage levels of the devices, reveal possible conflicting usages of different data sets on the same volume, and locate high-volume data sets with poor block sizes for the device. Because of the imprecision of the data, care must be taken that sufficiently high levels are identified, and further research may be called for in order to improve performance. Many data sets that are used by the system in a shared fashion are not recorded by SMF so that even moderate usage levels on those devices may be cause for concern.

Since SMF does not record the length of time from I/O start to I/O completion, it is necessary to develop an effective I/O time. This time is channel time, computed as above, plus device-dependent waiting times such as tape stop time, disk head seeks and forms or card movement. Most costing algorithms view all I/O time as additive, but in the opinion of the authors, if teleprocessing devices are excluded, selector and block-multiplexor channel devices should be considered additive, but byte-multiplexor devices should be considered limiting. That is, the I/O time ought to be taken as the sum of the I/O time from selector and block-multiplexor devices or the longest time of byte-multiplexor devices, whichever is longer. This more accurately reflects the fact that a card-to-tape job usually runs at the speed of the far slower card reader, but a tape-to-tape job takes the full time to read and write both files.

Development of an effective I/O time ignores the benefits derived by some jobs due to the overlapping of I/O activity, but in heavily loaded systems, such overlap more often accrues to the system (by increasing the multiprogramming level) than to a specific task. On the other hand, such a treatment has a high degree of repeatability, and with proper calibration can be made very precise using sampling and monitoring. For purposes of accounting for usage of I/O devices by job steps, serial devices must be treated differently from sharable devices.

Since serial devices such as tapes, card readers, mountable DASD and the like are in exclusive use the whole time they are allocated to a step, the actual measure of usage is, like main storage, the duration of the step. As noted above, this value is

quite variable, so that a better measure is again effective duration.

For shared devices, the most appropriate usage measure is simply the I/O time on those devices. This takes into account access method, access type, device latency such as rotational delay and arm movement, and data transfer time.

Some installations have divided direct-access devices by address, defining some as shared (PUBLIC or STORAGE) and others as serial (PRIVATE). Others have assigned these characteristics on the basis of volume serial number and, in at least one installation, by the temporary flag in the data set records.

6. Media

This class comprises printed lines, punched cards, paper tape, microfilm, magnetic ink and all other non-reusable recording media. Each physical medium may be further classified by usage. For example, the user may group his printed output into different Sysout classes according to content, or the installation may want to assign higher costs for printing during prime shift, at high priority, or on special forms.

Other than cards punched and lines printed, SMF provides very little direct information about non-reusable media resources. Specific devices such as a MICR printer or paper tape punch could be isolated and media usage calculated from the EXCP counts. With punched cards, the usage measure is good as long as the Sysout writer is used, since counts in the Type 6 record include the form number and one output record is one card. For printed output, the measure is not quite so precise, since a line count rather than a page and line count is provided. But the counts for printing are provided by form number so that proper conversions to page counts could be applied to develop media usage. If output devices are directly allocated to steps rather than producing output through Sysout, the computation becomes more complex, since the EXCP counts must be accumulated and then form numbers externally assigned.

Since the Sysout records contain line or punched-card counts, it is possible to allocate media to specific jobs in terms of lines printed or cards punched. The form-number field in the Type 6 record allows media usage to be categorized by type of form or card stock. In non-HASP systems, Sysout usage can only be allocated on a job basis; it cannot be further broken down to job steps using only the SMF data.

7. Volumes

The term "volumes" is used to describe physical devices within or upon which data is recorded and which are viewed by OS as limited in capacity. Each volume (disk pack, drum, tape reel, etc.) is identified by a serial number that must be unique within

OS at any one point in time. Volumes are a finite resource for each device class, in that each volume contains only a limited amount of data. Even within device classes, there may be variations in capacity, e.g., 2400, 1200, and 600-foot tape reels.

Space within a disk volume is controlled by Direct Access Device Space Management, DADSM (see "Direct-access storage" above), but control of volumes themselves is mainly left to the system operator or the user through JCL. OS uses the volume serial number to ensure that volumes requiring serial devices (e.g., tapes) are allocated to jobs one at a time. The allocation of non-specific references to volumes is primarily controlled by assignment of a volume state (public, private, or storage) by the system operator (via the MOUNT command) or by the user (via the VOLUME=PRIVATE parameter on the DD statement). The availability of a volume is also controlled by the assignment of a mount attribute to the volume at the time the volume is presented to the system, either by the operator when it is mounted or by the operating system if the volume is already mounted at IPL time.

The management of volumes in OS has a major impact on such resources as initiators (device allocation), operators (volume mounting), DASD space, and others. Volume usage can be measured in terms of frequency of reference, amount of capacity in use, duration of usage, and amount of time spent waiting for volumes. By matching data set records with the corresponding step termination records, the duration of volume usage by each job step can be determined. Volume serial numbers, EXCP counts, and channel and unit addresses are available directly from the data set records. Thus the measures of volume usage practically available from SMF data are frequency of reference and duration of usage. These measures can be used to isolate tape volumes with small data sets more appropriately allocated to disk or stacked on tape, disk volumes seldom used but inappropriately made resident, or disk volumes heavily used but mountable. By combining volume use data with DASD volume space analysis data, reorganization of data sets to reduce volume mounting can be accomplished.

8. Shared DASD

Shared DASD support is a feature of MVT that provides for sharing of direct-access storage devices between two or more CPU's. Shared DASD does not refer to those devices that are switchable from one CPU to another but rather to concurrent usage by two CPU's. The hardware that supports this feature is supplemented by routines in operating system components to use the feature.

Shared DASD should be considered a separate resource, since the use of a shared device by jobs running in one CPU can affect the operation of the other CPU. In this case, then, the usage of a resource cannot be viewed from the standpoint of the single

system, but must be viewed by considering the composite system.

Shared DASD provides many benefits, since data that is used by jobs that must run on different CPU's can be shared and thereby reduce disk space requirements and volume mounting. Likewise, work space can be shared and provide higher utilization of disk space and decrease total space requirements.

Since DASD volumes that are shared between systems must be mounted resident or reserved on the sharable devices while they are in concurrent use, they provide a unique operational problem. Likewise, because of the necessity to combine usage from two or more systems and the possibility that sharable devices may not always be used in a shared manner, shared DASD has unique measurement requirements. Shared DASD must be measured in the same units as DASD space and sharable I/O devices.

Like ordinary sharable devices, shared DASD usage can be measured by the activity level of the device. If the volumes contain such system data sets as catalogs, program libraries, and SMF or error recording data sets, a significant number of EXCP's to the devices will go unrecorded. For other data sets, the EXCP counts, block size, data set name, and access method are available from the data set records, and the usage period is available from the corresponding step termination.

For shared devices, it is necessary to combine the SMF data from both systems and establish an equivalence of device addresses. If shared DASD space is used for work space, then it will be necessary to combine data set, step termination, and scratch records for both systems in order to measure work space usage on each system.

D. Software Resources

1. Device allocation

Device allocation and deallocation is strictly a sequential process. Although several initiators may be active in the system at a time, only one of these may currently be allocating or deallocating I/O devices. Any other initiator(s) needing to do so must wait until the active one is finished. Thus delays in device allocation for one job prevent other jobs (eventually, all jobs) in the system from being initiated or terminated.

The measures of device allocation as a resource are simply the time (CPU, channel, wait time) spent allocating and deallocating devices and the time spent waiting to allocate or deallocate devices. This resource is so critical to system performance that software products which replace the MVT device allocation algorithm with a better one are now being marketed, and IBM itself uses an improved algorithm in OS/VS.

The only measure of device allocation available from SMF is the duration in device allocation and region allocation provided by the difference between the step start time and the problem program load time on the step termination record. By plotting this time against the number of initiators in use (derived from job durations and protect keys), a performance analyst could spot device allocation interlocks caused by too many initiators or bad class asignments.

2. Data set names

Another software resource for which contention may develop is data set names. If two concurrently executing jobs wish to use the same data set, and at least one of them requests exclusive control through its JCL, then whichever job is initiated first will force the other to wait for the data set until the first job terminates. (The operator has the option of cancelling the second job.) Since high usage of a data set does not necessarily develop contention, measures of exclusive use, shared use, initiator time waiting for shared use, and initiator time waiting for exclusive use must be provided to analyze this resource. SMF provides no straightforward method of analyzing queues of data set names, although this bottleneck can be most difficult for operations to control.

3. Transient areas

A potential system bottleneck is I/O error recovery. All I/O errors cause some form of recovery to be attempted under the control of the System Error Task. This task is performed by several modules which are loaded into a single transient area in the Supervisor (low memory addresses). Particularly on teleprocessing systems with hundreds of terminals it is possible for bursts of nearly simultaneous errors on numerous devices to keep the System Error Task busy for seconds at a time; meanwhile, nothing else will be happening in the system, not even operator console communication, since the System Error Task has higher priority than the Communications Task.

Portions of the operating system that are infrequently used may be stored on disk in SVCLIB and retrieved into core and executed only when required. These routines execute from fixed 1024-byte transient areas in the nucleus. Optionally, some of the more heavily-used routines may be made resident in the Link Pack area as resident SVC's. Each transient area can be in use by only one transient module at a time and all tasks must share the available transient areas, although higher priority routines can steal transient areas. The SVC transient areas and routines are a high-demand resource that is further limited by access to the SVCLIB data set, arm movement, and channel access to the Sysres device (upon which SVCLIB must reside). In a heavily loaded system, one of the major tuning functions is the balancing of the number of transient areas, placement of data sets on the Sysres volume, and the control of the Link Pack Area--all affected by transient SVC usage. Measures of this resource can

be expressed in terms of frequency of reference to each transient routine, time in use of each transient area, and length of time tasks were enqueued on transient areas and routines. The measures of the usage of the I/O error transient area are: length of time transient area was in use, length of time each transient routine was in use, and length of time tasks were enqueued for the use of the error transients. This information would, if readily available, make it easy to decide which routines should be made resident. As SMF provides no data for the analysis and management of transient area usage, alternative measurement techniques (special-purpose software monitors or hardware monitors) must be used.

4. System catalog

Another important resource on which SMF provides no information is the utilization of the OS catalog. The structure of the OS catalog is such that the overhead of searching for a data set name in the catalog depends far more on the distribution of index levels and control volumes than it does on the total number of data sets catalogued on the system. The usage of the catalog can vary radically from job to job, and the number of accesses may vary significantly from data set to data set or even from day to day for the same data set. The measure of usage of the catalog should be separated for system catalog and user catalogs and for the number of entries searched at each level. An overall measure of catalog usage could be derived from initiator time spent waiting on catalog searches and Sysres usage for catalog access.

5. Work queue

The system work queue (SYS1.SYSJOBQE and, in HASP systems, the HASP input queue) is a twofold resource. It is a finite repository for jobs awaiting execution and a serially reusable source of control blocks for steps in allocation, execution, or deallocation. Since a system can remain loaded only if executable jobs are enqueued, it is important that the input queue be large enough to contain a sufficient number of jobs awaiting execution. Inadequate queue size is indicated by an operator message and requires no special analysis. Overallocation of the input queue and consequent underutilization of disk space could be analyzed by a measure of the high water mark of queue records in use.

The serial use of the system work queue data set may result in wait time for many tasks in the system. The measures of this aspect of the resource are time spent waiting for Job Queue access and accesses by component.

Generally, SMF provides very little information for measurement or analysis of the work queue. It is possible, by using the time stamps on the job termination and step termination records to determine how long jobs spend in the queue. The average number of steps and DD statements per job can be

determined from the step termination records. These figures are very poor measures of queue utilization, but may be useful for estimating job queue parameters.

6. Initiators

MVT provides for a maximum of fifteen initiators--one for each non-zero protect key. One initiator is dedicated to a job for the life of that job; hence, a maximum of fifteen jobs can run in the system at any one time. Each initiator proceeds through a defined set of functions allocating and deallocating resources to each step of the job. Many of these functions can only be done by one initiator at a time, so each one enqueues on a class of resources prior to allocating from that class. Other functions require demand allocation of resources on a first-come, first-served basis.

Since time spent with the initiator is nonproductive time for job processing, the initiators have higher priority than any processing task to speed their functions. Even with this priority, initiators frequently spend relatively long periods waiting for such resources as main core, system queue, catalog, device allocation, operator action, and volume verification. To the same degree that initiators are interlocked, system throughput is affected. In many cases, the nature of available resources is such that fewer rather than more initiators will improve system throughput and overall performance. To determine the proper system balance, measures of initiator usage must be combined for analysis. Measures of initiator usage include: length of time each initiator has a job selected, length of time spent waiting for each resource, number of initiators in use and the status of each one, job classes assigned to each initiator, and the amounts of CPU time, core, and System Queue Area and the number of EXCP's used by each initiator.

SMF provides no measure of initiator CPU time, EXCP's, or System Queue Area usage. However, by using the job select time, step select time, step end time, job end time, protect key, and job class from the step records and job records, the number of initiators in use, job classes in use, and duration of time each initiator has a job selected can be determined.

7. Spool/HASP

The spooling of output destined for recording on external media is indispensable for flexible job scheduling. However, relieving the pressure on hardware resources (printers, punches, etc.) creates a software resource whose allocation affects throughput and job turnaround; consequently, measures of spool utilization are of interest to installation management.

The spool resource is composed of two elements, namely, space to hold spooled output (usually on DASD) and space to hold control information identifying spooled data sets. In OS/MVT without HASP, system output (Sysout) data sets may be spooled

onto DASD or tape; since spooling to tape is extremely rare, spool space is effectively a subset of direct-access space. Likewise, control information for Sysout data sets is contained in special records (Data Set Blocks or DSB's) in the OS Job Queue and thus can be considered a subset of the work queue resource. Because of its special function in the system and special treatment in the job control language, spool space is nevertheless worth isolating as a distinct resource. It can be controlled and accounted for separately from DASD in general.

In systems with HASP, the existence of a separate spool resource is much more obvious, since HASP's spool space is pre-allocated and pre-formatted, usually on one or more dedicated volumes, and is thus physically as well as logically distinct from other DASD space. This is, in fact, one of HASP's advantages, since the percentage utilization of spool space can always be calculated and displayed on the console in response to an operator inquiry. HASP is similar to OS in that jobs awaiting print (or punch) are controlled by records in the Job Queue (HASP's queue, not the OS queue), making it possible for heavy use of this element of the spool resource to prevent acceptance of further jobs into the system. Job entry may have to wait for reduction of the backlog of spooled output. There is a significant difference in that HASP requires one job-queue record per job awaiting print or punch, whereas OS requires one DSB per Sysout data set, so the second element of the spool resource has much more predictable utilization in a HASP system.

Allocation of space to Sysout data sets is done dynamically on a record basis under HASP, but statically on a data set basis in OS. This means that, under HASP, available spool space is used for Sysout records as they are created, and the number of records created by each job is kept track of; when this number exceeds the estimate provided on the JOB card, the operator is informed and the job may be cancelled or allowed to continue. Under OS/MVT, the programmer must estimate the space for each Sysout data set in advance, and the job will be abnormally terminated if it attempts to write output in excess of the maximum space allocation. An OUTLIM parameter is also provided to allow limitation of output per data set on the basis of number of records written.

Measurement of spool utilization thus will differ, at least in its interpretation, between HASP and non-HASP systems. The first element, spool space, is readily quantified under HASP as number of DASD tracks available and used for spooling, and the percentage utilization is immediately available. Under OS, the total space available on public and storage DASD volumes could be subdivided into free space, space allocated to Sysout data sets (and to spooled input data sets), and space allocated to other data sets; then spool space as a percentage of the sum of spool space and free space would serve as a measure of relative spool space utilization comparable to HASP's percentage utilization figure.

Similarly, in either system, the number of Job Queue entries being used to control spooled data could be expressed as a percentage of the maximum number of entries less the number currently in use for other purposes. This would give a measure of the relative utilization of the second element of the spool resource, namely, spool control space.

Because HASP is not integrated into SMF in the MVT version of OS, many installations using HASP have added the capability to HASP to write SMF records for system output. This is essential if any measure of HASP output is to be provided using SMF.

The spool usage data provided by SMF consists of line (or card) counts from the Type 6 records and DASD space and EXCP counts from the Type 14/15 records. These records provide time relations to show print (or punch) volumes and backlog by hour. The data set records can be used to determine the amount of I/O activity to the spool devices, space allocated and used, and effectiveness of block sizes.

As mentioned earlier, many HASP systems have been modified to write Sysout records, and some write Sysin counts as well. In systems without HASP, the Sysin record count is available in the step termination record. These counts and the Sysout records provide the ability to allocate usage of Sysin devices on a step-by-step basis and that of Sysout devices on a job basis.

In HASP systems, the EXCP count for the pseudo-devices from the step termination records can be used to allocate the usage of HASP itself to jobs. This value is preferable to the Sysin/Sysout counts since the EXCP count is unaffected by multiple copy parameters.

8. Communications task

The (operator) communications task is responsible for communication between the system operators and OS components. The communications task receives messages from operating system components or user programs and queues them for display at the appropriate operator's console(s). In the other direction, the communications task receives commands from the system operators and routes them to command processors or user programs.

The two-way flow of messages through the communications task is determined by the type of console devices supported, the message buffers allocated, the availability of transient areas, and the availability of command processing tasks.

In order to measure the usage of the communications task, it is necessary to measure the usage of the message buffers, with such measures as: duration each buffer is in use, total time tasks are waiting for buffers to become available, and average number of messages queued for each console. Additionally, the I/O usage of consoles could be considered by examining the frequency and length of console I/O's. However, none of these measurements is supplied by SMF.

9. The SMF writer

The SMF Writer task (see section IV.A.3 above) is a serially used resource whose efficiency is determined by the size of the SMF buffer, the recording media for SMF data, and the availability of the SMFWTM (SVC 83) routine and the TCLOSE (SVC 23) routines.

Measures of the usage of the SMF Writer include the number of records written by time interval, the average size of the records, the I/O load on the SMF device (channel and speed of the device as reflected by I/O duration of the SMF writer), and the time spent by tasks awaiting completion of SMF writes or enqueued on the SMF buffer.

The SMF data itself provides a number of measures of the SMF writer: frequency of records written, the amount of segmentation of records, and the modal time between records. These measures can be used to evaluate the overall usage of the SMF writer and the possible impact of changes in the SMF parameters.

VI. Summary and Recommendations

The first five sections of this report have described how SMF works, how to install and operate it, and the uses to which it is customarily put in system performance evaluation. This final section deals with three important questions that should be asked about any measurement tool: How accurate is it? How expensive is it to use? How may it be adjusted to improve its usefulness? These questions were investigated with some rigor in the course of this study, and the results are summarized below.

For purposes of this study, the "accuracy" of SMF was considered in two respects: absolute and relative. Absolute accuracy was understood to mean the degree to which SMF measurements of some system characteristic approximated measurements of that same characteristic by another technique accepted as accurate (a hardware monitor, in this instance). Relative accuracy, or variability, was understood to mean the extent to which a fixed set of problem programs would produce consistent SMF measurements after repeated execution of the same set. Absolute accuracy is of particular interest in performance tuning, where the actual consequences of variations in the system environment may be critical. Relative accuracy tends to be more important in chargeback, where inconsistent user charges are apt to produce more complaints than poor service.

To test the absolute accuracy of SMF, one specific SMF data item (system wait time) was compared with carefully calibrated, external measurements of the same event. The job stream used for this controlled experiment contained 15 types of jobs, including several "synthetic" jobs designed to have closely controlled patterns of resource usage, and "natural" jobs chosen to be representative of typical batch workloads. Most of the "natural" jobs used IBM-supplied programs such as compilers, utilities and the linkage editor. These jobs were run repeatedly as a single fixed job stream on a dedicated machine under simultaneous monitoring by SMF and a commercial hardware monitor. The main conclusion from these tests was that SMF system wait time measurements agreed closely enough with hardware monitor measurement to be useful for system tuning. The CPU-busy value obtained by computing the difference between system wait time and elapsed wall-clock time were only 0.7% lower than comparable values obtained by hardware monitoring.

Note that these conclusions apply only to a total job stream. If precise CPU timings must be obtained for a single job using only SMF as a measurement tool, the authors recommend that the following procedure be employed:

* run a single-threaded test consisting of one dummy job and the job to be measured;

* switch and dump SMF data sets;

* switch SMF data sets again, and run another single-thread test containing only the dummy job;

* switch SMF data sets once more, discard the one consisting of records written in the interlude between tests, and dump the one for the second test;

* use cumulative SMF wait time and elapsed time to derive CPU-busy time for each test; and

* take the difference of these figures as the true CPU cost of running the job in question.

In the above procedure or any application of SMF system wait time the elapsed time should be taken as the difference between the time stamp on the Type 12 record and the time stamp on the previous Type 12 or Type 0 record. If this latter time is unavailable, it can be approximated by inspection of the operator's console log.

It should be noted that this procedure differs from the SMF manual, which recommends that the measurement interval for wait time "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."(<3>) This method ignores the time between the start of SMF wait time collection and the writing of the first Type 1 record. SMF wait time collection begins with an IPL and starts afresh every time a Z EOD (Halt End-of-Day) or I SMF (Switch SMF) command is processed. Thus the time stamp on the Type 0 record or the time stamp on the previous Type 12 record (which would usually be found on the other SM data set) should be taken as the beginning of the measurement interval. If the interval begins with an IPL, the amount of wait time reported in the Type 0 record should not be included in the total derived from the Type 1 and Type 12 records, as this wait time accrued before the time stamp on the Type 0. Ignoring the wait time in the first Type 1 record in a similar way would not salvage the method of <3> because only the upper level of the two-level buffer used by SMF to accumulate the wait time is emptied when the Type 1 record is written. The writing of a Type 0 or Type 12 record, on the other hand, empties both levels. Since each SMF recording interval in the validation experiment began and ended with a Switch SMF command, and only one SMF data set was dumped to tape for each test run, the Type 12 record for the first Switch command was not available for elapsed time calculation (or for any other purpose). Therefore, the elapsed time was taken from the console log instead. The next major objective of this phase of the study was to determine the relative accuracy, or variability, of two critical SMF measurements: CPU time and EXCP counts. The test job stream consisted of three jobs which were

run repeatedly in a production environment over a period of several weeks, during which pertinent SMF data was collected and tabulated. Then these same jobs were run single-threaded, i.e., with only one initiator active, and the figures derived from these runs were used as base values for a statistical analysis of the fluctuations observed in the production runs. The major results of this experiment showed (1) that CPU time recorded by SMF for individual job steps exhibited a random variability ranging up to 25% for typical batch jobs, (2) that SMF EXCP counts are generally accurate and repeatable as long as chained scheduling is not used, (Chained scheduling can introduce more than 80% variability into EXCP counts.), (3) that the use of chained scheduling increases the variability of SMF-reported step CPU times, (The coefficient of variation went from less than 5% to more than 7.5% in the validation experiment when chained scheduling was used.), and (4) a significant percentage (in our observation, 12%) of overall CPU-busy time is lost by SMF when allocating CPU time to jobs. (This time is inaccurately classified by SMF as "O.S. overhead".)

Because of its obvious impact on SMF measurements, a word about chained scheduling might be in order. Chained scheduling of sequential I/O is an OS option which causes the channel program to be modified dynamically in an attempt to keep the channel and device busy continuously. The CPU activity involved in this attempt is synchronized with the I/O by means of program-controlled interruptions (PCI's), a 360 hardware feature. This form of I/O involves unpredictable amounts of CPU overhead for supervisor routines modifying and restarting channel programs. Moreover, chained scheduling is highly timing-dependent, and hence extremely sensitive to environmental fluctuations.

EXCP counting is also quite unpredictable under chained scheduling. The median value of EXCP's under chained scheduling was lower than the standalone base value because on a loaded system the CPU is less likely to have another block ready to be written at the time a PCI occurs; in this situation the channel program is allowed to terminate, so the processing resembles that for ordinary sequential output. It is paradoxical but true that SMF always records more EXCP's for chained scheduling than without, even though fewer I/O operations are actually started at the hardware level.

As an aside, the variability experiment also showed that the predictability of CPU time on the basis of published instruction timings for the IBM System/360 Model 65 depends on the class of instructions being executed. Times measured for the types of instructions executed in fixed binary arithmetic (RR and RX) agree cosely with values predicted in advance. Timings for storage-to-storage moves and floating-point arithmetic were found to be much less predictable.

On the basis of code analysis, all of the above conclusions can be applied qualitatively to OS/MFT systems as well as to OS/MVT. These quantitative figures should be expected to vary from one OS installation to another dependng on hardware configuration, system generation options and workload characteristics. The claim that OS overhead is not accurately measured by SMF is also based on code analysis, which indicates that I/O interrupt processing time is charged to the interrupted task rather than to the task issuing the I/O, or is not charged to any task at all if the system is in the wait state when interrupted.

The analysis of the SMF EXCP counting algorithm earlier in this report revealed a bias against steps having a large number of statements, a situation that is found with many teleprocessing systems. This finding was confirmed by the experimental results on SMF data collection overhead. Code analysis of the MVS system has indicated that the use of chained scheduling for most sequential I/O will cause major irregularities in the EXCP counts recorded by SMF. This analysis further predicts that the EXCP counts for MVS chained scheduling will always be less than the counts for the nonchained scheduling, which in turn are always less than the counts for chained scheduling under OS (MFT or MVT). In any attempt to perceive reality it is important to know how much of what is observed is the observer's own shadow. To gather data on resource utilization in OS/360 itself requires resources, which may either be external to the system (e.g., a hardware monitor or sensor-driven minicomputer) or internalized in the form of additional overhead. Any software instrumenation installed in the operating system for performance analysis purposes should include some means of measuring both OS overhead and that fraction of it caused by the instrumentation itself.

Although SMF generally fails to account for the resources used by OS, the overhead incurred by SMF itself can be deduced from SMF records. There are two components of SMF overhead, namely, that which can be allocated to individual jobs (and in fact is included in job/step-related usage measures) and that which is pure OS overhead. For example, a job step which repeatedly opens and closes a file causes many SMF Type 14/15 records to be produced, and the CPU time required to construct the records and move them to the SMF buffer is charged to the step in the Type 4 record. The actual writing of the Type 14/15 records to the SMF data set is performed by the SMF writer, a system task, and thus is not charged to any job step nor explicitly recorded by SMF in any other way. However, the resources consumed in writing the Type 14/15 records can be estimated from the size and number of the record themselves.

Note that the "overhead" cost of SMF was considered in the rather restricted sense of the additional load, over and above user program execution, that is imposed upon a computer system by SMF data collection and recording, and is expressed in units of online resource consumption (system time, CPU time, supervisor time and I/O load). No consideration was given to the benefits

that prudent use of SMF can bring to an EDP installation. No estimate was made, either, of the development costs for producing SMF data reduction and report generation software, the personnel costs of installing and operating the SMF subsystem, or the media and storage costs for collecting and retaining historical SMF data. Clearly, these questions must be taken into consideration before a true cost/benefit assessment of the tool can be made.

The basic method used to measure SMF overhead was to run a test multiprogramming job stream under various operating system conditions. In effect, two versions of the operating system, one with SMF and one without, were used to isolate SMF data collection overhead. In the version with SMF, four levels of implementation were benchmarked: (1) SMF recording suppressed, (2) only system SMF records, (3) only system and job records, and (4) system, job step and data set records. Other variations in the test environment made it possible to assess the effects of varying SMF buffer size from the minimum allowed by OS to full-track blocking (i.e., SMF buffer size equal to twice the track capacity of the 2314 disk, since the SMF buffer is really a double buffer); and the effect on data collection overhead of adding a teleprocessing job to the test stream.

The results of these tests showed that SMF data collection overhead (i.e., SMF without data recording) was in the range of 6-8% of OS supervisor state CPU time. Almost two-thirds of this was incurred in EXCP counting alone. Overhead is substantially greater (by approximately 25%), both relatively and absolutely, in systems with teleprocessing than without it. SMF data recording overhead added less CPU overhead to the system than EXCP counting, and its cost in system I/O overhead is trivial except at sites that dedicate a tape drive to this function.

It is recommended, therefore, that any attempt to reduce SMF overhead should begin with optimization of the code in the I/O supervisor that maintains EXCP counts in the TCTIOT's for nonsystem tasks. This could result in overall CPU time savings of one or two percent, depending on the nature of the workload.

The all-around usefulness of SMF could be improved most markedly by having the I/O Interruption Supervisor determine which task is responsible for each I/O interruption as soon as possible after it occurs, and charge that task for the bulk of the time required to process the interrupt. Another major improvement would be to add code to the PUT routine or appendage routine for chained scheduling to keep a count of the number of successful joins, and add this to the EXCP count to arrive at a number of I/O operations which would at least be repeatable from one run to the next of the same job. Ideally, of course, all access method routines should provide data to SMF on the amount, rate and type of I/O activity they perform; this would permit analysis of the load on the I/O system in a way that EXCP counts will never provide.

With regard to EXCP counts, it is strongly recommended that DD statements for job steps that do any significant amount of I/O be arranged in decreasing order of I/O activity in order to minimize EXCP-counting overhead. If an MVS installation decides that the fluctuations due to chained scheduling make EXCP counts worthless for its purposes, it should modify the EXCP counting routine to return control immediately to its caller, leaving the EXCP count at zero, and completely eliminating the largest single source of SMF overhead.

Appendix A: Summary of Job Management

A. Master Scheduler

1. Console Interrupt Routine. Provides the Supervisor with the information necessary to give control to the proper routine following an attention interrupt.

2. Master Command EXCP Routine. Processes the CANCEL, DISPLAY, MOUNT, REQ, START, STOP, UNLOAD, and LOAD commands.

3. Master Command Routine. Analyzes command verbs and gives control to the proper command executors.

4. Write-to-operator Routine. Processes messages to the operator and operator replies.

5. External Interrupt Routine. Handles switch to the alternate console when external interruptions occur.

B. Reader/Interpreter

1. Control Routine. Reads and interprets control statements, passing control to the proper statement processing routine.

2. Job Routine. Analyzes JOB statements and builds Job Control Tables from their contents.

3. Execute Routine. Analyzes EXEC statements and builds Step Control Tables from their contents.

4. Data Definition Routine. Analyzes DD statements and constructs Job File Control BLocks and Step I/O Tables from their contents.

C. Initiator/Terminator

1. System Control Routine. Performs housekeeping requirements as the first stage in initiating a job.

2. Execute Statement Condition Code Routine. Processes condition codes specified in the EXEC statement.

3. JFCB Housekeeping Control Routine. Determines the proper JFCB processing routine and directs control to it.

4. Allocation Control Routine. Performs Initiator/Terminator housekeeping for allocation and setup.

5. Demand Allocation Routine. Constructs the Allocate Work Table and the Volume Table, and begins device allocation to data sets that are assigned specific devices.

6. Automatic Volume Recognition Routine. Performs allocation of devices on which volumes are mounted.

7. Decision Allocation Routine. Assigns devices to data sets not already allocated.

8. TIOT Construction Routine. Obtains space for and builds the TIOT for the processing program.

9. External Action Routine. Issues mounting instructions, verifies correctly mounted volumes, and unloads incorrectly mounted ones.

10. Space Request Routine. Processes requests for space on direct-access volumes.

11. Allocation Error Routine. Processes error conditions occurring during allocation and setup.

12. Step Initiation Routine. Performs all initiation operations required before control is passed to the processing program.

13. Step Termination Routine. Performs general task termination procedures in addition to passing control to the appropriate routines for disposition and deallocation of data sets and processing condition codes.

14. Job Termination Routine. Performs general job termination procedures in addition to passing control to the appropriate routine to release Job Queue, performing disposition and deallocation, and user accounting.

Appendix B: Summary of Task Management

A. Interrupt Handlers

When an interruption (Supervisor call, program check, external interrupt, I/O interrupt, or machine check) occurs, the old PSW is stored in low memory and the CPU loads the corresponding new PSW, also from a fixed low memory location. This new PSW causes entry to the appropriate First-Level Interruption Handler (FLIH), which analyzes the interruption and routes control to the appropriate routine. In some cases, a Second-Level Interruption Handler is invoked to perform further analysis. There are three options (SER0, SER1, and MCH) for the Machine Check FLIH, depending on the model of 360 CPU being used.

1. SVC FLIH
2. SVC SLIH
3. Program Check FLIH
4. External FLIH
5. Timer SLIH
6. Console Switch Routine
7. I/O FLIH
8. SER0
9. SER1
10. Machine Check Handler (MCH)

B. Type-1 SVC Routines

These routines receive control directly from the SVC FLIH; they are resident in the nucleus, run disabled for all interruptions except machine checks, and issue no SVC's of their own. Consequently, they need less housekeeping than other SVC routines. In what follows, SVC routines are denoted by the OS Supervisor macro-instructions which invoke them.

1. CHAP (SVC 44)
2. EXIT (SVC 3)
3. EXTRACT (SVC 40)
4. FREEMAIN (SVC 5)
5. GETMAIN (SVC 4)
6. POST (SVC 2)
7. TIME (SVC 11)
8. TTIMER (SVC 46)
9. WAIT (SVC 1)

C. Type-2 SVC Routines

These routines are resident in the nucleus, but may issue SVC's and/or enable interrupts. In particular, they may perform I/O. They run under the control of an SVRB (Supervisor Request Block) provided by the SVC SLIH.

1. ATTACH (SVC 42)
2. DELETE (SVC 9)
3. DEQ (SVC 48)
4. DETACH (SVC 62)
5. ENQ (SVC 56)
6. Exit Effector, Stage 1 (SVC 43)
7. IDENTIFY (SVC 41)
8. LINK (SVC 6), XCTL (SVC 7), LOAD (SVC 8), and SYNCH (SVC 12)
9. SEGLD or SEGWT (SVC 37) and Overlay Supervisor (SVC 45)
10. SPIE (SVC 14)
11. STIMER (SVC 47)

D. Non-resident SVC Routines

These routines reside in SYS1.SVCLIB and execute out of a transient area in the nucleus, unless they were preloaded into the Link Pack Area at IPL time. The transient area handler section of the SVC SLIH schedules the loading of non-resident SVC routines by the Transient Area Fetch routine as required.

1. ABDUMP and SNAP (SVC 51)
2. ABEND (SVC 13)
3. CHKPT (SVC 63)
4. Communications Task Router (SVC 72)
5. Log and WRITELOG Post (SVC 34)
6. RESTART (SVC 52)
7. STAE (SVC 60)

E. Non-SVC Service Routines

These nucleus-resident routines are used by the Supervisor itself to perform various task management functions.

1. ABTERM
2. ABTERM Prologue
3. DAR (Damage Assessment Routines)
4. Dispatcher
5. EOT (End-of-Task) Routine
6. Exit Effector, Stage 2
7. Exit Effector, Stage 3
8. Program Fetch
9. Task Switch Routine
10. Transient Area Fetch
11. Transient Area Refresh
12. Type-1 Exit Routine

F. Communications Task Routines

These routines, in conjunction with the Console Switch and SVC 72 routines, provide special Supervisor support for the Communications Task, which handles all traffic to and from the operator's console(s). This system task, established at IPL time, requires close co-ordination of Job Management (Master Scheduler), Data Management (IOS and access methods) and Task Management functions.

1. Console Support Routines
2. Device-Independent Display Operator Console Support (DIDOCS)
3. External Processor
4. Initialization Routine
5. Miscellaneous Lookup Services
6. Multiple Console Support (MCS)
7. Reply Processor
8. Wait Routine

Appendix C: Summary of Data Management

A. Input/Output Supervisor

1. EXCP Supervisor. Tests channels for activity, handles queuing and dequeuing, starts I/O operations, and performs EXCP validity checking.

2. I/O Interrupt Supervisor. Performs I/O interruption analysis, UCB lookup, error recovery, channel channel search and restart, I/O purge, and recording of error statistics.

3. Error Statistics by Volume Routine (SVC 91). Summarizes tape error statistics as an SMF Type 21 record.

B. Open/Close/EOV

1. OPEN Routine. Performs volume verification and label checking, fills in DCB's, builds needed DEB's, and determines and loads the required access method modules.

2. RDJFCB Routine. Reads into main memory the JFCB's associated with the DCB's in the parameter list.

3. CLOSE Routine. Performs labelling operations, purges queued and active I/O requests associated associated with data sets that are being operated on at the EXCP level, and keeps track of the access method modules necessary to restore each DCB to its status prior to OPEN.

4. EOV Routine. Processes end-of-volume and end-of-data set conditions for data sets having sequential organization.

C. Direct-Access Device Space Management

1. Allocate Routine (SVC 32). Allocates initial space to data sets.

2. Extend Routine. Allocates additional space to a data set at end of existing extents.

3. Scratch Routine (SVC 29). Deletes data sets by deleting their respective data set control blocks.

4. Release Routine. Returns unused space to available storage when a data set is closed.

5. Rename Routine (SVC 30). Changes the name of a referenced data set.

6. Obtain Routine (SVC 27). Provides direct access to any block in the VTOC.

7. LSPACE Routine (SVC 78). Summarizes the available space on a volume.

8. Protect Routine (SVC 98). Maintains the password data set and marks status of protected data sets.

D. Catalog Management

1. Catalog Routine. Finds entries in the catalog; inserts, deletes, or replaces data set pointer entries, volume control block pointer entries, and volume control blocks; enters into and deletes from the catalog the indexes, generation index pointer entries, control volume pointer entries, and alias entries.

2. CVOL Routine. Provides services to the Catalog Routine by opening catalog data sets for processing and by writing format blocks in new catalog data sets.

E. Access Methods

1. READ and WRITE Routines. Construct device-specific channel programs based on generalized user requests, and issue EXCP to schedule them for execution.

2. CHECK Routine. Waits for I/O completion, then performs error analysis and detects end-of-file or end-of-volume conditions for sequential data sets.

3. GET and PUT Routines. Combine functions of READ or WRITE and CHECK routines, plus buffer management and blocking and deblocking of records.

4. NOTE/POINT Routine. Records current location within a sequential file, or repositions the file to a previously noted location.

5. FIND and BLDL Routines. Locate the starting position of one or several member(s) of a partitioned data set.

6. Miscellaneous Routines. Perform device-dependent control functions such as stacker selection on a card punch or conditioning of telecommunication lines.

RECORD TYPE: 0 (IPL). BUILT BY: MASTER SCHEDULER SMF INITIALIZATION ROUTINE.
WRITTEN BY: SAME (MODULE IEESMFI2).
LENGTH: 31.

FIELD NAME	EXTERNAL SOURCE	SOURCE CONTROL BLOCK	CONTROL BLOCK CREATED BY	FIELD INITIALIZED BY	FIELD MAINTAINED BY	FORMAT	D E S C R I P T I O N
SMFOBUF	SMFDEFLT	SMCA	MASTER SCHEDULER	MASTER SCHEDULER	IEESMFI3	BL4	SMF BUFFER SIZE IN BYTES
SMFODTE	OS OPERATOR OR TOD CLOCK	CVT	NIP	SVC 83 ROUTINE	N/A	PL4	DATE IN PACKED DECIMAL FORM: 00YYDDDF
SMFOJWT	SMFDEFLT	SMCA	MASTER SCHEDULER	MASTER SCHEDULER	IEESMFI3	BL4	JOB WAIT TIME DEFAULT VALUE IN MINUTES
SMFOOPT	SMFDEFLT	JCT	MASTER SCHEDULER	MASTER SCHEDULER	IEESMFI3	BL1	SMF OPTIONS BYTE: SEE NOTE 1.
SMFORST	HARDWARE	CVT	NIP	IEESMFI2	N/A	BL4	REAL STORAGE SIZE IN 1K BLOCKS
SMFOSID	SMFDEFLT	SMCA	MASTER SCHEDULER	SVC 83 ROUTINE	N/A	CL4	SYSTEM IDENTIFICATION (h)
SMFOTME	OS OPERATOR OR TOD CLOCK	N/A	N/A	SVC 83 ROUTINE	N/A	BL4	TIME WHEN RECORD WAS MOVED TO OUTPUT BUFFER

NOTE 1 - THE USAGE AND SOURCE OF THE FLAGS IN THE SMF OPTIONS BYTE IS AS FOLLOWS:

BIT	HEX VALUE	MEANING WHEN SET	SMFDEFLT PARAMETERS
0	80	SYSTEM AND JOB DATA TO BE COLLECTED	OPT=1, MAN=ALL
1	40	SYSTEM, JOB, AND JOB STEP DATA TO BE COLLECTED	OPT=2, MAN=ALL
2	20	EXITS REQUESTED	EXT=YES
3	10	DATA SET ACCOUNTING	DSV=2 OR 3
4	08	VOLUME ACCOUNTING	DSV=1 OR 3
5	04	UNUSED	
6	02	TEMPORARY DATA SET SCRATCH RECORDS	REC=2
7	01	ALWAYS ZERO	

RECORD TYPE: 1 (WAIT TIME). BUILT BY: MASTER SCHEDULER SMF INITIALIZATION ROUTINE.
WRITTEN BY: TERMINATOR (MODULE IEFSMFLK).
LENGTH: 22.

FIELD NAME	EXTERNAL SOURCE	SOURCE CONTROL BLOCK	CONTROL BLOCK CREATED BY	FIELD INITIALIZED BY	FIELD MAINTAINED BY	FORMAT	D E S C R I P T I O N
SMF1DTE	OS OPERATOR OR TOD CLOCK	CVT	NIP	SVC 83 ROUTINE	N/A	PL4	DATE IN PACKED DECIMAL FORM: 00YYDDDF
SMF1SID	SMFDEFLT	SMCA	MASTER SCHEDULER	SVC 83 ROUTINE	N/A	CL4	SYSTEM IDENTIFICATION (h)
SMF1SWT	N/A	SMCA	MASTER SCHEDULER	MASTER SCHEDULER	TIMING SUPERVISOR	BL4	SYSTEM WAIT TIME IN HUNDREDTHS OF SECONDS (i), (j), (k)
SMF1TME	OS OPERATOR OR TOD CLOCK	N/A	N/A	SVC 83 ROUTINE	N/A	BL4	TIME WHEN RECORD WAS MOVED TO OUTPUT BUFFER

RECORD TYPE: 2 (DUMP HEADER). BUILT BY: SMF DUMP PROGRAM (MODULE IFASMFDP).
WRITTEN BY: SAME.
LENGTH: 18.

FIELD NAME	EXTERNAL SOURCE	SOURCE CONTROL BLOCK	CONTROL BLOCK CREATED BY	FIELD INITIALIZED BY	FIELD MAINTAINED BY	FORMAT	D E S C R I P T I O N
SMF2DTE	OS OPERATOR OR TOD CLOCK	CVT	NIP	IFASMFDP	N/A	PL4	DATE IN PACKED DECIMAL FORM: 00YYDDDF
SMF2SID	SMFDEFLT	SMCA	MASTER SCHEDULER	IFASMFDP	N/A	CL4	SYSTEM IDENTIFICATION (h)
SMF2TME	OS OPERATOR OR TOD CLOCK	N/A	N/A	IFASMFDP	N/A	BL4	TIME WHEN RECORD WAS MOVED TO OUTPUT BUFFER

RECORD TYPE: 3 (DUMP TRAILER). BUILT BY: SMF DUMP PROGRAM (MODULE IFASMFDP).
WRITTEN BY: SAME.
LENGTH: 18.

FIELD NAME	EXTERNAL SOURCE	SOURCE CONTROL BLOCK	CONTROL BLOCK CREATED BY	FIELD INITIALIZED BY	FIELD MAINTAINED BY	FORMAT	D E S C R I P T I O N
SMF3DTE	OS OPERATOR OR TOD CLOCK	CVT	NIP	IFASMFDP	N/A	PL4	DATE IN PACKED DECIMAL FORM: 00YYDDDF
SMF3SID	SMFDEFLT	SMCA	MASTER SCHEDULER	IFASMFDP	N/A	CL4	SYSTEM IDENTIFICATION (h)
SMF3TME	OS OPERATOR OR TOD CLOCK	N/A	N/A	IFASMFDP	N/A	BL4	TIME WHEN RECORD WAS MOVED TO OUTPUT BUFFER

RECORD TYPE: 4 (STEP TERMINATION). BUILT BY: TERMINATOR (MODULE IEFSMFWI).
WRITTEN BY: TERMINATOR (MODULE IEFSMFLK).
LENGTH: VARIABLE (AT LEAST 120).

FIELD NAME	EXTERNAL SOURCE	SOURCE CONTROL BLOCK	CONTROL BLOCK CREATED BY	FIELD INITIALIZED BY	FIELD MAINTAINED BY	FORMAT	D E S C R I P T I O N
SMF4ACTF	EXEC STATEMENT	STEP ACT	READER/ INTERPRETER	EXEC STMT PROCESSOR	EXEC STMT PROCESSOR	EACH FLD. PRECEDED BY ONE LENGTH BYTE	EXEC STATEMENT ACCOUNTING FIELDS --NUMBER IS GIVEN BY SMF4NAF
SMF4AST	TOD CLOCK	TCT	INITIATOR	INITIATOR	INITIATOR	BL4	DEVICE ALLOCATION START TIME
SMF4CUAD	SYSTEM PROGRAMMER	UCB	SYSGEN	IPL/NIP	IPL/NIP	BL2	CHANNEL AND UNIT ADDRESS (ONE PER DEVICE AND DATA SET)
SMF4DEVC	SYSTEM PROGRAMMER	UCB	SYSGEN	IPL/NIP	IPL/NIP	BL1	DEVICE CLASS (ONE PER DEVICE AND DATA SET)
SMF4DTE	OS OPERATOR OR TOD CLOCK	CVT	NIP	IEFSMFWI	N/A	PL4	DATE IN PACKED DECIMAL FORM: 00YYDDDF
SMF4EXCP	N/A	TCTIOT	INITIATOR (IEFSMFAT)	INITIATOR (IEFSMFAT)	IOS (IEASMFEX)	BL4	COUNT OF EXCP'S ISSUED FOR THE DEVICE AND DATA SET
SMF4HOST	N/A	TCT	INITIATOR (IEFSMFIE)	INITIATOR (IEFSMFAT)	GETMAIN (IEASMFGF)	BL2	STORAGE USED (IN 1K BLOCKS) (c), (d), (e)
SMF4JBN	JOB STATEMENT	JMR	READER/ INTERPRETER	JOB STMT PROCESSOR	INITIATOR	CL8	JOB NAME
SMF4NAF	EXEC STATEMENT	STEP ACT	READER/ INTERPRETER	EXEC STMT PROCESSOR	EXEC STMT PROCESSOR	BL1	NUMBER OF ACCOUNTING FIELDS

FIELD NAME	EXTERNAL SOURCE	SOURCE CONTROL BLOCK	CONTROL BLOCK CREATED BY	FIELD INITIALIZED BY	FIELD MAINTAINED BY	FORMAT	D E S C R I P T I O N
SMF4NCI	JOB STREAM	SCT	READER/ INTERPRETER	EXEC STMT PROCESSOR	INTERPRETER SPOOL RTINE (IEFVHG)	BL4	NUMBER OF CARD IMAGES IN DD DATA OR DD * DATA SETS
SMF4PGMN	EXEC STATEMENT	SCT	READER/ INTERPRETER	EXEC STMT PROCESSOR	EXEC STMT PROCESSOR	CL8	PROGRAM NAME, OR "PGM=*.DD"
SMF4PPST	OS OPERATOR OR TOD CLOCK	TCT	INITIATOR (IEFSMFIE)	IEFSMFIE	IEFSMFIE	BL4	PROBLEM PROGRAM LOAD TIME
SMF4PRTY	EXEC STATEMENT	JOB STEP TCB	INITIATOR	SUPERVISOR	SUPERVISOR	BL1	PRIORITY AT WHICH STEP WAS LAST DISPATCHED NOT MORE THAN 16 * A + B, WHERE DPRTY=(A,B) (a)
SMF4RDS	OS OPERATOR OR TOD CLOCK	JMR	READER/ INTERPRETER	READER	INITIATOR	PL4	JOB ENTRY DATE, FORMAT 00YYDDDF
SMF4RSHO	JOB OR EXEC STATEMENT	TCT	READER/ INTERPRETER	IEFSMFLK	N/A	BL2	REGION SIZE IN 1K BLOCKS REQUESTED (c), (d), (e)
SMF4RST	OS OPERATOR OR TOD CLOCK	JMR	READER/ INTERPRETER	READER	INITIATOR	BL4	JOB ENTRY TIME (IN 100THS OF SECONDS)
SMF4SCC	PROBLEM PGM OR SUPERVSR	LCT	INITIATOR INITIALIZA-TION RTNE.	IEFSMFIK	N/A	BL2	STEP COMPLETION CODE (f), (g)
SMF4SETM	N/A	STEP ACT	READER/ INTERPRETER	READER (TO ZEROS)	INITIATOR TASK-DELETE ROUTINE (IEFSD164)	BL3	STEP EXECUTION TIME (IN 100THS OF SECONDS) (f), (g)
SMF4SID	SMFDEFLT	JMR	READER/ INTERPRETER	IEFSMFWI	N/A	CL4	SYSTEM IDENTIFICATION (h)

FIELD NAME	EXTERNAL SOURCE	SOURCE CONTROL BLOCK	CONTROL BLOCK CREATED BY	FIELD INITIALIZED BY	FIELD MAINTAINED BY	FORMAT	D E S C R I P T I O N
SMF4SIT	OS OPERATOR OR TOD CLOCK	JCT	INTERPRETER	INITIATOR	INITIATOR	BL4	STEP INITIATION TIME (IN 100THS OF SECONDS)
SMF4SPK	MASTER SCHEDULER RESIDENT DATA AREA	TCB OF INITR.	ATTACH ROUTINE	INITIATOR INITIALIZ'N ROUTINE (IEFSD160)	IEFSD160	BL1 (KEY IN LEFT FOUR BITS--REST ARE ZEROS)	STORAGE PROTECTION KEY
SMF4STID	OS OPERATOR OR TOD CLOCK	JCT	INTERPRETER	INITIATOR	INITIATOR	PL4	STEP INITIATION DATE (PACKED DECIMAL FORMAT)
SMF4STMN	EXEC STATEMENT	SCT	READER/ INTERPRETER	EXEC STMT PROCESSOR	EXEC STMT PROCESSOR	CL8	STEP NAME
SMF4STN	NONE	JMR	READER/ INTERPRETER	READER (TO ZERO)	INITIATOR	BL1	STEP NUMBER (b)
SMF4TME	OS OPERATOR OR TOD CLOCK	N/A	N/A	IEFSMFWI	N/A	BL4	TIME WHEN RECORD WAS MOVED TO OUTPUT BUFFER
SMF4UIF	INSTALLATION STANDARD	JMR	READER/ INTERPRETER	READER (TO BLANKS)	USER EXIT ROUTINE	CL8	USER IDENTIFICATION FIELD
SMF4UTYP	SYSTEM PROGRAMMER	UCB	SYSGEN	IPL/NIP	IPL/NIP	BL1	UNIT TYPE (ONE PER DEVICE AND DATA SET)

RECORD TYPE: 5 (JOB TERMINATION). BUILT BY: TERMINATOR (MODULE IEFSMFWI).
WRITTEN BY: TERMINATOR (MODULE IEFSMFLK).
LENGTH: VARIABLE (AT LEAST 120).

FIELD NAME	EXTERNAL SOURCE	SOURCE CONTROL BLOCK	CONTROL BLOCK CREATED BY	FIELD INITIALIZED BY	FIELD MAINTAINED BY	FORMAT	D E S C R I P T I O N
SMF5ACTF	JOB CARD	JOB ACT	READER/ INTERPRETER	JOB STMT PROCESSOR	JOB STMT PROCESSOR	BL1	NUMBER OF ACCOUNTING FIELDS
SMF5DTE	OS OPERATOR OR TOD CLOCK	CVT	NIP	IEFSMFWI	N/A	PL4	DATE IN PACKED DECIMAL FORM: 00YYDDDF
SMF5JBN	JOB CARD	JMR	READER/ INTERPRETER	JOB STMT PROCESSOR	INITIATOR	CL8	JOB NAME
SMF5JBTI	NONE	JCT	INTERPRETER	JOB STMT PROCESSOR	INITIATOR/ TERMINATOR	BL1	JOB TERMINATION INDICATOR (f)
SMF5JCC	PROBLEM PGM OR SUPERVSR	LCT	INITIATOR INITIALIZ'N ROUTINE	IEFSMFLK	N/A	BL2	JOB COMPLETION CODE
SMF5JCPU	N/A	JOB ACT	READER/ INTERPRETER	READER (TO ZEROS)	INITIATOR TASK-DELETE ROUTINE (IEFSD164)	BL3	JOB CPU TIME (IN 100THS OF SECONDS)
SMF5JICL	JOB CARD	QMPA IN THE LCT	INITIATOR INITIALIZ'N ROUTINE	IEFSMFWI	N/A	CL1	JOB INPUT CLASS
SMF5JID	OS OPERATOR OR TOD CLOCK	JCT	INTERPRETER	INITIATOR	INITIATOR	PL4	JOB INITIATION DATE, PACKED DECIMAL FORMAT
SMF5JIT	OS OPERATOR OR TOD CLOCK	JCT	INTERPRETER	INITIATOR	INITIATOR	BL4	JOB INITIATION TIME (IN 100THS OF SECONDS)

FIELD NAME	EXTERNAL SOURCE	SOURCE CONTROL BLOCK	CONTROL BLOCK CREATED BY	FIELD INITIALIZED BY	FIELD MAINTAINED BY	FORMAT	D E S C R I P T I O N
SMF5JPTY	JOB CARD	JCT	INTERPRETER	JOB STMT PROCESSOR	JOB STMT PROCESSOR	BL1	JOB PRIORITY
SMF5JSAF	JOB CARD	JOB ACT	READER/ INTERPRETER	JOB STMT PROCESSOR	JOB STMT PROCESSOR	0C	JOB STATEMENT ACCOUNTING FIELDS (VARIABLE)
SMF5NCI	JOB STREAM	JMR	READER/ INTERPRETER	READER	IEFSMFWI	BL4	NUMBER OF CARD-IMAGE RECORDS IN DD DATA OR D
SMF5NST	NONE	JMR	READER/ INTERPRETER	READER (TO ZERO)	INITIATOR	BL1	NUMBER OF STEPS IN THE JOB
SMF5PRGN	JOB CARD	JOB ACT	INTERPRETER	JOB STMT PROCESSOR	USER'S ACCOUNTING ROUTINE	CL20	PROGRAMMER'S NAME
SMF5RDCL	SYSTEM PROGRAMMER	JMR	READER/ INTERPRETER	READER	READER	BL1	READER DEVICE CLASS
SMF5RSD	OS OPERATOR OR TOD CLOCK	JMR	READER/ INTERPRETER	READER	READER	PL4	JOB ENTRY DATE, PACKED DECIMAL FORMAT
SMF5RST	OS OPERATOR OR TOD CLOCK	JMR	READER/ INTERPRETER	READER	READER	BL4	JOB ENTRY TIME FOR JOB (IN HUNDREDTHS OF SECONDS)
SMF5RSTD	OS OPERATOR OR TOD CLOCK	JMR	READER/ INTERPRETER	READER	READER	PL4	READER END DATE FOR JOB IN PACKED FORMAT (00YYDDDF)
SMF5RSTT	OS OPERATOR OR TOD CLOCK	JMR	READER/ INTERPRETER	READER	READER	BL4	READER END TIME FOR JOB IN HUNDREDTHS OF SECONDS

FIELD NAME	EXTERNAL SOURCE	SOURCE CONTROL BLOCK	CONTROL BLOCK CREATED BY	FIELD INITIALIZED BY	FIELD MAINTAINED BY	FORMAT	D E S C R I P T I O N
SMF5RUTY	SYSTEM PROGRAMMER	JMR	READER/ INTERPRETER	READER	READER	BL1	READER UNIT TYPE
SMF5SID	SMFDEFLT	JMR	READER/ INTERPRETER	IEFSMFWI	N/A	CL4	SYSTEM IDENTIFICATION (h)
SMF5SMCI	JOB STREAM	JCT	INTERPRETER	IEFSMFWI	N/A	5BL1	SYSOUT CLASSES AND MSGCLASS INDICATOR
SMF5SPK	MASTER SCHEDULER RESIDENT DATA AREA	TCB OF INITR.	ATTACH ROUTINE	INITIATOR INITIALIZ'N ROUTINE (IEFSD160)	IEFSD160	BL1 (KEY IN LEFT FOUR BITS--REST ARE ZEROS)	STORAGE PROTECTION KEY
SMF5TME	OS OPERATOR OR TOD CLOCK	N/A	N/A	IEFSMFWI ROUTINE	N/A	BL4	TIME WHEN RECORD WAS MOVED TO OUTPUT BUFFER
SMF5UIF	INSTALLATION STANDARD	JMR	READER/ INTERPRETER	READER (TO BLANKS)	USER EXIT ROUTINE	CL8	USER IDENTIFICATION FIELD

NOTE 2 - THE USAGE OF THE FLAGS IN THE JOB TERMINATION INDICATOR BYTE IS AS FOLLOWS:

BIT	HEX VALUE	MEANING WHEN SET
0	80	UNUSED
1	40	CANCELLED AT EXIT IEFUJV
2	20	CANCELLED AT EXIT IEFUJI
3	10	CANCELLED AT EXIT IEFUSI
4	08	CANCELLED AT EXIT IEFACTRT
5	04	UNUSED
6	02	ABEND IN AT LEAST ONE STEP
7	01	UNUSED

RECORD TYPE: 6 (OUTPUT WRITER). BUILT BY: OUTPUT WRITER DSB HANDLER (MODULE IEFSD085).
WRITTEN BY: OUTPUT WRITER (IEFSD085 OR IEFSD079).
LENGTH: 61.

FIELD NAME	EXTERNAL SOURCE	SOURCE CONTROL BLOCK	CONTROL BLOCK CREATED BY	FIELD INITIALIZED BY	FIELD MAINTAINED BY	FORMAT	DESCRIPTION
SMF6DTE	OS OPERATOR OR TOD CLOCK	CVT	NIP	SVC 83 ROUTINE	N/A	PL4	DATE IN PACKED DECIMAL FORM: 00YYDDDF
SMF6FMN	DD STATEMENT	DSB	READER/ INTERPRETER	INTERPRETER	INTERPRETER	CL4	FORM NUMBER
SMF6IOE	NONE	NONE	N/A	IEFSD085	JOB DELETE ROUTINE (IEFSD079)	BL1	I/O ERROR INDICATOR: SEE NOTE 3.
SMF6JBN	JOB CARD	JMR	READER/ INTERPRETER	JOB STMT PROCESSOR	INITIATOR	CL8	JOB NAME
SMF6NDS	NONE	NONE	N/A	IEFSD085	DATA SET DELETE ROUTINE (IEFSD171)	BL1	NUMBER OF DATA SETS PROCESSED BY THE OUTPUT WRITER FOR THIS FORM NUMBER AND JOB
SMF6NLR	NONE	NONE	N/A	IEFSD085	STANDARD OUTPUT WRITER (IEFSD087)	BL4	NUMBER OF LOGICAL RECORDS FOR THE WRITER
SMF6OWC	DD STATEMENT	DSB	READER/ INTERPRETER	INTERPRETER	INTERPRETER	CL1	OUTPUT WRITER CLASS
SMF6RSD	OS OPERATOR OR TOD CLOCK	JMR	READER/ INTERPRETER	READER	INITIATOR	PL4	JOB ENTRY DATE: 00YYDDDF

FIELD NAME	EXTERNAL SOURCE	SOURCE CONTROL BLOCK	CONTROL BLOCK CREATED BY	FIELD INITIALIZED BY	FIELD MAINTAINED BY	FORMAT	D E S C R I P T I O N
SMF6RST	OS OPERATOR OR TOD CLOCK	JMR	READER/ INTERPRETER	READER	INITIATOR	BL4	JOB ENTRY TIME
SMF6SID	SMFDEFLT	SMCA	MASTER SCHEDULER	SVC 83 ROUTINE	N/A	CL4	SYSTEM IDENTIFICATION (h)
SMF6TME	OS OPERATOR OR TOD CLOCK	N/A	N/A	SVC 83 ROUTINE	N/A	BL4	TIME WHEN RECORD WAS MOVED TO OUTPUT BUFFER
SMF6UIF	INSTALLATION STANDARD	JMR	READER/ INTERPRETER	READER (TO BLANKS)	USER EXIT ROUTINE	CL8	USER ID FIELD
SMF6WSD	OS OPERATOR OR TOD CLOCK	CVT	NIP	IEFSD085	N/A	PL4	WRITER START DATE
SMF6WST	OS OPERATOR OR TOD CLOCK	N/A	N/A	IEFSD085	N/A	BL4	WRITER START TIME

NOTE 3 - THE USAGE OF THE FLAGS IN THE I/O ERROR INDICATOR BYTE IS AS FOLLOWS:

BIT	HEX VALUE	MEANING WHEN SET
0	80	UNUSED
1	40	UNUSED
2	20	UNUSED
3	10	UNUSED
4	08	I/O DISCONTINUED (REMOTE OUTPUT ONLY)
5	04	INPUT ERROR ON THE SPOOL DATA SET
6	02	OUTPUT ERROR
7	01	INPUT ERROR ON SYS1.SYSJOBQE

RECORD TYPE: 7 (DATA LOST). BUILT BY: SMF WRITER (MODULE IEESMFWT).
WRITTEN BY: SAME.
LENGTH: 28.

FIELD NAME	EXTERNAL SOURCE	SOURCE CONTROL BLOCK	CONTROL BLOCK CREATED BY	FIELD INITIALIZED BY	FIELD MAINTAINED BY	FORMAT	D E S C R I P T I O N
SMF7DTE	OS OPERATOR OR TOD CLOCK	CVT	NIP	SVC 83 ROUTINE	N/A	PL4	DATE IN PACKED DECIMAL FORM: 00YYDDDF
SMF7NRO	NONE	SMCA	MASTER SCHEDULER	IEESMFWT	IEESMFWT	BL2	NUMBER OF SMF RECORDS OMITTED FROM DATA SET
SMF7SID	SMFDEFLT	SMCA	MASTER SCHEDULER	SVC 83 ROUTINE	N/A	CL4	SYSTEM IDENTIFICATION (h)
SMF7STD	OS OPERATOR OR TOD CLOCK	SMCA	MASTER SCHEDULER	IEESMFWT	IEESMFWT	PL4	DATE OF START OF DATA LOSS
SMF7STM	OS OPERATOR OR TOD CLOCK	SMCA	MASTER SCHEDULER	IEESMFWT	IEESMFWT	BL4	TIME OF START OF DATA LOSS
SMF7TME	OS OPERATOR OR TOD CLOCK	N/A	N/A	SVC 83 ROUTINE	N/A	BL4	TIME WHEN RECORD WAS MOVED TO OUTPUT BUFFER

RECORD TYPE: 8 (ONLINE DEVICES (I/O CONFIGURATION)).

BUILT BY: MASTER SCHEDULER SMF INITIALIZATION ROUTINE.
WRITTEN BY: SAME (MODULE IEESMFI2).
LENGTH: VARIABLE (18+4*(NUMBER OF DEVICES ON-LINE AND READY AT IPL)).

FIELD NAME	EXTERNAL SOURCE	SOURCE CONTROL BLOCK	CONTROL BLOCK CREATED BY	FIELD INITIALIZED BY	FIELD MAINTAINED BY	FORMAT	D E S C R I P T I O N
SMF8DTE	OS OPERATOR OR TOD CLOCK	CVT	NIP	SVC 83 ROUTINE	N/A	PL4	DATE IN PACKED DECIMAL FORM: 00YYDDDF
SMF8IODV	SYSTEM PROGRAMMER	UCB	SYSGEN	IEESMFI2	N/A	OCL4	ONE FOUR-BYTE AREA FOR EACH DEVICE CONTAINING DEVICE CLASS AND TYPE, BIN NUMBER, AND CHANNEL AND UNIT ADDRESS
SMF8SID	SMFDEFLT	SMCA	MASTER SCHEDULER	SVC 83 ROUTINE	N/A	CL4	SYSTEM IDENTIFICATION (h)
SMF8TME	OS OPERATOR OR TOD CLOCK	N/A	N/A	SVC 83 ROUTINE	N/A	BL4	TIME WHEN RECORD WAS MOVED TO OUTPUT BUFFER

RECORD TYPE: 9 (VARY ONLINE). BUILT BY: VARY COMMAND PROCESSOR (MODULE IEE2303D).
WRITTEN BY: SAME.
LENGTH: VARIABLE (18+4*(NUMBER OF ELEMENTS VARIED ONLINE)).

FIELD NAME	EXTERNAL SOURCE	SOURCE CONTROL BLOCK	CONTROL BLOCK CREATED BY	FIELD INITIALIZED BY	FIELD MAINTAINED BY	FORMAT	DESCRIPTION
SMF9DTE	OS OPERATOR OR TOD CLOCK	CVT	NIP	SVC 83 ROUTINE	N/A	PL4	DATE IN PACKED DECIMAL FORM: 00YYDDDF
SMF9DVAD	HARDWARE	UCB (DEVICES ONLY)	SYSGEN (DEVICES ONLY)	IEE2303D	N/A	OCL4	FOUR-BYTE AREA FOR EACH CPU, CHANNEL, I/O DEVICE OR STORAGE BOX ADDED TO THE CONFIGURATION
SMF9SID	SMFDEFLT	SMCA	MASTER SCHEDULER	SVC 83 ROUTINE	N/A	CL4	SYSTEM IDENTIFICATION (h)
SMF9TME	OS OPERATOR OR TOD CLOCK	N/A	N/A	SVC 83 ROUTINE	N/A	BL4	TIME WHEN RECORD WAS MOVED TO OUTPUT BUFFER

RECORD TYPE: 10 (ALLOCATION RECOVERY). BUILT BY: ALLOCATION RECOVERY ROUTINE (MODULE IEFXJIMP).
WRITTEN BY: SAME.
LENGTH: VARIABLE (18+4*(NUMBER OF DEVICES MADE AVAILABLE)).

FIELD NAME	EXTERNAL SOURCE	SOURCE CONTROL BLOCK	CONTROL BLOCK CREATED BY	FIELD INITIALIZED BY	FIELD MAINTAINED BY	FORMAT	D E S C R I P T I O N
SMF10DEV	SYSTEM PROGRAMMER	UCB	SYSGEN	IEFXJIMP	N/A	0CL4	FOUR-BYTE AREA FOR EACH DEVICE BEING MADE AVAILABLE: CONTAINS DEVICE CLASS AND TYPE, BIN NUMBER, AND CHANNEL AND UNIT ADDRESS
SMF10DTE	OS OPERATOR OR TOD CLOCK	CVT	NIP	SVC 83 ROUTINE	N/A	PL4	DATE IN PACKED DECIMAL FORM: 00YYDDDF
SMF10JBN	JOB CARD	JMR	READER/ INTERPRETER	JOB STMT PROCESSOR	INITIATOR	CL8	JOB NAME
SMF10RSD	OS OPERATOR OR TOD CLOCK	JMR	READER/ INTERPRETER	READER	INITIATOR	PL4	JOB ENTRY DATE, AS 00YYDDDF
SMF10RST	OS OPERATOR OR TOD CLOCK	JMR	READER/ INTERPRETER	READER	INITIATOR	BL4	JOB ENTRY TIME (100THS OF SECONDS)
SMF10SID	SMFDEFLT	SMCA	MASTER SCHEDULER	SVC 83 ROUTINE	N/A	CL4	SYSTEM IDENTIFICATION (h)
SMF10TME	OS OPERATOR OR TOD CLOCK	N/A	N/A	SVC 83 ROUTINE	N/A	BL4	TIME WHEN RECORD WAS MOVED TO OUTPUT BUFFER
SMF10UIF	INSTALLATION STANDARD	JMR	READER/ INTERPRETER	READER (TO BLANKS)	USER EXIT ROUTINE	CL8	USER IDENTIFICATION FIELD

RECORD TYPE: 11 (VARY OFFLINE). BUILT BY: VARY COMMAND PROCESSOR (MODULE IEE2303D).
WRITTEN BY: SAME.
LENGTH: VARIABLE (18+4*(NUMBER OF ELEMENTS VARIED OFFLINE)).

FIELD NAME	EXTERNAL SOURCE	SOURCE CONTROL BLOCK	CONTROL BLOCK CREATED BY	FIELD INITIALIZED BY	FIELD MAINTAINED BY	FORMAT	D E S C R I P T I O N
SMF11DEV	HARDWARE	UCB (DEVICES ONLY)	SYSGEN (DEVICES ONLY)	IEE2303D	N/A	OCL4	FOUR-BYTE AREA FOR EACH CPU, CHANNEL, I/O DEVICE OR STORAGE BOX REMOVED FROM THE CONFIGURATION
SMF11DTE	OS OPERATOR OR TOD CLOCK	CVT	NIP	SVC 83 ROUTINE	N/A	PL4	DATE IN PACKED DECIMAL FORM: 00YYDDDF
SMF11SID	SMFDEFLT	SMCA	MASTER SCHEDULER	SVC 83 ROUTINE	N/A	CL4	SYSTEM IDENTIFICATION (h)
SMF11TME	OS OPERATOR OR TOD CLOCK	N/A	N/A	SVC 83 ROUTINE	N/A	BL4	TIME WHEN RECORD WAS MOVED TO OUTPUT BUFFER

RECORD TYPE: 12 (END-OF-DAY). BUILT BY: HALT COMMAND PROCESSOR (MODULE IEE1403D).
WRITTEN BY: SAME.
LENGTH: 22.

FIELD NAME	EXTERNAL SOURCE	SOURCE CONTROL BLOCK	CONTROL BLOCK CREATED BY	FIELD INITIALIZED BY	FIELD MAINTAINED BY	FORMAT	D E S C R I P T I O N
SMF12DTE	OS OPERATOR OR TOD CLOCK	CVT	NIP	SVC 83 ROUTINE	N/A	PL4	DATE IN PACKED DECIMAL FORM: 00YYDDDF
SMF12SID	SMFDEFLT	SMCA	MASTER SCHEDULER	SVC 83 ROUTINE	N/A	CL4	SYSTEM IDENTIFICATION (h)
SMF12SWT	N/A	SMCA & WAIT TIME SAVE AREA	MASTER SCHEDULER	IEE1403D	N/A	BL4	SYSTEM WAIT TIME IN HUNDREDTHS OF SECONDS (i), (j), (k)
SMF12TME	OS OPERATOR OR TOD CLOCK	N/A	N/A	SVC 83 ROUTINE	N/A	BL4	TIME WHEN RECORD WAS MOVED TO OUTPUT BUFFER

RECORD TYPE: 14 (CLOSE/EOV FOR INPUT).
(RECORD TYPE 15 (CLOSE/EOV FOR OUTPUT)
CONTENTS ARE IDENTICAL TO TYPE 14.)

BUILT BY: CLOSE/EOV (MODULES IFG0202H AND IFG0202I).
WRITTEN BY: SAME.
LENGTH: VARIABLE (AT LEAST 292).

FIELD NAME	EXTERNAL SOURCE	SOURCE CONTROL BLOCK	CONTROL BLOCK CREATED BY	FIELD INITIALIZED BY	FIELD MAINTAINED BY	FORMAT	D E S C R I P T I O N
SMFDCBBL	N/A	DCB	PROBLEM PROGRAM	ASSEMBLER OR COMPILER	OPEN, ACCESS METHOD	BL4	DCBBLKCT (BLOCK COUNT FOR THIS VOLUME--MAGNETIC TAPE ONLY)
SMFDCBFL	N/A	DCB	PROBLEM PROGRAM	ASSEMBLER OR COMPILER	OPEN, ACCESS METHOD	BL1	DCBOFLGS (OPEN FLAGS)
SMFDCBMA	PROBLEM PROGRAM	DCB	PROBLEM PROGRAM	PROBLEM PROGRAM	PROBLEM PROGRAM	BL1	DCBMAC (EXTENSION OF THE DCBMACRF FIELD FOR ISAM)
SMFDCBMF	PROBLEM PROGRAM	DCB	PROBLEM PROGRAM	PROBLEM PROGRAM	PROBLEM PROGRAM	BL2	DCBMACRF (MACRO REFERENCES)
SMFDCBNL	N/A	DCB	PROBLEM PROGRAM	ASSEMBLER OR COMPILER	OPEN, ACCESS METHOD	BL1	DCBNLEV (NUMBER OF INDEX LEVELS--ISAM ONLY)
SMFDCBNO	DD STATEMENT	DCB	PROBLEM PROGRAM	ASSEMBLER OR COMPILER	OPEN, ACCESS METHOD	BL2	DCBNOREC (NUMBER OF RECORDS IN AN OVERFLOW AREA--ISAM ONLY)
SMFDCBNR	DD STATEMENT	DCB	PROBLEM PROGRAM	ASSEMBLER OR COMPILER	OPEN, ACCESS METHOD	BL4	DCBNREC (NUMBER OF RECORDS IN PRIME DATA AREA--ISAM ONLY)
SMFDCBOP	PROBLEM PROGRAM OR DD STATEMENT	DCB	PROBLEM PROGRAM	PROBLEM PROGRAM	PROBLEM PROGRAM	BL1	DCBOPTCD (OPTION CODES)
SMFDCBOR	PROBLEM PROGRAM	DCB	PROBLEM PROGRAM	PROBLEM PROGRAM	PROBLEM PROGRAM	BL2	DCBDSORG (DATA SET ORGANIZATION)

FIELD NAME	EXTERNAL SOURCE	SOURCE CONTROL BLOCK	CONTROL BLOCK CREATED BY	FIELD INITIALIZED BY	FIELD MAINTAINED BY	FORMAT	D E S C R I P T I O N
SMFDCBRF	PROBLEM PROGRAM, LABEL, OR DD STATEMENT	DCB	PROBLEM PROGRAM	PROBLEM PROGRAM	OPEN	BL1	DCBRECFM (RECORD FORMAT)
SMFDCBR1	FORMAT 2 DSCB	DCB	PROBLEM PROGRAM	ASSEMBLER OR COMPILER	OPEN, ACCESS METHOD	BL2	DCBRORG1 (NUMBER OF CYLINDER OVERFLOW AREAS THAT ARE FULL--ISAM ONLY)
SMFDCBR2	FORMAT 2 DSCB	DCB	PROBLEM PROGRAM	ASSEMBLER OR COMPILER	OPEN, ACCESS METHOD	BL2	DCBRORG2 (NUMBER OF TRACKS LEFT IN OVERFLOW AREA--ISAM ONLY)
SMFDCBR3	N/A	DCB	PROBLEM PROGRAM	ASSEMBLER OR COMPILER	ACCESS METHOD	BL4	DCBRORG3 (COUNT OF ACCESSES TO OVERFLOW RECORDS OTHER THAN THE FIRST--ISAM ONLY)
SMFDEBFL	DD STATEMENT	DEB	OPEN	OPEN	IOS	BL1	DEBOFLGS (DATA SET STATUS)
SMFDEBNI	FORMAT 2 DSCB	DEB	OPEN	OPEN	ACCESS METHOD	BL1	DEBNIEE (NUMBER OF INDEX EXTENTS--ISAM ONLY)
SMFDEBNO	FORMAT 2 DSCB	DEB	OPEN	OPEN	ACCESS METHOD	BL1	DEBNOEE (NUMBER OF OVERFLOW EXTENTS--ISAM ONLY)
SMFDEBNP	FORMAT 2 DSCB	DEB	OPEN	OPEN	ACCESS METHOD	BL1	DEBNPEE (NUMBER OF PRIME AREA EXTENTS--ISAM ONLY)
SMFDEBOP	PROBLEM PROGRAM	DEB	OPEN	OPEN	ABEND ROUTINE	BL1	DEBOPATB (OPEN/CLOSE/EOV OPTIONS)
SMFDEBVL	DATA SET HDR1 LABEL	DEB	OPEN/EOV	OPEN	EOV	BL2	DEBVOLSQ (VOLUME SEQUENCE NUMBER FOR MULTIVOLUME SEQUENTIAL DATA SETS)

FIELD NAME	EXTERNAL SOURCE	SOURCE CONTROL BLOCK	CONTROL BLOCK CREATED BY	FIELD INITIALIZED BY	FIELD MAINTAINED BY	FORMAT	D E S C R I P T I O N
SMFDSSNO	DATA SET HDR1 LABEL	UCB	SYSGEN	IPL/NIP	OPEN	CL6	DATA SET SERIAL NUMBER--VOLUME SERIAL NUMBER OF FIRST TAPE VOLUME (1)
SMFEXCP	N/A	TCTIOT	INITIATOR (IEFSMFAT)	INITIATOR (IEFSMFAT)	IOS (IEASMFEX)	BL4	EXCP COUNT
SMFJFCB1	DD STATEMENT	JFCB	INTERPRETER	N/A	N/A	XL176	JFCB SEGMENT (ENTIRE JFCB)
SMFNCYLS	VTOC	DEB	OPEN	IFG0202I	N/A	BL2	NUMBER OF CYLINDERS IN INDEPENDENT INDX AREA
SMFNOCYL	VTOC	DEB	OPEN	IFG0202I	N/A	BL2	NUMBER OF CYLINDERS IN INDEPENDENT OVFL AREA
SMFNPCYL	VTOC	DEB	OPEN	IFG0202I	N/A	BL2	NUMBER OF CYLINDERS IN PRIME AREA
SMFSRTEF	MAGNETIC TAPE REEL	UCB	SYSGEN	IPL/NIP	OPEN/EOV	BL2	SRTEFSCT (PHYSICAL DATA SET SEQUENCE COUNT--MAGNETIC TAPE ONLY)
SMFSRTEQ	DATA SET HDR1 LABEL	UCB	SYSGEN	IPL/NIP	OPEN/EOV	BL2	SRTEFSEQ (LOGICAL DATA SET SEQUENCE NUMBER--MAGNETIC TAPE ONLY)
SMFSRTES	MOUNT COMMAND OR DD STATEMENT	UCB	SYSGEN	IPL/NIP	MOUNT COMMAND PROCR. (IEEVMNT2)	BL1	SRTESTAB (VOLUME STATUS: SHARABILITY AND PRIVATE/PUBLIC/STORAGE)
SMFSRTEV	VOLUME LABEL	UCB	SYSGEN	IPL/NIP	INITIATOR (IEFWD000)	CL6	SRTEVOLI (VOLUME SERIAL NUMBER)
SMFTIOE1	N/A	TIOT	INITIATOR	INITIATOR	INITIATOR	BL1	TIOELNGH (LENGTH OF THIS DD ENTRY)

FIELD NAME	EXTERNAL SOURCE	SOURCE CONTROL BLOCK	CONTROL BLOCK CREATED BY	FIELD INITIALIZED BY	FIELD MAINTAINED BY	FORMAT	D E S C R I P T I O N
SMFTIOE2	DD STATEMENT	TIOT	INITIATOR	INITIATOR	INITIATOR	BL1	TIOESSTA (STATUS BYTE: TAPE LABEL PROCESSING TYPE & ALLOCATION FLAGS)
SMFTIOE3	DD STATEMENT	TIOT	INITIATOR	INITIATOR	INITIATOR	BL1	TIOEWTCT (NUMBER OF DEVICES REQUESTED FOR THIS DATA SET)
SMFTIOE4	N/A	TIOT	INITIATOR	INITIATOR	TSO OR TCAM	BL1	TIOELINK (TSO AND TCAM FLAGS)
SMFTIOE5	DD STATEMENT	TIOT	INITIATOR	INITIATOR	INITIATOR	CL8	TIOEDDNM (DD NAME)
SMFTIOE6	SYSTEM WORK QUEUE	TIOT	INITIATOR	INITIATOR	INITIATOR	BL3	TIOEJFCB (RELATIVE TRACK ADDRESS OF THE JFCB)
SMFTIOE7	N/A	TIOT	INITIATOR	INITIATOR	INITIATOR	BL1	TIOESTTC (ALLOCATION STATUS BYTE: ALWAYS ZERO AT CLOSE/EOV TIME)
SMFUCBCH	SYSTEM PROGRAMMER	UCB	SYSGEN	IPL/NIP	IOS	BL1	UCBCHA (CHANNEL ADDRESS AND FLAGS)
SMFUCBTY	SYSTEM PROGRAMMER	UCB	SYSGEN	IPL/NIP	IPL/NIP	BL4	UCBTYP (DEVICE TYPE)
SMFUCBUA	SYSTEM PROGRAMMER	UCB	SYSGEN	IPL/NIP	IPL/NIP	BL1	UCBUA (UNIT ADDRESS)
SMF14DTE	OS OPERATOR OR TOD CLOCK	CVT	NIP	SVC 83 ROUTINE	N/A	PL4	DATE IN PACKED DECIMAL FORM: 0CYYDDDF

FIELD NAME	EXTERNAL SOURCE	SOURCE CONTROL BLOCK	CONTROL BLOCK CREATED BY	FIELD INITIALIZED BY	FIELD MAINTAINED BY	FORMAT	D E S C R I P T I O N
SMF14JBN	JOB CARD	JMR	READER/ INTERPRETER	READER	READER	CL8	JOB NAME
SMF14NER	"RLSE" SUB-PARAMETER	TCTIOT	INITIATOR (IEFSMFAT)	INITIATOR (TO ZEROS)	RELEASE ROUTINE (IGG020P2)	BL1	NUMBER OF EXTENTS RELEASED BY DADSM
SMF14NEX	VTOC	DEB	OPEN/EOV	OPEN	EOV	BL1	NUMBER OF EXTENTS
SMF14NTA	VTOC	DEB	OPEN/EOV	OPEN	EOV	BL4	NUMBER OF TRACKS ALLOCATED
SMF14NTR	"RLSE" SUB-PARAMETER	TCTIOT	INITIATOR (IEFSMFAT)	INITIATOR (TO ZEROS)	RELEASE ROUTINE (IGG020P2)	BL1	NUMBER OF TRACKS RELEASED BY DADSM RELSE RTN
SMF14NTU	FORMAT 1 DSCB	DCB	PROBLEM PROGRAM	OPEN	ACCESS METHOD	BL4	NUMBER OF TRACKS USED, GIVEN AS LAST TTR (DCBFDAD CONVERTED TO TTRN FORMAT)
SMF14NUC	NONE	NONE	N/A	CLOSE/EOV	N/A	BL1	NUMBER OF UCB SEGMENTS
SMF14RIN	VARIOUS SOURCES	DCB, JFCB, UCB	VARIOUS COMPONENTS	IFG0202H	N/A	BL2	RECORD INDICATORS: SEE NOTE 4.
SMF14RSD	OS OPERATOR OR TOD CLOCK	JMR	READER/ INTERPRETER	READER	INITIATOR	PL4	JOB ENTRY DATE, PACKED FORMAT: 00YYDDDF
SMF14RST	OS OPERATOR OR TOD CLOCK	JMR	READER/ INTERPRETER	READER	INITIATOR	BL4	JOB ENTRY TIME (100THS OF SECONDS)
SMF14SDC	ASSEMBLED INTO CODE	NONE	N/A	IFG0202H	N/A	BL1	SIZE OF DCB/DEB SEGMENT (CONSTANT: 24 BYTES)

FIELD NAME	EXTERNAL SOURCE	SOURCE CONTROL BLOCK	CONTROL BLOCK CREATED BY	FIELD INITIALIZED BY	FIELD MAINTAINED BY	FORMAT	D E S C R I P T I O N
SMF14SET	ASSEMBLED INTO CODE	NONE	N/A	IFG0202I	N/A	BL1	SIZE OF ISAM EXTENSION SEGMENT (CONSTANT: 28 BYTES)
SMF14SID	SMFDEFLT	SMCA	MASTER SCHEDULER	SVC 83 ROUTINE	N/A	CL4	SYSTEM IDENTIFICATION (h)
SMF14SUC	ASSEMBLED INTO CODE	NONE	N/A	IFG0202H	N/A	BL1	SIZE OF EACH UCB SEGMENT (CONSTANT: 24 BYTES)
SMF14TME	OS OPERATOR OR TOD CLOCK	N/A	N/A	SVC 83 ROUTINE	N/A	BL4	TIME WHEN RECORD WAS MOVED TO OUTPUT BUFFER
SMF14UID	INSTALLATION STANDARD	JMR	READER/ INTERPRETER	READER (TO BLANKS)	USER EXIT ROUTINE	CL8	USER IDENTIFICATION

NOTE 4 - THE SECOND BYTE OF THE RECORD INDICATORS FIELD (SMF14RIN) IS ENTIRELY UNUSED.
THE FLAGS IN THE FIRST BYTE HAVE THE FOLLOWING MEANINGS:

BIT	HEX VALUE	MEANING WHEN SET
0	80	UNUSED
1	40	THIS RECORD WAS WRITTEN BY EOV, NOT BY CLOSE
2	20	DEVICE IS DIRECT-ACCESS TYPE
3	10	DATA SET IS TEMPORARY
4	08	DATA SET IS DIRECT-ACCESS ("DSORG=DA" IN THE DCB)
5	04	DATA SET IS INDEXED SEQUENTIAL ("DSORG=IS" IN THE DCB)
6	02	DATA SET IS INDEXED SEQUENTIAL ("DSORG=IS" ON THE DD CARD)
7	01	UNUSED

RECORD TYPE: 17 (SCRATCH). BUILT BY: SVC 29 ROUTINE (MODULE IGG0290D).
WRITTEN BY: SAME.
LENGTH: VARIABLE (AT LEAST 100).

FIELD NAME	EXTERNAL SOURCE	SOURCE CONTROL BLOCK	CONTROL BLOCK CREATED BY	FIELD INITIALIZED BY	FIELD MAINTAINED BY	FORMAT	D E S C R I P T I O N
SMF17DSN	IEHPROGM CNTRL STMT OR DD STMT	SCRATCH PARAM. LIST	ISSUER OF SVC 29	IGG0290D	N/A	CL44	DSNAME
SMF17DTE	OS OPERATOR OR TOD CLOCK	CVT	NIP	SVC 83 ROUTINE	N/A	PL4	DATE IN PACKED DECIMAL FORM: 00YYDDDF
SMF17FVL	VOLUME LABEL	UCB	SYSGEN	IPL/NIP	INITIATOR (IEFWD000)	CL6	FIRST VOLUME SERIAL NUMBER (SRTEVOLI)
SMF17JBN	JOB CARD	JMR	READER/ INTERPRETER	JOB STMT PROCESSOR	INITIATOR	CL8	JOB NAME
SMF17NVL	IEHPROGM CNTRL STMT OR DD STMT	SCRATCH PARAM. LIST	ISSUER OF SVC 29	IGG0290D	N/A	BL1	NUMBER OF VOLUME SERIAL NUMBERS SPECIFIED
SMF17RSD	OS OPERATOR OR TOD CLOCK	JMR	READER/ INTERPRETER	READER	INITIATOR	PL4	JOB ENTRY DATE, PACKED FORMAT: 00YYDDDF
SMF17RST	OS OPERATOR OR TOD CLOCK	JMR	READER/ INTERPRETER	READER	INITIATOR	BL4	JOB ENTRY TIME
SMF17SID	SMFDEFLT	SMCA	MASTER SCHEDULER	SVC 83 ROUTINE	N/A	CL4	SYSTEM IDENTIFICATION (h)
SMF17TME	OS OPERATOR OR TOD CLOCK	N/A	N/A	SVC 83 ROUTINE	N/A	BL4	TIME WHEN RECORD WAS MOVED TO OUTPUT BUFFER
SMF17UID	INSTALLATION STANDARD	JMR	READER/ INTERPRETER	READER (TO BLANKS)	USER EXIT ROUTINE	CL8	USER IDENTIFICATION

RECORD TYPE: 18 (RENAME). BUILT BY: SVC 30 ROUTINE (MODULE IGG03003).
WRITTEN BY: SAME.
LENGTH: VARIABLE (AT LEAST 144).

FIELD NAME	EXTERNAL SOURCE	SOURCE CONTROL BLOCK	CONTROL BLOCK CREATED BY	FIELD INITIALIZED BY	FIELD MAINTAINED BY	FORMAT	D E S C R I P T I O N
SMF18DTE	OS OPERATOR OR TOD CLOCK	CVT	NIP	SVC 83 ROUTINE	N/A	PL4	DATE IN PACKED DECIMAL FORM: 0CYYDDDF
SMF18FVL	VOLUME LABEL	UCB	SYSGEN	IPL/NIP	INITIATOR (IEFWD000)	CL6	FIRST VOLUME SERIAL NUMBER (SRTFVOLI)
SMF18JBN	JOB CARD	JMR	READER/ INTERPRETER	JOB STMT PROCESSOR	INITIATOR	CL8	JOB NAME
SMF18NDS	IEHPROGM CONTROL STATEMENT	RENAME PARAM. LIST	ISSUER OF SVC 30	IGG03003	N/A	CL44	NEW DSNAME
SMF18NVL	IEHPROGM CONTROL STATEMENT	RENAME VOLUME LIST	ISSUER OF SVC 30	IGG03003	N/A	BL1	NUMBER OF VOLUME SERIAL NUMBERS SPECIFIED
SMF18ODS	IEHPROGM CONTROL STATEMENT	RENAME PARAM. LIST	ISSUER OF SVC 30	IGG03003	N/A	CL44	OLD DSNAME
SMF18RSD	OS OPERATOR OR TOD CLOCK	JMR	READER/ INTERPRETER	READER	INITIATOR	PL4	JOB ENTRY DATE, PACKED FORMAT: 0CYYDDDF
SMF18RST	OS OPERATOR OR TOD CLOCK	JMR	READER/ INTERPRETER	READER	INITIATOR	BL4	JOB ENTRY TIME

FIELD NAME	EXTERNAL SOURCE	SOURCE CONTROL BLOCK	CONTROL BLOCK CREATED BY	FIELD INITIALIZED BY	FIELD MAINTAINED BY	FORMAT	D E S C R I P T I O N
SMF18SID	SMFDEFLT	SMCA	MASTER SCHEDULER	SVC 83 ROUTINE	N/A	CL4	SYSTEM IDENTIFICATION (h)
SMF18TME	OS OPERATOR OR TOD CLOCK	N/A	N/A	SVC 83 ROUTINE	N/A	BL4	TIME WHEN RECORD WAS MOVED TO OUTPUT BUFFER
SMF18UID	INSTALLATION STANDARD	JMR	READER/ INTERPRETER	READER (TO BLANKS)	USER EXIT ROUTINE	CL8	USER IDENTIFICATION

RECORD TYPE: 19 (D. A. VOLUME). BUILT BY: SVC 78 (LSPACE) ROUTINE (MODULE IGC0107H).
WRITTEN BY: INITIATOR/TERMINATOR OR MASTER SCHEDULER.
LENGTH: 64.

FIELD NAME	EXTERNAL SOURCE	SOURCE CONTROL BLOCK	CONTROL BLOCK CREATED BY	FIELD INITIALIZED BY	FIELD MAINTAINED BY	FORMAT	D E S C R I P T I O N
SMF19CUU	SYSTEM PROGRAMMER	UCB	SYSGEN	SVC 78 ROUTINE	N/A	BL2	CHANNEL AND UNIT ADDRESS
SMF19DEV	SYSTEM PROGRAMMER	UCB	SYSGEN	SVC 78 ROUTINE	N/A	BL4	DEVICE TYPE
SMF19DSR	VTOC	FORMAT 4 DSCB	DASDI	DASDI	DADSM	BL2	NUMBER OF FORMAT 0 DSCB'S (A MEASURE OF THE AMOUNT OF FREE SPACE IN THE VTOC)
SMF19DTE	OS OPERATOR OR TOD CLOCK	CVT	NIP	SVC 83 ROUTINE	N/A	PL4	DATE IN PACKED DECIMAL FORM: 0CYYDDDF
SMF19IND	2314 OR 3330 HARDWARE	NONE	N/A	SVC 78 ROUTINE	N/A	BL2	DASD MODULE ID OR DRIVE NUMBER
SMF19LEX	VTOC	FORMAT 5 DSCB	DASDI	DASDI	DADSM	BL4	NUMBER OF CYLINDERS AND TRACKS IN THE LARGEST UNALLOCATED EXTENT ON THE VOLUME
SMF19NAT	VTOC	FORMAT 4 DSCB	DASDI	DASDI	DADSM	BL2	NUMBER OF UNUSED ALTERNATE TRACKS
SMF19NDS	VTOC	FORMAT 4 DSCB	DASDI	DASDI	DADSM	BL2	NUMBER OF DSCB'S (A MEASURE OF THE SIZE OF THE VTOC)
SMF19NUE	VTOC	FORMAT 5 DSCB	DADSM	SVC 78 ROUTINE	N/A	BL2	NUMBER OF UNALLOCATED EXTENTS

FIELD NAME	EXTERNAL SOURCE	SOURCE CONTROL BLOCK	CONTROL BLOCK CREATED BY	FIELD INITIALIZED BY	FIELD MAINTAINED BY	FORMAT	D E S C R I P T I O N
SMF19OID	LABEL TRACK	VOLUME LABEL	DASDI	DASDI	DASDI	CL10	OWNER IDENTIFICATION
SMF19SID	SMFDEFLT	SMCA	MASTER SCHEDULER	SVC 83 ROUTINE	N/A	CL4	SYSTEM IDENTIFICATION (h)
SMF19SPC	VTOC	FORMAT 5 DSCB	DADSM	SVC 78 ROUTINE	N/A	BL4	TOTAL NUMBER OF UNALLOCATED CYLINDERS AND TRACKS ON THE VOLUME
SMF19TME	OS OPERATOR OR TOD CLOCK	N/A	N/A	SVC 83 ROUTINE	N/A	BL4	TIME WHEN RECORD WAS MOVED TO OUTPUT BUFFER
SMF19VOL	LABEL TRACK	VOLUME LABEL	DASDI	DASDI	DASDI	CL6	VOLUME SERIAL NUMBER
SMF19VTC	LABEL TRACK	VOLUME LABEL	DASDI	DASDI	DASDI	BL5	VTOC ADDRESS
SMF19VTI	VTOC	FORMAT 4 DSCB	DASDI	DASDI	DADSM	BL1	VTOC CONTAMINATION INDICATORS

RECORD TYPE: 20 (JOB INITIATION). BUILT BY: INITIATOR (MODULE IEFSMFIE).
WRITTEN BY: SAME.
LENGTH: VARIABLE (AT LEAST 65).

FIELD NAME	EXTERNAL SOURCE	SOURCE CONTROL BLOCK	CONTROL BLOCK CREATED BY	FIELD INITIALIZED BY	FIELD MAINTAINED BY	FORMAT	D E S C R I P T I O N
SMF20ACT	JOB CARD	JOB ACT	READER/ INTERPRETER	JOB STMT PROCESSOR	JOB STMT PROCESSOR	QC	JOB STATEMENT ACCOUNTING FIELDS (VARIABLE)
SMF20DTE	OS OPERATOR OR TOD CLOCK	CVT	NIP	SVC 83 ROUTINE	N/A	PL4	DATE IN PACKED DECIMAL FORM: 00YYDDDF
SMF20JBN	JOB CARD	JMR	READER/ INTERPRETER	JOB STMT PROCESSOR	INITIATOR	CL8	JOB NAME
SMF20NAF	JOB CARD	ACT	READER/ INTERPRETER	JOB STMT PROCESSOR	JOB STMT PROCESSOR	BL1	NUMBER OF ACCOUNTING FIELDS
SMF20PGM	JOB CARD	ACT	READER/ INTERPRETER	JOB STMT PROCESSOR	JOB STMT PROCESSOR	CL20	PROGRAMMER-NAME FIELD
SMF20RSD	OS OPERATOR OR TOD CLOCK	JMR	READER/ INTERPRETER	READER	INITIATOR	PL4	JOB ENTRY DATE, PACKED FORMAT: 00YYDDDF
SMF20RST	OS OPERATOR OR TOD CLOCK	JMR	READER/ INTERPRETER	READER	INITIATOR	BL4	JOB ENTRY TIME
SMF20SID	SMFDEFLT	SMCA	MASTER SCHEDULER	SVC 83 ROUTINE	N/A	CL4	SYSTEM IDENTIFICATION (h)
SMF20TME	OS OPERATOR OR TOD CLOCK	N/A	N/A	SVC 83 ROUTINE	N/A	BL4	TIME WHEN RECORD WAS MOVED TO OUTPUT BUFFER
SMF20UID	INSTALLATION STANDARD	JMR	READER/ INTERPRETER	READER (TO BLANKS)	USER EXIT ROUTINE	CL8	USER IDENTIFICATION

RECORD TYPE: 21 (ESV). BUILT BY: SVC 91 (VOLSTAT) ROUTINE (MODULE IGC0009A).
WRITTEN BY: SAME.
LENGTH: 48.

FIELD NAME	EXTERNAL SOURCE	SOURCE CONTROL BLOCK	CONTROL BLOCK CREATED BY	FIELD INITIALIZED BY	FIELD MAINTAINED BY	FORMAT	D E S C R I P T I O N
SMF21BLS	PROBLEM PGM, DD STATEMENT OR DS LABEL	DCB	PROBLEM PROGRAM	PROBLEM PROGRAM	OPEN	BL2	BLOCK SIZE
SMF21CA	SYSTEM PROGRAMMER	UCB	SYSGEN	IPL/NIP	IPL/NIP	BL2	CHANNEL AND UNIT ADDRESS
SMF21CLN	N/A	UCB EXTENSION	SYSGEN	IPL/NIP	IOS	BL2	NUMBER OF CLEANER ACTIONS
SMF21DEN	PROBLEM PROGRAM OR DD STATEMENT	DEB	OPEN	IGC0009A	N/A	BL1	TAPE MODE-SET COMMAND: SETS DENSITY, PARITY, TRANSLATE AND DATA CONVERSION IF APPLICABLE
SMF21DTE	OS OPERATOR OR TOD CLOCK	CVT	NIP	SVC 83 ROUTINE	N/A	PL4	DATE IN PACKED DECIMAL FORM: 00YYDDDF
SMF21ERG	N/A	UCB EXTENSION	SYSGEN	IPL/NIP	IOS	BL2	NUMBER OF ERASE GAPS
SMF21NB	N/A	UCB EXTENSION	SYSGEN	IPL/NIP	IOS	BL1	NUMBER OF NOISE BLOCKS
SMF21PR	N/A	UCB EXTENSION	SYSGEN	IPL/NIP	IOS	BL1	NUMBER OF PERMANENT READ ERRORS
SMF21PW	N/A	UCB EXTENSION	SYSGEN	IPL/NIP	IOS	BL1	NUMBER OF PERMANENT WRITE ERRORS

FIELD NAME	EXTERNAL SOURCE	SOURCE CONTROL BLOCK	CONTROL BLOCK CREATED BY	FIELD INITIALIZED BY	FIELD MAINTAINED BY	FORMAT	D E S C R I P T I O N
SMF21SID	SMFDEFLT	SMCA	MASTER SCHEDULER	SVC 83 ROUTINE	N/A	CL4	SYSTEM IDENTIFICATION (h)
SMF21SIO	N/A	UCB EXTENSION	SYSGEN	IPL/NIP	IOS	BL2	NUMBER OF START I/O'S
SMF21TME	OS OPERATOR OR TOD CLOCK	N/A	N/A	SVC 83 ROUTINE	N/A	BL4	TIME WHEN RECORD WAS MOVED TO OUTPUT BUFFER
SMF21TR	N/A	UCB EXTENSION	SYSGEN	IPL/NIP	IOS	BL1	NUMBER OF TEMPORARY READ ERRORS
SMF21TW	N/A	UCB EXTENSION	SYSGEN	IPL/NIP	IOS	BL1	NUMBER OF TEMPORARY WRITE ERRORS
SMF21UCB	SYSTEM PROGRAMMER	UCB	SYSGEN	IPL/NIP	IPL/NIP	BL4	UCB DEVICE TYPE CODE
SMF21VOL	VOLUME LABEL	UCB	SYSGEN	IPL/NIP	INITIATOR (IEFWD000)	CL6	VOLUME SERIAL NUMBER

Notes on the Table of Symbolic Names in SMF

(a) Most of the data elements supplied by SMF are precisely defined by the table in Appendix D. Some, however, require more discussion.

In the Type 4 record, the SMF4PRTY field gives the dispatching priority of the job step TCB at the time the step is terminated. This is an unsigned, 8-bit binary number in the range 0 to 255. It is not necessarily the same as the limit priority, though it will never exceed this value. The limit priority specified by the DPRTY parameter on the EXEC statement is not available from the basic SMF records.

(b) The step number, SMF4STN, refers to the ordinal position of the step currently being terminated within the sequence of steps constituting the job. It includes steps cancelled during allocation, steps bypassed because of condition codes and steps not executed because a previous step terminated abnormally. Such steps are tallied as they are processed in "flush mode" by the Initiator/Terminator; they are characterized by having zero CPU time. It should be noted, however, that if a job is restarted at a step other than the first, by using the "RESTART=" parameter on the JOB statement, the numbering of steps begins at the one where restart occurs, not at the first EXEC statement physically present in the JCL for the job.

(c) The amount of main storage allocated (SMF4RSHO) is a two-byte binary number giving the size in kilobytes of that portion of the region contained in high-speed memory. It is immediately followed in the record by another two-byte field giving the size in kilobytes of that portion of the region contained in LCS (bulk core, or hierarchy- 1, storage). These values will be those specified in the REGION parameter on the EXEC or JOB card, unless they are less than the MINPART parameter supplied by the system programmer; in that case, the MINPART values are used.

(d) Similarly, the four-byte field SMF4HOST, which gives the number of kilobytes of high-speed memory actually used, is followed immediately by another four-byte field giving the number of kilobytes of LCS actually used. Neither of the hierarchy-1 fields has a symbolic name supplied by the IFASMFR macroinstruction.

(e) The precise meaning of the SMF4HOST field (and its hierarchy-1 counterpart) is best explained in terms of its derivation. Main storage within a region is allocated both from the bottom up and the top down, depending on the type of request. The Main Storage Supervisor maintains, in the Timing Control Table, a low-water mark for the low end of the region (the highest address currently allocated there) and a similar high-water mark for the high end of the region. It also maintains the minimum difference between the high- and low-water marks. The value of SMF4HOST is the difference between the

region size and this minimum unused storage amount at the end of the job step. The entries in the TCT are updated only when a 2K block of main storage is allocated or freed, so the SMF4HOST value has an inherent imprecision of two kilobytes. Where the value of SMF4HOST is lower than the region size, the job step could generally have run successfully in the SMF4HOST figure, allowing about 10% more space for environmental variability. There are, however, programs such as Sort/Merge and some compilers that use GETMAIN to measure the region size and consequently are recorded by SMF as using all available space. Such programs will, in fact, run in smaller regions.

(f) The precise meaning of the SMF4SCC (step completion code) field depends on a careful analysis of the CPU time (SMF4SETM) fields for all steps in the job, together with the job termination indicator byte (SMF5JBTI). Even then it may be ambiguous.

(g) The OS SMF manual for release 21.7 (<2>) states that for normal completion SMF4SCC contains the two low-order bytes of register 15 at termination; for flushed steps, it contains zeros; and for steps that are abnormally terminated, it has the form 'xyyy' where 'yyy' is the ABEND code and 'x' is 0 or 8 depending on whether there was a system or user ABEND. The difference between a flushed step and one that completed normally with a return code of zero is that only the flushed step will have SMF4SETM equal to zero. The SMF manual further states that abnormal or normal termination can be determined from the job termination indicator, but this is not always the case. For example, in a multi-step job with at least two nonzero completion codes, a value of 4 for SMF4SCC could mean either a system ABEND code of 004 or normal completion with a return code of 4. This possibility arises from the fact that abnormal termination of a job step does not necessarily preclude the execution of subsequent steps.

(h) The system identification field which occurs in the standard header portion of each record (symbolic names SMF0SID, SMF1SID, SMF2SID, etc.) is made up of the SID and MDL parameters taken from member SMFDEFLT of SYS1.PARMLIB, possibly overridden by operator intervention at IPL time. These two two-byte parameters are concatenated to form one four-byte field, e.g., SID=SF and MDL=65 would result in an SMF0SID value of 'SF65'.

(i) System wait time appears in two SMF data elements: SMF1SWT and SMF12SWT. Basically, the time during which the CPU is in the wait state is accumulated and added into a total figure every ten minutes. Whenever a job step terminates and the total wait time is nonzero, a Type 1 record containing that total as its SMF1SWT field is written and the total is reset to zero. Note that the ten-minute accumulator is not reset until the current interval expires; then it is added into the (zeroed) total. Thus SMF1SWT normally contains the amount of CPU wait time accumulated during one or more ten-minute intervals.

(j) In the first Type 1 record written after an IPL (i.e. the first Type 1 immediately following a Type 0 record), the value of SMF1SWT is the amount of CPU wait time accumulated during the IPL process. More precisely, this is the wait time accrued between the time the dispatcher becomes usable, i.e. when the nucleus has been initialized, and the time the SMF initialization routine is run. Before the dispatcher is usable, the CPU will not enter the wait state, because the IPL and NIP routines check for completion of I/O operations by repeatedly testing rather than by waiting for an interruption. Thus the first SMF1SWT value following IPL reflects the wait time accumulated during Master Scheduler initialization.

(k) When a HALT or SWITCH command is issued by the operator, the wait accumulated during the current ten-minute interval is added into the total since the last Type 1 record, and the resulting value is placed in the SMF12SWT field of the Type 12 record.

(l) The SMFDSSNO field in the Type 14 or 15 record depends on the medium used for the data set described by the record. For tape data sets, it is the volume serial number of the first reel on which the data set resides. For data sets on DASD, it consists of a six-byte segment of the UCB, which includes a one-byte RESERVE count for shared DASD, a device reservation indicator byte, the two-byte address of the RQE for AVR volume serial number verification, a byte of volume serial number verification flags, and the high-order byte of the address of the DEB for the first user on the queue for the device.

(m) The SMFTIOE2 field in the Type 14/15 record is useful in analyzing tape mounts. In this one-byte field, bit 6 (X'02') is set to indicate that the current volume of a tape data set is to be rewound and unloaded when end-of-volume occurs or the data set is closed. If this bit is off, the volume will still be unloaded whenever the number of volumes exceeds the number of units allocated to the data set. Otherwise, the tape remains mounted and any data set on it can be opened without operator intervention.

APPENDIX E: BASIC SMF RECORD LAYOUTS WITH OFFSETS

26 SEP 75

```
LOC  OBJECT CODE    ADDR1 ADDR2  STMT   SOURCE STATEMENT

                                    2 ****************************************************************
                                    3 *                                                              *
                                    4 *  WARNING--THE FOLLOWING RECORD LAYOUTS CONTAIN FIELDS WHICH  *
                                    5 *  ARE DEFINED ONLY FOR VS OPERATING SYSTEMS.  THESE FIELDS    *
                                    6 *  ARE OMITTED FROM APPENDIX D.  APPENDIX D SHOULD BE TAKEN    *
                                    7 *  AS THE PRINCIPAL SOURCE OF DETAILED INFORMATION ON THE SMF  *
                                    8 *  DATA ELEMENTS, AND THIS APPENDIX SHOULD BE USED ONLY AS A   *
                                    9 *  SUPPLEMENT TO IT, PROVIDING HEXADECIMAL OFFSETS, A CROSS-   *
                                   10 *  REFERENCE AND THE RELATIVE LOCATIONS OF THE FIELDS WITH-    *
                                   11 *  IN THE RECORDS.                                             *
                                   12 *                                                              *
                                   13 ****************************************************************

000000                             15 SMFRECS  CSECT

                                   35+*  THIS DSECT DEFINES THE FORMAT FOR RECORD TYPE 0 (IPL RECORD)
                                   37+*
000000                             38+         DS    0F          ALIGN TO FULL WORD BOUNDARY
                          00000    39+SMFRCD0  EQU   *           HEADER SEGMENT
000000                             40+SMF0LEN  DS    BL2'0'      RECORD LENGTH
000002                             41+SMF0SEG  DS    BL2'0'      SEGMENT DESCRIPTOR
000004 00                          42+SMF0FLG  DC    BL1'0'      HEADER FLAG BYTE
000005 00                          43+SMF0RTY  DC    BL1'0'      RECORD TYPE 0
000006 00000000                    44+SMF0TME  DC    4BL1'0'     TOD, USING FORMAT FROM TIME MACRO W/BIN. INTVL
00000A 0000000C                    45+SMF0DTE  DC    PL4'0000'         DATE IN PACKED DECIMAL FORM: 00YY00DF
                                   46+*                                ( F IS A SIGN)
00000E 40404040                    47+SMF0SID  DC    4CL1' '     SYSTEM IDENTIFICATION                    XL03034
000012 00000000                    48+SMF0JWT  DC    4BL1'0'     JOB WAIT TIME DEFAULT VALUE IN MINUTES(BINARY)
000016 00000000                    49+SMF0BUF  DC    4BL1'0'     SMF BUFFER SIZE IN BYTES(BINARY)
00001A 00000000                    50+SMF0VST  DC    4BL1'0'     VIRTUAL STORAGE SIZE IN K BLOCKS(BINARY) M4432
00001E 00                          51+SMF0OPT  DC    BL1'0'      OPTIONS BYTE:
                                   52+*                BIT0 - SYSTEM & JOB DATA TO BE COLLECTED
                                   53+*                BIT1 - SYSTEM, JOB, & STEP DATA TO BE COLLECTED
                                   54+*                BIT2 - EXITS REQUESTED
                                   55+*                BIT3 - DATA SET ACCOUNTING
                                   56+*                BIT4 - VOLUME ACCOUNTING
                                   57+*                BIT5 - RESERVED(WAS ESV IN 14 AND 15)
                                   58+*                BIT6 - TEMPORARY-DATA-SET SCRATCH RECORDS
                                   59+*                BIT7 - =0 BACKGROUND JOB; =1 FOREGROUND JOB
00001F 00000000                    60+SMF0RST  DC    4BL1'0'     REAL STORAGE SIZE IN K BLOCKS(BINARY)    M4432
                                   61 *
                                   62 *  NOTE THAT FOR DS/MVT WITHOUT VIRTUAL STORAGE, THE REAL STORAGE SIZE
                                   63 *       IS PLACED IN THE SMF0VST FIELD AT OFFSET X'1A', AND THE RECORD
                                   64 *       LENGTH (INCLUDING THE SEGMENT DESCRIPTOR WORD) IS 31 BYTES
                                   65 *       RATHER THAN 35 BYTES.
```

```
 LOC  OBJECT CODE    ADDR1 ADDR2  STMT   SOURCE STATEMENT

                                   80+*
                                   81+* THIS RECORD IS WRITTEN AFTER EVERY IPL AND AT THE FIRST JOB/STEP
                                   82+* TERMINATION AFTER THE EXPIRATION OF ONE 10 MINUTE INTERVAL.
                                   84+*
000000                             85+          DS    0F         ALIGN TO FULL WORD BOUNDARY
                            00000  86+SMFRCD1   EQU   *          HEADER SEGMENT
000000                             87+SMF1LEN   DS    BL2'0'     RECORD LENGTH
000002                             88+SMF1SEG   DS    BL2'0'     SEGMENT DESCRIPTOR
000004 00                          89+SMF1FLG   DC    BL1'0'     HEADER FLAG BYTE
000005 00                          90+SMF1RTY   DC    BL1'0'     RECORD TYPE 1
000006 00000000                    91+SMF1TME   DC    4BL1'0'    TOD, USING FORMAT FROM TIME MACRO W/BIN. INTVL
00000A 0000000C                    92+SMF1DTE   DC    PL4'0000'        DATE IN PACKED DECIMAL FORM: 00YYDDDF
                                   93+*                                ( F IS A SIGN)
00000E 40404040                    94+SMF1SID   DC    4CL1' '    SYSTEM IDENTIFICATION                    XL03034
000012 00000000                    95+SMF1SWT   DC    4BL1'0'          SYSTEM WAIT TIME IN HUNDREDTHS OF SECONDS
                                   96+*                                COLLECTED DURING EXPIRED 10 MINUTE
                                   97+*                                INTERVALS SINCE LAST WAIT TIME RECORD
000016 00000000                    98+SMF1TEXP  DC    4BL1'0'    COLLECTION END TIME
00001A 00000000                    99+SMF1PGIN  DC    4BL1'0'    TOTAL PAGE-INS
00001E 00000000                   100+SMF1PGOT  DC    4BL1'0'    TOTAL PAGE-OUTS
000022 00000000                   101+SMF1PGRL  DC    4BL1'0'    TOTAL PAGES RECLAIMED
```

```
 LOC  OBJECT CODE    ADDR1 ADDR2  STMT   SOURCE STATEMENT

                                  121+* THIS RECORD IS WRITTEN BY THE SMF DUMP UTILITY PROGRAM AT THE
                                  122+* BEGINNING OF A DUMP DATA SET.
                                  123+*
000000                            124+          DS    0F        ALIGN TO FULL WORD BOUNDARY
                            00000 125+SMFRCD2   EQU   *         HEADER SEGMENT
000000                            126+SMF2LEN   DS    BL2'0'    RECORD LENGTH
000002                            127+SMF2SEG   DS    BL2'0'    SEGMENT DESCRIPTOR
000004 00                         128+SMF2FLG   DC    BL1'0'    HEADER FLAG BYTE
000005 00                         129+SMF2RTY   DC    BL1'0'    RECORD TYPE 2
000006 00000000                   130+SMF2TME   DC    4BL1'0'   TOD, USING FORMAT FROM TIME MACRO W/BIN. INTVL
00000A 0000000C                   131+SMF2DTE   DC    PL4'0000'       DATE IN PACKED DECIMAL FORM: 00YYDDDF
                                  132+*                           ( F IS A SIGN)
00000E 40404040                   133+SMF2SID   DC    4CL1' '   SYSTEM IDENTIFICATION                    XL03034
```

```
LOC  OBJECT CODE    ADDR1 ADDR2  STMT   SOURCE STATEMENT

                                        153+* THIS RECORD  IS WRITTEN BY THE SMF DUMP UTILITY PROGRAM AT THE END
                                        154+* OF A DUMP DATA SET.
                                        155+*
000000                                  156+          DS    0F        ALIGN TO FULL WORD BOUNDARY
                                00000   157+SMFRCD3   EQU   *         HEADER SEGMENT
000000                                  158+SMF3LEN   DS    BL2'0'    RECORD LENGTH
000002                                  159+SMF3SEG   DS    BL2'0'    SEGMENT DESCRIPTOR
000004 00                               160+SMF3FLG   DC    BL1'0'    HEADER FLAG BYTE
000005 00                               161+SMF3RTY   DC    BL1'0'    RECORD TYPE 3
000006 00000000                         162+SMF3TME   DC    4BL1'0'   TOD, USING FORMAT FROM TIME MACRO W/BIN. INTVL
00000A 0000000C                         163+SMF3DTE   DC    PL4'0000'       DATE IN PACKED DECIMAL FORM: 00YYDDDF
00000E 40404040                         164+SMF3SID   DC    4CL1' '   SYSTEM IDENTIFICATION                    XL03034
```

```
 LOC  OBJECT CODE    ADDR1 ADDR2  STMT   SOURCE STATEMENT

                                  184+* THIS RECORD IS WRITTEN AT NORMAL OR ABNORMAL TERMINATION OF A JOB
                                  185+* STEP.RECORD LENGTH IS VARIABLE.
                                  186+*
000000                            187+          OS    0F        ALIGN TO FULL WORD BOUNDARY
                            00000 188+SMFRCD4   EQU   *         HEADER SEGMENT
000000                            189+SMF4LEN   DS    BL2'0'    RECORD LENGTH
000002                            190+SMF4SEG   DS    BL2'0'    SEGMENT DESCRIPTOR
000004 00                         191+SMF4FLG   DC    BL1'0'    HEADER FLAG BYTE
000005 00                         192+SMF4RTY   DC    BL1'0'    RECORD TYPE 4
000006 00000000                   193+SMF4TME   DC    4BL1'0'   TOD USING FORMAT FROM TIME MACRO W/BIN. INTVL.
00000A 0000000C                   194+SMF4DTE   DC    PL4'0000' DATE IN PACKED DECIMAL FORM: 00Y4DDDF
00000E 40404040                   195+SMF4SID   DC    4CL1' '   SYSTEM IDENTIFICATION                    XL03034
000012 4040404040404040           196+SMF4JBN   DC    8CL1' '   JOB NAME
00001A 00000000                   197+SMF4RST   DC    4BL1'0'   READER START TIME (IN 100THS SECONDS)
00001E 0000000C                   198+SMF4ROS   DC    PL4'0000' READER START DATE
000022 4040404040404040           199+SMF4UIF   DC    8CL1' '   USER IDENTIFICATION FIELD
00002A 00                         200+SMF4STN   DC    BL1'0'    STEP NUMBER
00002B 00000000                   201+SMF4SIT   DC    4BL1'0'   STEP INITIATION TIME (IN 100THS SECONDS)
00002F 0000000C                   202+SMF4STID  DC    PL4'0000' STEP INITIATION DATE (PACKED DECIMAL FORMAT)
000033 00000000                   203+SMF4NCI   DC    4BL1'0'   NUMBER OF CARD IMAGES IN DD DATA OR DD  *
                                  204+*                         DATA SETS
000037 0000                       205+SMF4SCC   DC    2BL1'0'   STEP COMPLETION CODE
000039 00                         206+SMF4PRTY  DC    BL1'0'    PRIORITY AT WHICH STEP WAS DISPATCHED :
                                  207+*                         ACTUAL PRIORITY=251-(15-USER PRIORITY)*16
00003A 4040404040404040           208+SMF4PGMN  DC    CL8' '    PROGRAM NAME
000042 4040404040404040           209+SMF4STMN  DC    CL8' '    STEP NAME
00004A 0000                       210+SMF4RSHO  DC    2BL1'0'   REGION SIZE IN 1K BLKS REQUESTED
00004C 0000                       211+          DC    2BL1'0'   RESERVED
00004E 0000                       212+SMF4HOST  DC    2BL1'0'   STORAGE USED(VIRT OR REAL-SEE SMF4RIN)
000050 000000000000               213+          DC    6BL1'0'   RESERVED
000056 00                         214+SMF4SPK   DC    1BL1'0'   STORAGE PROTECT KEY                      A40791
000057 00                         215+SMF4STI   DC    1BL1'0'   STEP TERMINATION INDICATOR
                                  216+*                          BIT0 - RESERVED
                                  217+*                          BIT1 - CANCELLED AT EXIT IEFUJV
                                  218+*                          BIT2 - CANCELLED AT EXIT IEFUJI
                                  219+*                          BIT3 - CANCELLED AT EXIT IEFUSI
                                  220+*                          BIT4 - CANCELLED AT EXIT IEFACTRT
                                  221+*                          BIT5 - RESERVED
                                  222+*                          BIT6 - 0=NORMAL COMPLETION
                                  223+*                                 1=ABEND
                                  224+*                          BIT7 - 1=STEP FLUSHED
000058 0000                       225+          DC    2BL1'0'     RESERVED                               A40791
00005A 00000000                   226+SMF4AST   DC    4BL1'0'     DEVICE ALLOC START TIME                A40791
00005E 00000000                   227+SMF4PPST  DC    4BL1'0'     PROBLEM PROGRAM START TIME             A40791
000062 00000000                   228+          DC    4BL1'0'     RESERVED
000066 0000                       229+SMF4RIN   DC    2BL1'0'     RECORD INDICATORS
                                  230+*                            BIT7 =0 STORAGE IS VIRTUAL
                                  231+*                                 =1 STORAGE IS REAL
000068 0000                       232+SMF4RLCT  DC    2BL1'0'     OFFSET TO RELOCATE SECTION
00006A 0000                       233+SMF4LENN  DC    2BL1'0'     LENGTH OF EXCP PORTION OF RECORD
```

```
LOC    OBJECT CODE     ADDR1 ADDR2  STMT   SOURCE STATEMENT

                                    234+*          FOR EACH DEVICE ASSIGNED TO EACH DATA SET THERE IS AN 8 BYTE
                                    235+*          ENTRY HAVING THE FOLLOWING FORMAT:
                                    236+*
00006C 00                           237+SMF4DEVC DC     BL1'0'     DEVICE CLASS
00006D 00                           238+SMF4UTYP DC     BL1'0'     UNIT TYPE
00006E 0000                         239+SMF4CUAD DC     2BL1'0'    CHANNEL AND UNIT ADDRESS
000070 00000000                     240+SMF4EXCP DC     4BL1'0'    COUNT OF EXCP'S ISSUED FOR THE DEVICE AND
                                    241+*                          DATA SET
000074                       0006C  242+           ORG    SMF4DEVC
00006C 00                           243+SMF4LNTH DC     BL1'0'     TOTAL LENGTH OF CPU AND ACCT. SECTION
00006D 000000                       244+SMF4SETM DC     3BL1'0'    STEP EXECUTION TIME (IN 100THS SECONDS)
000070 00                           245+SMF4NAF  DC     BL1'0'     NUMBER OF ACCOUNTING FIELDS
000071                              246+SMF4ACTF DS     0C         EXEC STATEMENT ACCOUNTING FIELDS(VARIABLE)
                                    247+*
                                    248+*          EACH ENTRY FOR AN ACCOUNTING FIELD CONTAINS THE LENGTH OF THE
                                    249+*          FIELD (1 BYTE,BINARY), FOLLOWED BY THE FIELD(EBCDIC).AN
                                    250+*          OMITTED FIELD IS REPRESENTED BY A LENGTH INDICATOR OF 0.
                                    251+*
                                    252+*          RELOCATE SECTION
000071                              253+SMF4PGIN DS     4BL1'0'    NUMBER OF PAGE-INS
000075                              254+SMF4PGDT DS     4BL1'0'    NUMBER OF PAGE-OUTS
```

```
 LOC  OBJECT CODE    ADDR1 ADDR2  STMT   SOURCE STATEMENT

                                  274+* THIS RECORD IS WRITTEN AT  NORMAL OR ABNORMAL TERMINATION OF A JOB.
                                  275+* RECORD LENGTH IS VARIABLE.
                                  276+*
000000                            277+          DS    0F        ALIGN TO FULL WORD BOUNDARY
                           00000  278+SMFRCD5   EQU   *         HEADER SEGMENT
000000                            279+SMF5LEN   DS    BL2'0'    RECORD LENGTH
000002                            280+SMF5SEG   DS    BL2'0'    SEGMENT DESCRIPTOR
000004 00                         281+SMF5FLG   DC    BL1'0'    HEADER FLAG BYTE
000005 00                         282+SMF5RTY   DC    BL1'0'    RECORD TYPE 5
000006 00000000                   283+SMF5TMF   DC    4BL1'0'   TOD USING FORMAT FROM TIME MACRO W/BIN. INTVL.
00000A 0000000C                   284+SMF5DTE   DC    PL4'0000' DATE, PACKED DEC. FORM : 00YYDDDF(F IS A SIGN)
00000E 40404040                   285+SMF5SID   DC    4CL1' '   SYSTEM IDENTIFICATION                    XL03034
000012 4040404040404040           286+SMF5JBN   DC    8CL1' '   JOB NAME
00001A 00000000                   287+SMF5RST   DC    4BL1'0'   READER START TIME FOR JOB (IN 100THS/SECONDS)
00001E 0000000C                   288+SMF5RSD   DC    PL4'0000' READER START DATE,PACKED DECIMAL FORMAT
000022 4040404040404040           289+SMF5UIF   DC    8CL1' '   USER IDENTIFICATION FIELD
00002A 00                         290+SMF5NST   DC    BL1'0'    NUMBER OF STEPS IN THE JOB
00002B 00000000                   291+SMF5JIT   DC    4BL1'0'   JOB INITIATION TIME (IN 100THS/SECONDS)
00002F 0000000C                   292+SMF5JID   DC    PL4'0000' JOB INITIATION DATE, PACKED DECIMAL FORMAT
000033 00000000                   293+SMF5NCI   DC    4BL1'0'   NUMBER OF CARD-IMAGE RECORDS IN DD DATA OR DD*
                                  294+*                         DATA SETS
000037 0000                       295+SMF5JCC   DC    2BL1'0'   JOB COMPLETION CODE
000039 00                         296+SMF5JPTY  DC    BL1'0'    JOB PRIORITY
00003A 00000000                   297+SMF5RSTT  DC    4BL1'0'   READER STOP TIME FOR JOB (IN 100THS/SECONDS)
00003E 0000000C                   298+SMF5RSTD  DC    PL4'0000' READER STOP DATE FOR JOB (IN PACKED DEC.FORM)
000042 00                         299+SMF5JBTI  DC    BL1'0'    JOB TERMINATION INDICATOR
                                  300+*                         BIT0-RESERVED
                                  301+*                            1-CANCELLED AT EXIT IEFUJV
                                  302+*                            2-CANCELLED AT EXIT IEFUJI
                                  303+*                            3-CANCELLED AT EXIT IEFUSI
                                  304+*                            4-CANCELLED AT EXIT IEFACTRT
                                  305+*                            5-RESERVED
                                  306+*                            6-0=NORMAL COMPLETION
                                  307+*                              1=ABEND
                                  308+*                            7-RESERVED
000043 0000000000                 309+SMF5SMCI  DC    5BL1'0'   SYSOUT CLASSES AND MSGCLASS INDICATOR
000048 00                         310+          DC    BL1'0'    RESERVED
000049 00                         311+SMF5RDCL  DC    BL1'0'    READER DEVICE CLASS
00004A 00                         312+SMF5RUTY  DC    BL1'0'    READER UNIT TYPE
00004B 40                         313+SMF5JICL  DC    CL1' '    JOB INPUT CLASS
00004C 00                         314+SMF5SPK   DC    1BL1'0'   STORAGE PROTECT KEY                      A40791
00004D 000000                     315+          DC    3BL1'0'   RESERVED                                 X02992
000050 4040404040404040           316+SMF5QID   DC    8CL1' '   RES USER IDENTIFICATION                  XM34563
000058 0000000000000000           317+          DC    8BL1'0'   RESERVED                                 XM34563
000060 00                         318+SMF5TLEN  DC    BL1'0'    TOTAL LENGTH OF REST OF RECORD
000061 4040404040404040           319+SMF5PRGN  DC    20CL1' '  PROGRAMMER'S NAME
000075 000000                     320+SMF5JCPU  DC    3BL1'0'   JOB CPU TIME (IN 100THS/SECONDS)
000078 00                         321+SMF5ACTF  DC    BL1'0'    NUMBER OF ACCOUNTING FIELDS
000079                            322+SMF5JSAF  DS    0C        JOB STATEMENT ACCOUNTING FIELDS(VARIABLE)
                                  323+*
                                  324+*         EACH ENTRY FOR AN ACCOUNTING FIELD CONTAINS THE LENGTH OF THE
                                  325+*         FIELD(1 BYTE,BINARY),FOLLOWED BY THE FIELD (EBCDIC).AN OMITTED
                                  326+*         FIELD IS REPRESENTED BY A LENGTH INDICATOR OF 0.
```

```
LOC    OBJECT CODE     ADDR1 ADDR2  STMT   SOURCE STATEMENT

                                    346+* THIS RECORD IS WRITTEN WHEN PROCESSING OF A SYSOUT CLASS IS COMPLETED
                                    347+* FOR A JOB,AND WHEN THE FORM CHANGES WITHIN A CLASS.
                                    349+*
000000                              350+          DS    0F          ALIGN TO FULL WORD BOUNDARY
                              00000 351+SMFRCD6   EQU   *           HEADER SEGMENT
000000                              352+SMF6LEN   DS    BL2'0'      RECORD LENGTH
000002                              353+SMF6SEG   DS    BL2'0'      SEGMENT DESCRIPTOR
000004 00                           354+SMF6FLG   DC    BL1'0'      HEADER FLAG BYTE
000005 00                           355+SMF6RTY   DC    BL1'0'      RECORD TYPE 6
000006 00000000                     356+SMF6TME   DC    4BL1'0'     TOD, USING FORMAT FROM TIME MACRO W/BIN. INTVL
00000A 0000000C                     357+SMF6DTE   DC    PL4'0000'         DATE IN PACKED DECIMAL FORM: 00YYDDDF
                                    358+*                           ( F IS A SIGN)
00000E 40404040                     359+SMF6SID   DC    4CL1' '     SYSTEM IDENTIFICATION                    XL03034
000012 4040404040404040             360+SMF6JBN   DC    8CL1' '     JOB NAME
00001A 00000000                     361+SMF6RST   DC    4BL1'0'     READER START TIME
00001F 0000000C                     362+SMF6RSD   DC    PL4'0000'   READER START DATE ; 00YYDDDF
000022 4040404040404040             363+SMF6UIF   DC    CL8' '      USER ID FIELD
00002A 40                           364+SMF6OWC   DC    CL1' '      OUTPUT WRITER CLASS
00002B 00000000                     365+SMF6WST   DC    4BL1'0'     WRITER START TIME
00002F 0000000C                     366+SMF6WSD   DC    PL4'0000'   WRITER START DATE
000033 00000000                     367+SMF6NLR   DC    4BL1'0'     NUMBER OF LOGICAL RECORDS FOR THE WRITER
                                    368+*                           PER FORM NUMBER PER CLASS
000037 00                           369+SMF6IOE   DC    BL1'0'      IO ERROR INDICATOR:
                                    370+*                           BITS 0-3 RESERVED
                                    371+*                           4 - I/O DISCONTINUED(REMOTE OUTPUT ONLY)
                                    372+*                           5 - INPUT ERROR
                                    373+*                           6 - OUTPUT ERROR
                                    374+*                           7 - INPUT ERROR ON SYS1.SYSJOBQE
000038 00                           375+SMF6NDS   DC    BL1'0'      NUMBER OF DATA SETS PROCESSED  BY THE OUTPUT
                                    376+*                           WRITER FOR THE JOB
000039 40404040                     377+SMF6FMN   DC    4CL1' '     FORM NUMBER
00003D 4040404040404040             378+SMF6QID   DC    8CL1' '     RES USER IDENTIFICATION                  XM34563
```

```
 LOC  OBJECT CODE    ADDR1 ADDR2  STMT   SOURCE STATEMENT

                                  398+* THIS RECORD IS WRITTEN WHEN AN SMF DATA SET BECOMES AVAILABLE
                                  399+* FOLLOWING THE UNAVAILABILITY OF AN SMF DATA SET.
                                  401+*
000000                            402+          DS    0F         ALIGN TO FULL WORD BOUNDARY
                            00000 403+SMFRCD7   EQU   *          HEADER SEGMENT
000000                            404+SMF7LEN   DS    BL2'0'     RECORD LENGTH
000002                            405+SMF7SEG   DS    BL2'0'     SEGMENT DESCRIPTOR
000004 00                         406+SMF7FLG   DC    BL1'0'     HEADER FLAG BYTE
000005 00                         407+SMF7RTY   DC    BL1'0'     RECORD TYPE 7
000006 00000000                   408+SMF7TME   DC    4BL1'0'    TOD, USING FORMAT FROM TIME MACRO W/BIN. INTVL
00000A 0000000C                   409+SMF7DTF   DC    PL4'0000'        DATE IN PACKED DECIMAL FORM: 00YYDDDF
                                  410+*                              ( F IS A SIGN)
00000E 40404040                   411+SMF7SID   DC    4CL1' '    SYSTEM IDENTIFICATION                    XL03034
000012 0000                       412+SMF7NRO   DC    2BL1'0'  NUMBER OF SMF RECORDS OMITTED FROM DATA SET
000014 00000000                   413+SMF7STM   DC    4BL1'0'  STARTING TIME IN HUNDREDTHS OF SECONDS AT
                                  414+*                        WHICH NO DATA SET WAS AVAILABLE FOR RECORDING
                                  415+*                        SMF RECORDS
000018 0000000C                   416+SMF7STD   DC    PL4'0000' STARTING DATE AT WHICH NO DATA SET WAS
                                  417+*                        AVAILABLE FOR RECORDING SMF RECORDS
```

```
         APPENDIX E:  BASIC SMF RECORD LAYOUTS WITH OFFSETS

   LOC  OBJECT CODE    ADDR1 ADDR2  STMT   SOURCE STATEMENT

                                     437+* THIS RECORD IS WRITTEN AFTER COMPLETION OF IPL, FOLLOWING THE SET
                                     438+* DATE COMMAND. RECORD LENGTH IS VARIABLE.
                                     439+*
000000                               440+         DS    0F        ALIGN TO FULL WORD BOUNDARY
                                00000 441+SMFRCD8  EQU   *         HEADER SEGMENT
000000                               442+SMF8LEN  DS    BL2'0'    RECORD LENGTH
000002                               443+SMF8SEG  DS    BL2'0'    SEGMENT DESCRIPTOR
000004 00                            444+SMF8FLG  DC    BL1'0'    HEADER FLAG BYTE
000005 00                            445+SMF8RTY  DC    BL1'0'    RECORD TYPE 8
000006 00000000                      446+SMF8TME  DC    4BL1'0'   TOD USING FORMAT FROM TIME MACRO W/BIN. INTVL.
00000A 0000000C                      447+SMF8DTE  DC    PL4'0000' DATE IN PACKED DEC.FORMAT: 00YYDDDF
00000E 40404040                      448+SMF8SID  DC    4CL1' '   SYSTEM IDENTIFICATION                    XL03034
000012 0000                          449+SMF8LENN DC    2BL1'0'   LENGTH OF FIELD
                                     450+*        FOR EACH DEVICE ONLINE THERE IS A FOUR-BYTE ENTRY HAVING THE
                                     451+*        FOLLOWING FORMAT:
000014                               452+SMF8IODV DS    0C        RESERVE FOUR BYTE AREA FOR EACH DEVICE
                                     453+*                        BYTE 0 - DEVICE CLASS
                                     454+*                             1 - UNIT TYPE
                                     455+*                             2 - CHANNEL ADDRESS
                                     456+*                             3 - UNIT ADDRESS
```

```
 LOC  OBJECT CODE    ADDR1 ADDR2  STMT   SOURCE STATEMENT

                                  476+* THIS RECORD IS WRITTEN WHEN A VARY ONLINE COMMAND IS PROCESSED.
                                  477+* RECORD LENGTH IS VARIABLE.
                                  478+*
000000                            479+         DS    0F        ALIGN TO FULL WORD BOUNDARY
                            00000 480+SMFRCD9  EQU   *         HEADER SEGMENT
000000                            481+SMF9LEN  DS    BL2'0'    RECORD LENGTH
000002                            482+SMF9SEG  DS    BL2'0'    SEGMENT DESCRIPTOR
000004 00                         483+SMF9FLG  DC    BL1'0'    HEADER FLAG BYTE
000005 00                         484+SMF9RTY  DC    BL1'0'    RECORD TYPE 9
000006 00000000                   485+SMF9TME  DC    4BL1'0'   TOD USING FORMAT FROM TIME MACRO W/BIN. INTVL.
00000A 0000000C                   486+SMF9DTE  DC    PL4'0000' DATE IN PACKED DEC. FORM: 00YYDDDF
00000E 40404040                   487+SMF9SID  DC    4CL1' '   SYSTEM IDENTIFICATION                    XL03034
000012 0000                       488+SMF9LENN DC    2BL1'0'   LENGTH OF THE FIELD
000014                            489+SMF9DVAD DS    0C        DEVICES BEING ADDED TO THE I/O CONFIGURATION
                                  490+*                        (VARIABLE,BINARY)
                                  491+*        FOR EACH DEVICE BEING ADDED THERE IS A FOUR BYTE ENTRY HAVING
                                  492+*        THE FOLLOWING FORMAT:
                                  493+*                        BYTE 0-DEVICE CLASS
                                  494+*                             1-UNIT TYPE
                                  495+*                             2-CHANNEL ADDRESS
                                  496+*                             3-UNIT ADDRESS
```

```
LOC  OBJECT CODE     ADDR1 ADDR2  STMT   SOURCE STATEMENT

                                  516+** THIS RECORD IS WRITTEN AFTER SUCCESSFUL ALLOCATION. RECORD LENGTH
                                  517+** IS VARIABLE.
                                  518+**
000000                            519+          DS    0F          ALIGN TO FULL WORD BOUNDARY
                            00000 520+SMFRCD10 EQU   *           HEADER SEGMENT
000000                            521+SMF10LEN DS    BL2'0'      RECORD LENGTH
000002                            522+SMF10SEG DS    BL2'0'      SEGMENT DESCRIPTOR
000004 00                         523+SMF10FLG DC    BL1'0'      HEADER FLAG BYTE
000005 00                         524+SMF10RTY DC    BL1'0'      RECORD TYPE 10
000006 00000000                   525+SMF10TME DC    4BL1'0'     TOD USING FORMAT FROM TIME MACRO W/BIN. INTVL.
00000A 0000000C                   526+SMF10DTE DC    PL4'0000'   DATE IN PACKED DEC. FORM: 00YYDDDF
00000E 40404040                   527+SMF10SID DC    4CL1' '     SYSTEM IDENTIFICATION                    XL03034
000012 4040404040404040           528+SMF10JBN DC    8CL1' '     JOB NAME
00001A 00000000                   529+SMF10RST DC    4BL1'0'     READER START TIME
00001E 0000000C                   530+SMF10RSD DC    PL4'0000'   READER START DATE
000022 4040404040404040           531+SMF10UIF DC    8CL1' '     USER  IDENTIFICATION FIELD
00002A 0000                       532+SMF10LN  DC    2BL1'0'     LENGTH OF THE FIELD
00002C                            533+SMF10DEV DS    0C          DEVICES BEING MADE AVAILABLE (VARIABLE)
                                  534+**
                                  535+*FOR EACH DEVICE THERE IS A   4 -BYTE ENTRY HAVING THE FOLLOWING FORMAT
                                  536+**                          BYTE 0-DEVICE CLASS
                                  537+**                               1-UNIT TYPE
                                  538+**                               2-CHANNEL ADDRESS
                                  539+**                               3-UNIT ADDRESS
```

```
 LOC  OBJECT CODE    ADDR1 ADDR2  STMT   SOURCE STATEMENT

                                   559+* THIS RECORD IS WRITTEN WHEN A VARY OFFLINE COMMAND IS PROCESSED.
                                   560+* RECORD LENGTH IS VARIABLE.
                                   561+*
000000                             562+         DS    0F        ALIGN TO FULL WORD BOUNDARY
                             00000 563+SMFRCD11 EQU   *         HEADER SEGMENT
000000                             564+SMF11LEN DS    BL2'0'    RECORD LENGTH
000002                             565+SMF11SEG DS    BL2'0'    SEGMENT DESCRIPTOR
000004 00                          566+SMF11FLG DC    BL1'0'    HEADER FLAG BYTE
000005 00                          567+SMF11RTY DC    BL1'0'    RECORD TYPE 11
000006 00000000                    568+SMF11TME DC    4BL1'0'   TOD USING FORMAT FROM TIME MACRO W/BIN. INTVL.
00000A 0000000C                    569+SMF11DTE DC    PL4'0000' DATE IN PACKED DEC. FORM: 00YYDDDF
00000E 40404040                    570+SMF11SID DC    4CL1' '   SYSTEM IDENTIFICATION                    XL03034
000012 0000                        571+SMF11LN  DC    2BL1'0'   LENGTH OF THE FIELD
000014                             572+SMF11DEV DS    0C        DEVICES BEING REMOVED FROM I/O CONFIGURATION
                                   573+*
                                   574+* FOR EACH DEVICE BEING REMOVED, THERE IS A 4-BYTE ENTRY HAVING THE
                                   575+* FOLLOWING FORMAT:
                                   576+*              BYTE 0-DEVICE CLASS
                                   577+*                   1-UNIT TYPE
                                   578+*                   2-CHANNEL ADDRESS
                                   579+*                   3-UNIT ADDRESS
```

```
 LOC  OBJECT CODE     ADDR1 ADDR2  STMT   SOURCE STATEMENT

                                          599+* THIS RECORD IS WRITTEN WHEN THE HALT COMMAND IS PROCESSED. RECORD
000000                                    601+         DS    0F          ALIGN TO FULL WORD BOUNDARY
                                    00000 602+SMFRCD12 EQU   *           HEADER SEGMENT
000000                                    603+SMF12LEN DS    BL2'0'      RECORD LENGTH
000002                                    604+SMF12SEG DS    BL2'0'      SEGMENT DESCRIPTOR
000004 00                                 605+SMF12FLG DC    BL1'0'      HEADER FLAG BYTE
000005 00                                 606+SMF12RTY DC    BL1'0'      RECORD TYPE 12
000006 00000000                           607+SMF12TME DC    4BL1'0'     TOD, USING FORMAT FROM TIME MACRO W/BIN. INTVL
00000A 0000000C                           608+SMF12DTE DC    PL4'0000'        DATE IN PACKED DECIMAL FORM: 00YYDDDF
                                          609+*                               ( F IS A SIGN)
00000E 40404040                           610+SMF12SID DC    4CL1' '     SYSTEM IDENTIFICATION                    XL03034
000012 00000000                           611+SMF12SWT DC    4BL1'0'     SYSTEM WAIT TIME IN HUNDREDTHS OF SECONDS
                                          612+*                          SINCE LAST TYPE1 RECORD
000016 00000000                           613+SMF12TEX DC    4BL1'0'     COLLECT END TIME
00001A 00000000                           614+SMF12PGI DC    4BL1'0'     TOTAL PAGE-INS
00001E 00000000                           615+SMF12PGO DC    4BL1'0'     TOTAL PAGE-OUTS
000022 00000000                           616+SMF12PGR DC    4BL1'0'     TOTAL PAGES RECLAIMED
```

```
 LOC  OBJECT CODE    ADDR1 ADDR2  STMT   SOURCE STATEMENT

                                  636+** RECORD TYPE: 14- DATA SET OPENED FOR INPUT/RDBACK
                                  637+**               15-DATA SET OPENED FOR OUTPUT/INOUT/OUTIN/UPDAT
000000                            638+         OS    0F        ALIGN TO FULL WORD BOUNDARY
                           00000  639+SMF14HDR EQU   *         HEADER SEGMENT FOR RECORD TYPE 14/15
000000                            640+SMF14LEN OS    BL2'0'    RECORD LENGTH
000002                            641+SMF14SEG OS    BL2'0'    SEGMENT DESCRIPTOR
000004                            642+SMF14FLG DS    BL1'0'    HEADER FLAG BYTE
000005                            643+SMF14RTY OS    BL1'0'    RECORD TYPE 14
000006                            644+SMF14TME OS    BL4'0'    TOD, USING TIME MACRO W/BIN. INTVL.
00000A                            645+SMF14DTE DS    PL4'0000' DATE, PACKED/DEC FORM; 00YYDDDF
00000E                            646+SMF14SID DS    CL4' '    SYSTEM ID                                  XL03034
000012                            647+SMF14JBN DS    CL8' '    JOB NAME
00001A                            648+SMF14RST OS    BL4'0'    READER START TIME
00001E                            649+SMF14RSD OS    PL4'0000' READER START DATE, PK/DEC FORM: 00YYDODF
000022                            650+SMF14UID OS    CL8' '    USERID
00002A                            651+SMF14RIN OS    BL2'0'    RECORD INDICATORS:
                           00080  652+SMF14RVO EQU   128       RESERVED
                           00040  653+SMF14EOV EQU   64        RECORD WRITTEN BY EOV
                           00020  654+SMF14DAD EQU   32        DASD DEVICE
                           00010  655+SMF14TOS EQU   16        TEMPORARY DATA SET
                           0000B  656+SMF14DOA EQU   B         DCBDSORG = DA
                           00004  657+SMF14IS  EQU   4         DCBDSORG = IS
                           00002  658+SMF14JIS EQU   2         JFCDSORG = IS
                                  659+*        BITS 7 - 15     RESERVED
00002C                            660+SMF14SDC DS    BL1'0'    SIZE OF DCB/DEB SEGMENT
00002D                            661+SMF14NUC OS    BL1'0'    NO. UCB SEGMENTS
00002E                            662+SMF14SUC DS    BL1'0'    SIZE OF EACH UCB SEGMENT
00002F                            663+SMF14SET DS    BL1'0'    SIZE OF EXTENSION SEGMENT
000030                            664+SMF14RV1 DS    BL4'0'    RESERVED
                           00034  665+SMFTIOT  EQU   *         TIOT SEGMENT (FROM DD ENTRY)
000034                            666+SMFTIOE1 DS    BL1'0'    TIOELNGH
000035                            667+SMFTIOE2 OS    BL1'0'    TIOESSTA
000036                            668+SMFTIOE3 DS    BL1'0'    TIOEWTCT
000037                            669+SMFTIOE4 OS    BL1'0'    TIOELINK
00003B                            670+SMFTIOE5 DS    BL8'0'    TIOEDDNM
000040                            671+SMFTIOE6 DS    BL3'0'    TIOEJFCB
000043                            672+SMFTIOE7 OS    BL1'0'    TIOESTTC
                                  673+*
000044                            674+SMFJFCB1 OS    XL176'0'  JFCB SEGMENT (ENTIRE JFCB)
                                  675+*
                           000F4  676+SMFDCBDE EQU   *         DCB/DEB SEGMENT (TAPE AND DASD)
0000F4                            677+SMFDCBOR OS    BL2'0'    DCBDSORG
0000F6                            678+SMFDCBRF DS    BL1'0'    DCBRECFM
0000F7                            679+SMFDCBMF OS    BL2'0'    DCBMACRF
0000F9                            680+SMFDCBFL OS    BL1'0'    DCBOFLGS
0000FA                            681+SMFDCBOP OS    BL1'0'    DCBOPTCD
0000FB                            682+SMF14RV2 DS    BL1'0'    RESERVED
0000FC                            683+SMFDEBFL DS    BL1'0'    DEBOFLGS
0000FD                            684+SMFDEBOP DS    BL1'0'    DEBOPATB
0000FE                            685+SMFDFBVL OS    BL2'0'    DEBVOLSQ
```

```
 LOC  OBJECT CODE    ADDR1 ADDR2  STMT   SOURCE STATEMENT

                           00100   686+SMFTDDEX EQU   *          TAPE DCB/DEB EXTENSION
000100                             687+SMFDCBBL DS    BL4'0'     DCBBLKCT
000104                             688+SMFDSSNO DS    BL6'0'     DATA SET SERIAL NUMBER
00010A                             689+SMF14RV3 DS    BL2'0'     RESERVED
00010C                     00100   690+         ORG   SMFTDDEX
000100                             691+SMF14NTU DS    BL4'0'     NUMBER OF TRACKS USED(LAST TTR-FROM DCBFDAD)
000104                             692+SMF14NTR DS    BL4'0'     NUMBER OF TRACKS RELEASED BY DADSM RELSE RTN
000108                             693+SMF14NER DS    BL1'0'     NUMBER OF EXTENTS RELEASED
000109                             694+SMF14RV4 DS    BL3'0'     RESERVED
                           0010C   695+SMF14UCB EQU   *          UCB SEGMENT(ONE FOR EACH UCB FOR THE DATA SET)
00010C                             696+SMFUCBCH DS    BL1'0'     UCBCHA
00010D                             697+SMFUCBUA DS    BL1'0'     UCBUA
00010E                             698+SMFSRTEV DS    BL6'0'     SRTEVOLI
000114                             699+SMFUCBTY DS    BL4'0'     UCBTYP
000118                             700+SMFSRTES DS    BL1'0'     SRTESTAB
000119                             701+SMF14NEX DS    BL1'0'     NUMBER EXTENTS
00011A                             702+SMF14RV5 DS    BL2'0'     RESERVED
00011C                             703+SMFEXCP  DS    BL4'0'     EXCP COUNT
                           00120   704+SMFTAPEX EQU   *          SMF TAPE EXTENSION
                                   705+*        THIS SECTION DESCRIBES THE TAPE EXTENSION SEGMENT
000120                             706+SMFSRTEF DS    BL2'0'     SRTEFSCT
000122                             707+SMFSRTEQ DS    BL2'0'     SRTEFSEQ
000124                     00120   708+         ORG   SMFTAPEX
                                   709+*        THIS SECTION DESCRIBES THE DASD EXTENSION SEGMENT
000120                             710+SMF14NTA DS    BL4'0'     NUMBER TRACKS ALLOCATED
                           00124   711+SMF14UCE EQU   *               UCB SEGMENT END
000124                     0010C   712+         ORG   SMF14UCB
                                   713+*
                           0010C   714+SMFISAMX EQU   *          ISAM EXTENSION SEG.(FOR DCBDSORG = IS)
00010C                             715+SMF14RV6 DS    BL2'0'     RESERVED
00010E                             716+SMFDCBMA DS    BL1'0'     DCBMAC
00010F                             717+SMFDCBNL DS    BL1'0'     DCBNLEV
000110                             718+SMFDCBR3 DS    BL4'0'     DCBRORG3
000114                             719+SMFDCBNR DS    BL4'0'     DCBNREC
000118                             720+SMFDCBR2 DS    BL2'0'     DCBRORG2
00011A                             721+SMFDCBNO DS    BL2'0'     DCBNOREC
00011C                             722+SMFDCBR1 DS    BL2'0'     DCBRORG1
00011E                             723+SMF14RV7 DS    BL1'0'     RESERVED
00011F                             724+SMFDEBNI DS    BL1'0'     DEBNIEE
000120                             725+SMFDEBNP DS    BL1'0'     DEBNPEE
000121                             726+SMFDEBNO DS    BL1'0'     DEBNOEE
000122                             727+SMFNCYLS DS    BL2'0'     # CYL'S IN INDEPENDENT INDEX AREA
000124                             728+SMFNPCYL DS    BL2'0'     NUMBER OF CYL'S IN PRIME AREA
000126                             729+SMFNOCYL DS    BL2'0'     NUMBER OF CYL'S IN INDEPENDENT OVFL AREA
000128                     0010C   730+         ORG   SMFISAMX
```

```
 LOC  OBJECT CODE    ADDR1 ADDR2  STMT   SOURCE STATEMENT

000000                          750+          DS    0F        ALIGN TO FULL WORD BOUNDARY
                          00000 751+SMF17HDR  EQU   *         HEADER SEGMENT FOR RECORD TYPE17(SCRATCH)
000000                          752+SMF17LEN  DS    BL2'0'    RECORD LENGTH
000002                          753+SMF17SEG  DS    BL2'0'    SEGMENT DESCRIPTOR
000004                          754+SMF17FLG  DS    BL1'0'    HEADER FLAG BYTE
000005                          755+SMF17RTY  DS    BL1'0'    RECORD TYPE 17
000006                          756+SMF17TME  DS    BL4'0'    TOD, USING TIME MACRO W/BIN. INTVL.
00000A                          757+SMF17DTE  DS    PL4'0000' DATE, PACKED/DEC FORM: 00YYDDDF
00000E                          758+SMF17SID  DS    CL4' '    SYSTEM ID                                XL03034
000012                          759+SMF17JBN  DS    CL8' '    JOB NAME
00001A                          760+SMF17RST  DS    BL4'0'    READER START TIME
00001E                          761+SMF17RSD  DS    PL4'0000' READER START DATE, PK/DEC FORM: 00YYDDDF
000022                          762+SMF17UID  DS    CL8' '    USERID
00002A                          763+SMF17RIN  DS    BL2'0'    RECORD INDICATORS: ALL BITS RESERVED(0)
00002C                          764+SMF17DSN  DS    CL44' '   DSNAME
000058                          765+SMF17RV1  DS    BL3'0'    RESERVED
000058                          766+SMF17NVL  DS    BL1'0'    N, # VOL SERS
                                767+* VOLUME SERIAL ENTRY (ONE FOR EACH VOLUME)
00005C                          768+SMF17RV2  DS    BL2'0'    RESERVED
00005E                          769+SMF17FVL  DS    XL6'00'   FIRST VOL SER # (SRTEVOLI)
                          00008 770+SMF17LNV  EQU   *-SMF17RV2    LENGTH OF VOL SER ENTRY
000064                          0005C 771+          ORG   SMF17RV2
```

```
 LOC  OBJECT CODE    ADDR1 ADDR2  STMT   SOURCE STATEMENT

000000                          791+          DS    0F        ALIGN TO FULL WORD BOUNDARY
                          00000 792+SMF18HDR  EQU   *         HEADER SEGMENT FOR RECORD TYPE18(RENAME)
000000                          793+SMF18LEN  DS    BL2'0'    RECORD LENGTH
000002                          794+SMF18SEG  DS    BL2'0'    SEGMENT DESCRIPTOR
000004                          795+SMF18FLG  DS    BL1'0'    HEADER FLAG BYTE
000005                          796+SMF18RTY  DS    BL1'0'    RECORD TYPE 18
000006                          797+SMF18TME  DS    BL4'0'    TOD, USING TIME MACRO W/BIN. INTVL.
00000A                          798+SMF18DTE  DS    PL4'0000' DATE, PACKED/DEC FORM: 00YYDDDF
00000E                          799+SMF18SID  DS    CL4' '    SYSTEM ID                                XL03034
000012                          800+SMF18JBN  DS    CL8' '    JOB NAME
00001A                          801+SMF18RST  DS    BL4'0'    READER START TIME
00001E                          802+SMF18RSD  DS    PL4'0000' READER START DATE, PK/DEC FORM: 00YYDDDF
000022                          803+SMF18UID  DS    CL8' '    USERID
00002A                          804+SMF18RIN  DS    BL2'0'    RECORD INDICATORS: ALL BITS RESERVED(0)
00002C                          805+SMF18ODS  DS    CL44' '   OLD DSNAME
000058                          806+SMF18NDS  DS    CL44' '   NEW DSNAME
000084                          807+SMF18RV1  DS    BL3'0'    RESERVED
000087                          808+SMF18NVL  DS    BL1'0'    N, THE # VOL SERS
                                809+* VOLUME SERIAL ENTRY (ONE FOR EACH ENTRY)
000088                          810+SMF18RV2  DS    BL2'0'    RESERVED
00008A                          811+SMF18FVL  DS    CL6' '    FIRST VOL SER # (SRTEVOLI)
                          00008 812+SMF18LNV  EQU   *-SMF18RV2    LENGTH OF VOL SER ENTRY
000090                  00088   813+          ORG   SMF18RV2
```

```
  LOC  OBJECT CODE    ADDR1 ADDR2  STMT    SOURCE STATEMENT

000000                           833+          DS    0F        ALIGN TO FULL WORD BOUNDARY
                           00000 834+SMF19HDR  EQU   *         HEADER SEGMENT FOR RECORD TYPE19(LSPACE)
000000                           835+SMF19LEN  DS    BL2'0'    RECORD LENGTH
000002                           836+SMF19SEG  DS    BL2'0'    SEGMENT DESCRIPTOR
000004                           837+SMF19FLG  DS    BL1'0'    HEADER FLAG BYTE
000005                           838+SMF19RTY  DS    BL1'0'    RECORD TYPE 19
000006                           839+SMF19TME  DS    BL4'0'    TOD, USING TIME MACRO W/BIN. INTVL.
00000A                           840+SMF19DTE  DS    PL4'0000' DATE, PK/DEC. FORM: 00YYDDDF
00000E                           841+SMF19SID  DS    CL4' '    SYSTEM IDENTIFICATION                    XL03034
000012                           842+SMF19RV1  DS    BL2'0'    RESERVED
000014                           843+SMF19VOL  DS    CL6' '    VOLUME SERIAL #
00001A                           844+SMF19OID  DS    CL10' '   OWNER ID
000024                           845+SMF19DEV  DS    BL4'0'    DEVICE TYPE
000028                           846+SMF19VTC  DS    BL5'0'    VTOC ADDRESS
00002D                           847+SMF19VTI  DS    BL1'0'    DS4VTOCI
00002E                           848+SMF19NDS  DS    BL2'0'    # DSCB'S
000030                           849+SMF19DSR  DS    BL2'0'    # FORMAT0 DSCB'S
000032                           850+SMF19NAT  DS    BL2'0'    # UNUSED ALTERNATE TRACKS
000034                           851+SMF19SPC  DS    BL4'0'    TOTAL # UNALLOCATED CYL'S AND TRKS
000038                           852+SMF19LEX  DS    BL4'0'    # CYL'S AND TRKS IN LARGEST UNALLOCATED EXT.
00003C                           853+SMF19NUE  DS    BL2'0'    # UNALLOCATED EXTENTS
00003E                           854+SMF19RV2  DS    BL2'0'    RESERVED
000040                           855+SMF19CUU  DS    BL2'0'    CHANNEL AND UNIT ADDRESS                 A40793
000042                           856+SMF19IND  DS    BL2'0'    DASD MODULE IO OR DRIVE NUMBER           A40793
                                 857+*                         INDICATING PHYSICAL IDENTITY OF DEVICES
                                 858+*                         HAVING MOVEABLE ADDRESS PLUGS
```

APPENDIX E: BASIC SMF RECORD LAYOUTS WITH OFFSETS

```
 LOC  OBJECT CODE    ADDR1 ADDR2  STMT   SOURCE STATEMENT

000000                         878+         DS    0F        ALIGN TO FULL WORD BOUNDARY
                         00000 879+SMF20HDR EQU   *  HEADER SEGMENT FOR RECORD TYPE20(JOB COMMENCEMENT)
000000                         880+SMF20LEN DS    BL2'0'    RECORD LENGTH
000002                         881+SMF20SEG DS    BL2'0'    SEGMENT DESCRIPTOR
000004                         882+SMF20FLG DS    BL1'0'    HEADER FLAG BYTE
000005                         883+SMF20RTY DS    BL1'0'    RECORD TYPE 20
000006                         884+SMF20TME DS    BL4'0'    TOD, USING TIME MACRO W/BIN. INTVL.
00000A                         885+SMF20DTE DS    PL4'0000' DATE, PACKED/DEC FORM; 00YYDDDF
00000E                         886+SMF20SID DS    CL4' '    SYSTEM ID                                XL03034
000012                         887+SMF20JBN DS    CL8' '    JOB NAME
00001A                         888+SMF20RST DS    BL4'0'    READER START TIME
00001E                         889+SMF20RSD DS    PL4'0000' READER START DATE, PK/DEC FORM: 00YYDDDF
000022                         890+SMF20UID DS    CL8' '    USERID
00002A                         891+SMF20RIN DS    BL2'0'    RECORD INDICATORS: ALL BITS RESERVED(0)
00002C                         892+SMF20PGM DS    CL20' '   PROGRAMMERS NAME
000040                         893+SMF20NAF DS    BL1'0'    NO. ACCOUNTING FIELDS
                         00041 894+SMF20ACT EQU   *         ACCOUNTING FIELDS
```

```
 LOC  OBJECT CODE    ADDR1 ADDR2  STMT   SOURCE STATEMENT

                                  914+* THIS RECORD IS WRITTEN BY THE ERROR STATISTICS BY VOLUME OPTION
                                  915+* WHENEVER A VOLUME IS DISMOUNTED.
000000                            917+         OS    OF         ALIGN TO FULL WORD BOUNDRY
                            00000 918+SMF21HDR EQU   *          HEADER SEGMENT FOR RECORD 21 (ESV)
000000                            919+SMF21LEN OS    BL2'0'     RECORD LENGTH
000002                            920+SMF21SEG OS    BL2'0'     SEGMENT DESCRIPTOR
000004                            921+SMF21FLG DS    BL1'0'     HEADER FLAG BYTE
000005                            922+SMF21RTY DS    BL1'0'     RECORD TYPE 21
000006                            923+SMF21TME DS    BL4'0'     TOD, USING TIME MACRO W/BIN. INTVL.
00000A                            924+SMF21DTE DS    PL4'0000'  DATE, PACKED/OFC FORM: 00YYDDDF
00000E                            925+SMF21SID DS    CL4' '     SYSTEM ID                                  XL03034
000012                            926+SMF21LGH OS    2BL1'0'    LENGTH OF DATA PORTION
000014                            927+SMF21VOL OS    6CL1' '    VOL SERIAL
00001A                            928+SMF21CA  OS    2BL1'0'    CHANNEL ADDRESS
00001C                            929+SMF21UCB OS    4BL1'0'    UCB TYPE CODE
000020                            930+SMF21TR  DS    BL1'0'     TEMPERARY READ ERRORS
000021                            931+SMF21TW  DS    BL1'0'     TEMPERARY WRITE ERRORS
000022                            932+SMF21SIO OS    2BL1'0'    NO OF START I/O
000024                            933+SMF21PR  DS    BL1'0'     PERMANEMT READ ERRORS
000025                            934+SMF21PW  DS    BL1'0'     PERMANENT WRITE ERRORS
000026                            935+SMF21NB  OS    BL1'0'     NOISE BLOCKS
000027                            936+SMF21ERG DS    2BL1'0'    ERASE GAPS
000029                            937+SMF21CLN OS    2BL1'0'    CLEANER ACTIONS
00002B                            938+SMF21DEN DS    BL1'0'     TAPE DENSITY
00002C                            939+SMF21BLS OS    2BL1'0'    BLOCK SIZE
00002E                            940+SMF21RVO DS    2BL1'0'    RESERVED


                                  942        END
```

APPENDIX E: BASIC SMF RECORD LAYOUTS WITH OFFSETS

CROSS-REFERENCE

SYMBOL LENGTH,VALUE,DEFN REFERENCES

```
SMFDCBBL  4,100,687    SMFDCBDE  1,F4,676    SMFDCBFL  1,F9,680    SMFDCBMA  1,10E,716
SMFDCBMF  2,F7,679     SMFDCBNL  1,10F,717   SMFDCBNO  2,11A,721   SMFDCBNR  4,114,719
SMFDCBOP  1,FA,681     SMFDCBOR  2,F4,677    SMFDCBRF  1,F6,678    SMFDCBR1  2,11C,722
SMFDCBR2  2,118,720    SMFDCBR3  4,110,718   SMFDEBFL  1,FC,683    SMFDEBNI  1,11F,724
SMFDEBNO  1,121,726    SMFDEBNP  1,120,725   SMFDEBOP  1,FD,684    SMFDEBVL  2,FE,685
SMFDSSNO  6,104,688    SMFEXCP   4,11C,703   SMFISAMX  1,10C,714  730   SMFJFCB1  176,44,674
SMFNCYLS  2,122,727    SMFNDCYL  2,126,729   SMFNPCYL  2,124,728   SMFRCDO  1,0,39
SMFRCD1   1,0,86    SMFRCD10  1,0,520    SMFRCD11  1,0,563    SMFRCD12  1,0,602    SMFRCD2  1,0,125
SMFRCD3   1,0,157    SMFRCD4  1,0,188    SMFRCD5  1,0,278    SMFRCD6  1,0,351    SMFRCD7  1,0,403
SMFRCD8   1,0,441    SMFRCD9  1,0,480    SMFRECS  1,0,15  17 62 103 135 166 256 328 380 419 458 498 541 581 618 732 773
  815 860 896   SMFSRTEF  2,120,706    SMFSRTEQ  2,122,707    SMFSRTES  1,118,700
SMFSRTFV  6,10F,698    SMFTAPEX  1,120,704  708   SMFTDDEX  1,100,686  690   SMFTIOE1  1,34,666
SMFTIOF2  1,35,667    SMFTIOF3  1,36,668    SMFTIOE4  1,37,669    SMFTIOE5  8,38,670
SMFTIOE6  3,40,671    SMFTIOE7  1,43,672    SMFTIOT  1,34,665    SMFUCBCH  1,10C,696
SMFUCBTY  4,114,699    SMFUCBUA  1,10D,697    SMFOBUF  1,16,49    SMFODTE  4,A,45    SMFOFLG  1,4,42
SMFOJWT   1,12,48    SMFOLEN  2,0,40    SMFOOPT  1,1E,51    SMFORST  1,1F,60    SMFORTY  1,5,43
SMFOSEG   2,2,41    SMFOSID  1,E,47    SMFOTME  1,6,44    SMFOVST  1,1A,50    SMF1DTE  4,A,92
SMF1FLG   1,4,89    SMF1LEN  2,0,87    SMF1PGIN  1,1A,99    SMF1PGOT  1,1E,100    SMF1PGRL  1,22,101
SMF1RTY   1,5,90    SMF1SEG  2,2,88    SMF1SID  1,E,94    SMF1SWT  1,12,95    SMF1TEXP  1,16,98
SMF1TME   1,6,91    SMF10DEV  1,2C,533    SMF10DTE  4,A,526    SMF10FLG  1,4,523    SMF10JBN  1,12,528
SMF10LEN  2,0,521    SMF10LN  1,2A,532    SMF10RSD  4,1E,530    SMF10RST  1,1A,529
SMF10RTY  1,5,524    SMF10SEG  2,2,522    SMF10SID  1,E,527    SMF10TME  1,6,525    SMF10UIF  1,22,531
SMF11DEV  1,14,572    SMF11DTF  4,A,569    SMF11FLG  1,4,566    SMF11LEN  2,0,564    SMF11LN  1,12,571
SMF11RTY  1,5,567    SMF11SEG  2,2,565    SMF11SID  1,E,570    SMF11TME  1,6,568    SMF12DTF  4,A,608
SMF12FLG  1,4,605    SMF12LEN  2,0,603    SMF12PGI  1,1A,614    SMF12PGO  1,1E,615
SMF12PGR  1,22,616    SMF12RTY  1,5,606    SMF12SEG  2,2,604    SMF12SID  1,E,610
SMF12SWT  1,12,611    SMF12TEX  1,16,613    SMF12TME  1,6,607    SMF14DAD  1,20,654
SMF14DDA  1,8,656    SMF14DTE  4,A,645    SMF14EOV  1,40,653    SMF14FLG  1,4,642    SMF14HDR  1,0,639
SMF14IS  1,4,657    SMF14JBN  8,12,647    SMF14JIS  1,2,658    SMF14LEN  2,0,640
SMF14NER  1,108,693    SMF14NEX  1,119,701    SMF14NTA  4,120,710    SMF14NTR  4,104,692
SMF14NTU  4,100,691    SMF14NUC  1,2D,661    SMF14RIN  2,2A,651    SMF14RSD  4,1E,649
SMF14RST  4,1A,648    SMF14RTY  1,5,643    SMF14RV0  1,80,652    SMF14RV1  4,30,664
SMF14RV2  1,FB,682    SMF14RV3  2,10A,689    SMF14RV4  3,109,694    SMF14RV5  2,11A,702
SMF14RV6  2,10C,715    SMF14RV7  1,11E,723    SMF14SDC  1,2C,660    SMF14SEG  2,2,641
SMF14SET  1,2F,663    SMF14SID  4,E,646    SMF14SUC  1,2E,662    SMF14TDS  1,10,655
SMF14TME  4,6,644    SMF14UCB  1,10C,695  712   SMF14UCF  1,124,711    SMF14UID  8,22,650
SMF17DSN  44,2C,764    SMF17DTE  4,A,757    SMF17FLG  1,4,754    SMF17FVL  6,5E,769
SMF17HDR  1,0,751    SMF17JBN  8,12,759    SMF17LEN  2,0,752    SMF17LNV  1,8,770
SMF17NVL  1,5B,766    SMF17RIN  2,2A,763    SMF17RSD  4,1F,761    SMF17RST  4,1A,760
SMF17RTY  1,5,755    SMF17RV1  3,58,765    SMF17RV2  2,5C,768  770 771   SMF17SEG  2,2,753
SMF17SID  4,E,758    SMF17TME  4,6,756    SMF17UID  8,22,762    SMF18DTE  4,A,798    SMF18FLG  1,4,795
SMF18FVL  6,8A,811    SMF18HDR  1,0,792    SMF18JBN  8,12,800    SMF18LEN  2,0,793
SMF18LNV  1,8,812    SMF18NDS  44,58,806    SMF18NVL  1,87,808    SMF18ODS  44,2C,805
SMF18RIN  2,2A,804    SMF18RSD  4,1E,802    SMF18RST  4,1A,801    SMF18RTY  1,5,796
SMF18RV1  3,84,807    SMF18RV2  2,88,810  812 813   SMF18SEG  2,2,794    SMF18SID  4,E,799
SMF18TME  4,6,797    SMF18UID  8,22,803    SMF19CUU  2,40,855    SMF19DEV  4,24,845
SMF19DSR  2,30,849    SMF19DTE  4,A,840    SMF19FLG  1,4,837    SMF19HDR  1,0,834
SMF19IND  2,42,856    SMF19LEN  2,0,835    SMF19LEX  4,38,852    SMF19NAT  2,32,850
SMF19NDS  2,2E,848    SMF19NUF  2,3C,853    SMF19OID  10,1A,844    SMF19RTY  1,5,838
SMF19RV1  2,12,842    SMF19RV2  2,3E,854    SMF19SEG  2,2,836    SMF19SID  4,E,841
SMF19SPC  4,34,851    SMF19TME  4,6,839    SMF19VOL  6,14,843    SMF19VTC  5,28,846
SMF19VTI  1,2D,847    SMF2DTE  4,A,131    SMF2FLG  1,4,128    SMF2LEN  2,0,126    SMF2RTY  1,5,129
```

APPENDIX E: BASIC SMF RECORD LAYOUTS WITH OFFSETS

CROSS-REFERENCE

SYMBOL LENGTH,VALUE,DEFN REFERENCES

```
SMF2SEG  2,2,127    SMF2SID  1,E,133    SMF2TME  1,6,130    SMF20ACT  1,41,894    SMF20DTE  4,A,885
SMF20FLG  1,4,882    SMF20HDR  1,0,879    SMF20J8N  8,12,887    SMF20LEN  2,0,880
SMF20NAF  1,40,893    SMF20PGM  20,2C,892    SMF20RIN  2,2A,891    SMF20RSD  4,1E,889
SMF20RST  4,1A,888    SMF20RTY  1,5,883    SMF20SEG  2,2,881    SMF20SID  4,E,886    SMF20TME  4,6,884
SMF20UID  8,22,890    SMF213LS  1,2C,939    SMF21CA  1,1A,928    SMF21CLN  1,29,937
SMF21DEN  1,2B,938    SMF21DTE  4,A,924    SMF21ERG  1,27,936    SMF21FLG  1,4,921
SMF21HDR  1,0,918    SMF21LEN  2,0,919    SMF21LGH  1,12,926    SMF21N8  1,26,935    SMF21PR  1,24,933
SMF21PW  1,25,934    SMF21RTY  1,5,922    SMF21RVO  1,2E,940    SMF21SEG  2,2,920    SMF21SID  4,F,925
SMF21SIO  1,22,932    SMF21TME  4,6,923    SMF21TR  1,20,930    SMF21TW  1,21,931
SMF21UC8  1,1C,929    SMF21VOL  1,14,927    SMF3DTE  4,A,163    SMF3FLG  1,4,160    SMF3LEN  2,0,158
SMF3RTY  1,5,161    SMF3SEG  2,2,159    SMF3SID  1,E,164    SMF3TME  1,6,162    SMF4ACTF  1,71,246
SMF4AST  1,5A,226    SMF4CUAD  1,6E,239    SMF4DEVC  1,6C,237  242    SMF4DTF  4,A,194
SMF4EXCP  1,70,240    SMF4FLG  1,4,191    SMF4HOST  1,4E,212    SMF4JBN  1,12,196    SMF4LEN  2,0,189
SMF4LENN  1,6A,233    SMF4LNTH  1,6C,243    SMF4NAF  1,70,245    SMF4NCI  1,33,203
SMF4PGIN  1,71,253    SMF4PGMN  8,3A,208    SMF4PGOT  1,75,254    SMF4PPST  1,5E,227
SMF4PRTY  1,39,206    SMF4RDS  4,1E,198    SMF4RIN  1,66,229    SMF4RLCT  1,68,232
SMF4RSHO  1,4A,210    SMF4RST  1,1A,197    SMF4RTY  1,5,192    SMF4SCC  1,37,205    SMF4SEG  2,2,190
SMF4SETM  1,6D,244    SMF4SID  1,E,195    SMF4SIT  1,2B,201    SMF4SPK  1,56,214    SMF4STI  1,57,215
SMF4STID  4,2F,202    SMF4STMN  8,42,209    SMF4STN  1,2A,200    SMF4TME  1,6,193    SMF4UIF  1,22,199
SMF4UTYP  1,6D,238    SMF5ACTF  1,78,321    SMF5DTE  4,A,284    SMF5FLG  1,4,281    SMF5J8N  1,12,286
SMF5J8TI  1,42,299    SMF5JCC  1,37,295    SMF5JCPU  1,75,320    SMF5JICL  1,4B,313
SMF5JID  4,2F,292    SMF5JIT  1,2B,291    SMF5JPTY  1,39,296    SMF5JSAF  1,79,322    SMF5LEN  2,0,279
SMF5NCI  1,33,293    SMF5NST  1,2A,290    SMF5PRGN  1,61,319    SMF5QID  1,50,316
SMF5RDCL  1,49,311    SMF5RSD  4,1E,288    SMF5RST  1,1A,287    SMF5RSTD  4,3F,298
SMF5RSTT  1,3A,297    SMF5RTY  1,5,282    SMF5RUTY  1,4A,312    SMF5SEG  2,2,280    SMF5SID  1,E,285
SMF5SMCI  1,43,309    SMF5SPK  1,4C,314    SMF5TLEN  1,60,318    SMF5TME  1,6,283    SMF5UIF  1,22,289
SMF6DTE  4,A,357    SMF6FLG  1,4,354    SMF6FMN  1,39,377    SMF6IOE  1,37,369    SMF6JBN  1,12,360
SMF6LEN  2,0,352    SMF6NDS  1,38,375    SMF6NLR  1,33,367    SMF6OWC  1,2A,364    SMF6QID  1,3D,378
SMF6RSD  4,1E,362    SMF6RST  1,1A,361    SMF6RTY  1,5,355    SMF6SFG  2,2,353    SMF6SID  1,E,359
SMF6TME  1,6,356    SMF6UIF  8,22,363    SMF6WSD  4,2F,366    SMF6WST  1,28,365    SMF7DTF  4,A,409
SMF7FLG  1,4,406    SMF7LEN  2,0,404    SMF7NRO  1,12,412    SMF7RTY  1,5,407    SMF7SEG  2,2,405
SMF7SID  1,F,411    SMF7STD  4,18,416    SMF7STM  1,14,413    SMF7TME  1,6,408    SMF8DTF  4,A,447
SMF8FLG  1,4,444    SMF8IODV  1,14,452    SMF8LEN  2,0,442    SMF8LENN  1,12,449    SMF8RTY  1,5,445
SMF8SFG  2,2,443    SMF8SID  1,E,448    SMF8TME  1,6,446    SMF9DTE  4,A,486    SMF9DVAD  1,14,489
SMF9FLG  1,4,483    SMF9LEN  2,0,481    SMF9LENN  1,12,488    SMF9RTY  1,5,484    SMF9SEG  2,2,482
SMF9SID  1,E,487    SMF9TME  1,6,485
```

Appendix F: Removing SMF from OS/MVT, Release 21.6

The following modules of OS/360 MVT, Release 21.6, were altered using the IMASPZAP utility program ("Superzap") described in <36> to remove SMF overhead from the system for the validation experiments.

1. Master Scheduler Initialization Routine

This routine (module IEEVIPL in SYS1.LINKLIB) is created during the Sysgen process. Since the inclusion of SMF is a Sysgen option, the generated code includes an unconditional XCTL to SMF initialization, which must be removed. The changes were as follows:

a. Instead of placing option flags in the CVTSMCA field prior to SMF initialization, the CVTSMCA pointer was zeroed out.

b. The XCTL was altered to transfer to routine IEEVWAIT, signalling the end of Master Scheduler initialization.

2. External and I/O FLIH's

These routines, in CSECT IEAQNU00 of the nucleus, are also created during Sysgen and include unconditional BAL instructions to invoke the SMF System Wait Time Collection Routine, IEAQWAIT. The BAL's were changed to NOP's.

3. SVC 83 Routine

This routine (module IGC0008C in SYS1.SVCLIB) is never entered during normal OS processing without SMF and hence contains no check to ensure that SMF is in the system before performing its functions. Since HASP had been altered to issue SVC 83 also without checking, the SVC 83 routine was changed to cause it to zero out register 15 and return via register 14 immediately upon entry. It was later decided to alter HASP as described below to prevent it from issuing SVC 83, but the change to IGC0008C was left in to take care of any other components of the system which might issue SVC 83 unexpectedly.

In addition to the Superzap changes noted above, a HASP "REP" card was used to change the SVC 83 issued by HASP to a NOP. This was done for all systems, not just those with SMF removed. The point of the HASP change was to reduce SMF overhead to only that generated by the OS record types, eliminating user-defined types such as the HASP records.

Appendix G: Chained Scheduling and EXCP Counting

The purpose of this appendix is to explain the variability of EXCP counts for chained-scheduled I/O operations. This discussion will cover both OS, where chained scheduling is a seldom-used option, and MVS (VS2 Release 2 and later releases) where it is mandatory. Under MVS, chained scheduling is forced for all sequential I/O that uses either the Virtual Storage Access Method (VSAM) or the Queued Sequential Access Method (QSAM). The following discussion assumes, for concreteness, that the data set in question is opened for output using QSAM.

At OPEN time, access method routines construct one channel program for each buffer. (For chained scheduling to be applicable, the data set must have at least two buffers.) Each channel program consists of a series of CCW's (channel command words) chained together so that the channel will fetch and obey them consecutively. The last CCW in each series is a no-operation (NO-OP), with chaining not specified; this causes the channel to cease fetching CCW's and interrupt the CPU. As part of the interruption, a CSW (channel status word) indicating channel end and device end is stored in low core. In the address field of the last NO-OP CCW is a pointer to the start of the channel program for the next buffer.

The first time a buffer becomes full, the PUT routine requests execution of the corresponding channel program by issuing an EXCP. Each subsequent occurrence of a buffer filling up causes the PUT routine to change the NO-OP CCW in the channel program for the previous buffer to a TIC (transfer in channel). This operation is referred to as "joining" the channel programs. If the channel has not yet fetched the NO-OP CCW, the join will be successful, and the processing by the channel will proceed smoothly from one buffer to the next with no channel-end/device-end interruption. If the join is unsuccessful, the PUT routine will detect this fact and issue another EXCP for the channel program corresponding to the most recently filled buffer.

Under OS, the first CCW in each channel program has a bit set to make the channel interrupt the CPU as soon as the channel fetches and decodes the CCW. The CSW stored as a result has a bit on that indicates the occurrence of a PCI (program-controlled interruption). Unlike other I/O interruptions, PCI's are not stacked by the hardware; the only result of a PCI which occurs when the CPU is masked for interrupts is that the CSW stored for the next interruption taken on that channel will have the PCI bit on. The effect of the PCI at the beginning of each channel program is to activate the PCI appendage routine. If the channel program just starting is the first in a chain of joined channel programs, this routine does nothing. If it is the second or subsequent, the appendage forces its last CCW to be a no-op, thereby "parting" it from the one following. The same routine

also serves as the channel-end appendage; it posts ICB's (interruption control blocks) and IOB's (I/O blocks) to keep the PUT routine informed of which buffers are available for re-use and which are scheduled to be written out. The chained-scheduling appendage routine is cleverly designed to handle the possible masking of PCI's which can cause them to occur simultaneously with channel-end interrupts. The details of this design are given in the description of module IGG019CU in the Sequential Access Methods PLM (<41>); what is referred to here as the PUT routine is discussed in <41> under the heading "Chained Channel-Program Scheduling End-of-Block Routines."

From the foregoing description it can be seen that the two extreme cases of chained scheduling are (1) all joins succeed, and (2) all joins fail. In the first case, only one EXCP is issued, and all blocks are written to the file in a continuous spurt of activity by the channel. In the second case, each channel program requires an EXCP after the PUT routine notices that the join was unsuccessful. In both cases, each block written causes a PCI. As explained in section IV.A.4.b, SMF counts PCI's as EXCP's, so the number of EXCP's recorded by SMF may range from one more than the block count to twice the block count, depending on how many joins succeed. (For data sets on tape, two more EXCP's may be incurred for header and trailer label processing.)

The likelihood of a successful join decreases as the speed of the output device increases. In the limiting case of instantaneous I/O - completion, the channel-end interruption would occur long before the program had filled another buffer. On the other hand, a very slow output device might never catch up with the program, so that no join would ever fail. Note that in this case all other I/O requests on the channel being used for chained scheduling would be kept pending, and the tasks that issued them would have to wait for the entire file to be processed. Such monopolization of the channel may have a negative impact on the performance of the system as a whole, although it reduces the residency time of the chained-scheduling job. The effect of contention for resources in a multiprogramming environment can be to either increase or decrease the probability of a successful join, depending on whether the process that fills the buffers or the process that empties them is interfered with more.

The operation of chained-scheduled I/O under MVS is similar to what has just been described, except that the first CCW in each channel program no longer has the PCI bit on. Since no PCI's are issued, the appendage routine only gets control at channel-end time, and consequently cannot notify the PUT routine that buffers are available for re-use until the completion of the channel program. It follows that the number of blocks that can be written in one continuous burst by the channel is limited to the number of buffers, regardless of the speed of the device. It also follows that (apart from label processing) the minimum EXCP count is the smallest integer not less than the ratio of the

block count to the number of buffers, while the maximum EXCP count is just the block count. Thus the maximum EXCP count for MVS is less than the minimum EXCP count for OS when the same data set is processed using chained scheduling.

The hazard of channel monopolization has been removed under MVS by eliminating PCI's, but the variability of EXCP counts can still be expected to invalidate charging algorithms which rely on them. Also, since unsuccessful joins entail more CPU processing to restart the channel programs, another source of CPU-time variability has been made mandatory under MVS. Of course, the conclusions of this analysis could be invalidated by changes to future releases of MVS, which may eventually rectify these problems arising from making chained scheduling standard for sequential I/O.

Appendix H: Glossary of Acronyms

ABEND - Abnormal End of a task or job step
ACT - Accounting Control Table (<5>)
APAR - Authorized Program Analysis Report (i.e. a known bug in OS)
AVR - Automatic Volume Recognition (<5>, <7>)
BSAM - Basic Sequential Access Method (<1>)
BTAM - Basic Telecommunications Access Method (<31>)
CPU - Central Processing Unit
CSW - Channel Status Word (<21>, <17>)
CVT - Communication Vector Table (<26>)
DADSM - Direct Access Device Space Management (<1>, <10>)
DASD - Direct Access Storage Device
DCB - Data Control Block (<1>, <7>, <26>)
DDnames - Names on DD (data definition) statements in JCL
DEB - Data Extent Block (<26>)
DSB - Data Set Block (<5>)
DSCB - Data Set Control Block (<1>, <10>, <26>)
DSN - Data Set Name (<1>, <7>)
ECB - Event Control Block (<1>, <26>)
EOV - End of Volume (<1>, <11>)
ESV - Error Statistics by Volume (<2>, <3>, <12>)
EXCP - EXecute Channel Program (<1>, <12>)
GDG - Generation Data Group (<1>)
GTF - Generalized Trace Facility (<36>, <40>)
HASP - Houston Automatic Spooling Priority system
I/O - Input/Output
IBM - International Business Machines Corporation
ICB - Interruption Control Block (<26>)
IEBGENER - An IBM-supplied data set utility program (<6>)
IEHLIST - An IBM-supplied system utility program (<6>)
IOB - Input/Output Block (<26>)
IOS - Input/Output Supervisor (<12>)
IPL - Initial Program Loader (<13>)
IRB/IQE - Interruption Request Block/Interruption Queue Element, used in scheduling asynchronous exit routines (<4>)
ISAM - Indexed Sequential Access Method (<1>)
JCL - Job Control Language (<7>)
JCT - Job Control Table (<5>)
JFCB - Job File Control Block (<26>)
JMR - Job Management Record (<5>)
JSCB - Job Step Control Block (<26>)
K - Kilobyte (1024 bytes)
LCS - Large Core Storage
MICR - Magnetic Ink Character Recognition
MFT - Multiprogramming with a Fixed number of Tasks
MF/1 - Measurement Facility/1
MVT - Multiprogramming with Variable number of Tasks
NIP - Nucleus Initialization Program (<13>)
OS - Operating System; specifically, IBM System/360 Operating System
PCI - Program-Controlled Interruption (<21>, <17>)

PDS - Partitioned Data Set, or Library (<1>)
PLM - Program Logic Manual
PSW - Program Status Word (<21>, <17>)
PTF - Product Temporary Fix (an IBM-supplied patch to OS) (<34>, <36>)
QMPA - Queue Manager Parameter Area (<5>)
QSAM - Queued Sequential Access Method (<1>)
RPS - Rotational Position Sensing
SCT - Step Control Table (<5>)
SMCA - System Management Control Area (<5>, <26>)
SMF - System Management Facilities (<2>, <3>)
SMP - System Modification Program (<34>)
SQA - System Queue Area (<4>, <23>)
STAE - Specify Task Asynchronous Exit (<23>)
SVC - Supervisor Call (<4>, <17>, <21>)
Sysout - System Output (<1>, <23>)
TCB - Task Control Block (<26>)
TCT - Timing Control Table (<5>, <26>)
TCTIOT - Timing Control Task Input/Output Table (<5>, <26>)
TIOT - Task Input/Output Table (<26>)
TQE - Timer Queue Element (<4>)
TTR - Relative Track and Record (<1>)
UCB - Unit Control Block (<26>)
VTOC - Volume Table of Contents (<1>)

Appendix I: Bibliography

<1> OS Data Management Services Guide (GC26-3746).

<2> IBM System/360 Operating System: System Management Facilities (GC28-6712).

<3> OS/VS System Management Facilities (SMF) (GC35-0004).

<4> OS/360 MVT Supervisor Program Logic Manual (GY28-6659).

<5> OS/360 MVT Job Management Program Logic Manual (GY28-6660).

<6> IBM System/360 Operating System: Utilities (GC28-6586).

<7> IBM System/360 Operating System: Job Control Language Reference (GC28-6704).

<8> Johnson, R.H. "SLAC Changes to the SMF Data Dumping Mechanism of OS," SHARE Computer Measurement and Evaluation, Vol. II (1974).

<9> OS System Generation (GC28-6554).

<10> OS DADSM Logic (GY28-6607).

<11> OS Open/Close/EOV Logic (GY28-6609).

<12> IBM System/360 Operating System: Input/Output Supervisor Program Logic Manual (GY28-6616).

<13> IBM System/360 Operating System: Initial Program Loader and Nucleus Initialization Program (GY28-6661).

<14> Betz, R.E. "Use of SMF Data for Performance Analysis and Resource Accounting on IBM Large-Scale Computers." In Computer Performance Evaluation, NBS Special Pub. 401 (1974).

<15> Graves, J.M. "Using SMF and TFLOW for Performance Enhancement." In Computer Performance Evaluation, NBS Special Pub. 401 (1974).

<16> Leavitt, Don. "Three Users Find SMF Inadequate in Measuring Performance." ComputerWorld, Vol. IX, No. 39 (Sept. 24, 1975).

<17> IBM System/370 Principles of Operation (GA22-7000).

<18> SyncSort: A Hardware Monitor Evaluation, Whitlow Computer Systems, Inc., 222 S. Marginal Rd., Fort Lee, NJ 07024 (1975).

<19> Drummond, M.E. "A Perspective on System Performance Evaluation." IBM Systems Journal 8, No. 4, 252-263 (1969).

<20> Buchholz, W. "A Synthetic Job for Measuring System Performance." IBM Systems Journal 8, No. 4, 309-318 (1969).

<21> IBM System/360 Principles of Operation (GA22-6821).

<22> IBM System/360 Model 65: Functional Characteristics (GA22-6884).

<23> IBM System/360 Operating System: MVT Guide (GC28-6720).

<24> IBM System/360 Model 65: Operating Procedures (GA27-2728).

<25> IBM System/360 Operating System: Model 65 Shared Main Storage, Multiprocessing, Preliminary Description (GC28-6671).

<26> IBM System/360 Operating System: System Control Blocks (GC28-6628).

<27> System Management Reporting System, User Reference Manual, Tesseract Corp., 101 Howard St., San Francisco, CA 94120 (1972).

<28> SMF Selectable Analyzer, Field Developed Program Number 5798-AAR (SB21-0047).

<29> Feller, William, An Introduction to Probability Theory and Its Applications, Vol. I (Second Edition), John Wiley and Sons, New York, 1957.

<30> IBM System/360 Operating System: PL/I (F) Language Reference (GC28-8201).

<31> IBM System/360 Operating System: Basic Telecommunications Access Method (GC30-2004).

<32> System Utilization Monitor User's Manual, Tesdata Systems Corp., Measurement Systems Division , 7900 Westpark Dr., McLean, VA 22101.

<33> Sensor Point Reference Manual, Tesdata Systems Corporation, Measurement Systems Division, 7900 Westpark Dr., McLean, VA 22101.

<34> OS System Modification Program (SMP) (GC28-6791).

<35> Currah, Brian. "Some Causes of Variability in CPU Time." SHARE Computer Measurement and Evaluation, Vol. III (December 1973 - March 1975) pp. 389-392.

<36> IBM System/360 Operating System: Service Aids (GC28-6719).

<37> Currah, Brian, and Mario Morino. "What to Do With SMF Data," SHARE Computer Measurement and Evaluation, Vol. II (November 1971 - December 1973) pp. 20-65.

<38> Chung, S.Y. "SMF Survey," SHARE Computer Measurement and Evaluation, Vol. I, No. 12 (July 1971) pp. 87-91.

<39> Cogger, Alison. "TESTEXIT Critique," SHARE Computer Measurement and Evaluation, Vol. I, No. 7 (July 1970) pp. 7-8.

<40> Obergsfell, George. "Using GTF as a Performance Improvement Tool," SHARE Computer Measurement and Evaluation, Vol. III (December 1973 - March 1975) pp. 105 - 114.

<41> IBM System/360 Operating System: Sequential Access Methods Program Logic Manual (GY28-6604).

NBS-114A (REV. 11-77)

U.S. DEPT. OF COMM. BIBLIOGRAPHIC DATA SHEET	1. PUBLICATION OR REPORT NO. NBS SP 500-40	2. Gov't Accession No.	3. Recipient's Accession No.

4. TITLE AND SUBTITLE

COMPUTER SCIENCE & TECHNOLOGY

Guideline on Major Job Accounting Systems: The System Management Facilities (SMF) for IBM Systems Under OS/MVT

5. Publication Date: October 1978

6. Performing Organization Code

7. AUTHOR(S): Gary Durbin, Todd Kinney, Peter Lamasney, Edward Newman, and Edward Syrett of the Tesseract Corporation

8. Performing Organ. Report No.

9. PERFORMING ORGANIZATION NAME AND ADDRESS

NATIONAL BUREAU OF STANDARDS
DEPARTMENT OF COMMERCE
WASHINGTON, D.C. 20234

10. Project/Task/Work Unit No.

11. Contract/Grant No.

12. Sponsoring Organization Name and Complete Address (*Street, City, State, ZIP*)

Institute for Computer Sciences and Technology,
National Bureau of Standards

13. Type of Report & Period Covered

14. Sponsoring Agency Code

15. SUPPLEMENTARY NOTES

Library of Congress Catalog Card Number 78-600113

16. ABSTRACT (*A 200-word or less factual summary of most significant information. If document includes a significant bibliography or literature survey, mention it here.*)

This document reports the results of a study which was commissioned in response to the need for a better understanding of how job accounting systems work, what they measure, and how accurately they measure. The accounting system described is IBM's System Management Facilities (SMF) for 360/370 environments operating under OS/MVT. Considerable detail is provided on SMF's activity in collecting the data necessary to account for resource utilization by the individual jobs in a multiprogramming system and to provide some indicators of the performance of the system itself. Included within the scope of this study was an investigation of the accuracy, both absolute and relative, of SMF as a measurement tool and the costs entailed in the use of SMF. The experimental methodology used to explore these questions is summarized in the report, as are the conclusions reached. Performance analysts, system programmers, and users interested in optimal use of their computer resources will find much help here in the application of a valuable measurement tool.

17. KEY WORDS (*six to twelve entries; alphabetical order; capitalize only the first letter of the first key word unless a proper name; separated by semicolons*) Computer performance analysis; computer performance measurement; CPU time; EXCP counts; job accounting systems; IBM OS/MVT; resource utilization measurement; System Management Facilities; system wait time

18. AVAILABILITY

[X] Unlimited

[ ] For Official Distribution. Do Not Release to NTIS

[X] Order From Sup. of Doc., U.S. Government Printing Office Washington, D.C. 20402, SD Stock No. SN003-003-01989-5

[ ] Order From National Technical Information Service (NTIS) Springfield, Virginia 22151

19. SECURITY CLASS (THIS REPORT): UNCLASSIFIED

21. NO. OF PAGES: 170

20. SECURITY CLASS (THIS PAGE): UNCLASSIFIED

22. Price: $4.00

USCOMM-DC 66035-P78

ANNOUNCEMENT OF NEW PUBLICATIONS ON COMPUTER SCIENCE & TECHNOLOGY

Superintendent of Documents,
Government Printing Office,
Washington, D. C. 20402

Dear Sir:

Please add my name to the announcement list of new publications to be issued in the series: National Bureau of Standards Special Publication 500-.

Name _____

Company _____

Address _____

City _____ State _____ Zip Code _____

(Notification key N-503)

☆ U. S. GOVERNMENT PRINTING OFFICE : 1978 O - 275-073

U.S. DEPARTMENT OF COMMERCE
National Bureau of Standards
Washington, D.C. 20234

OFFICIAL BUSINESS

Penalty for Private Use, $300

POSTAGE AND FEES PAID
U.S. DEPARTMENT OF COMMERCE
COM-215

U.S.MAIL

SPECIAL FOURTH-CLASS RATE
BOOK