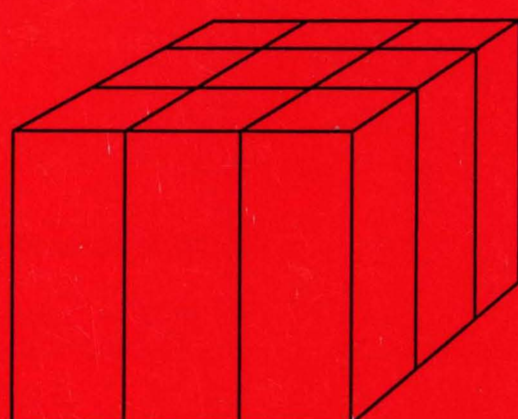
IBM

VSE/Advanced Functions

Service Aids



VSE/Advanced Functions

Service Aids

Program Number 5666-301

Order Number SC33-6195-1
File No. S370/4300-40

Second Edition (March 1985)

This edition is a major revision of SC33-6195-0, an edition which IBM has not made generally available. This edition applies to Version 2, Release 1 of IBM Virtual Storage Extended/Advanced Functions, Program Number 5666-301 and to all subsequent releases until otherwise indicated in new editions or Technical Newsletters. Changes are made periodically to the information herein; before using this publication in connection with the operation of IBM systems, consult the latest IBM System/370, 30XX and 4300 Processors Bibliography, GC20-0001, for the editions that are applicable and current.

References in this publication to IBM products, programs, or services do not imply that IBM intends to make these available in all countries in which IBM operates. Any reference to an IBM program product in this document is not intended to state or imply that only IBM's program product may be used. Any functionally equivalent program may be used instead.

Publications are not stocked at the addresses given below; requests for copies of IBM publications should be made to your IBM representative or to the IBM branch office serving your locality.

A form for readers' comments is provided at the back of this publication. If the form has been removed, comments may be addressed either to:

IBM Corporation
Dept. 6R1
180 Kost Road
Mechanicsburg, PA 17055, USA

or to:

IBM Deutschland GmbH
Dept. 3248
Schoenaicher Strasse 220
D-7030 Boeblingen, Federal Republic of Germany

IBM may use or distribute whatever information you supply in any way it believes appropriate without incurring any obligation to you.

PREFACE

This publication tells you how to use the diagnosis and service aids under the VSE System. These service aids are referred to in VSE diagnosis publications, VSE/System Package Diagnosis Guide, SC33-6182.

Where the aids are interactive, you will find examples of the panels and the interactive dialogs that are used to invoke and run the programs. For running in batch mode, you will find examples of job control streams. If an explanation of the output is necessary, sample listings are included.

Readers of this publication should be familiar with the operational concept of the VSE System.

VSE/Advanced Functions includes service aids that help you in information gathering and problem diagnosis when a system or program malfunction occurs.

This publication describes the use of these service aids. When to use the services and under what circumstances is explained in VSE/System Package Diagnosis Guide.

> Restriction: If any of these service aids writes the output to a 3211 printer and this printer's indexing feature is being used, a number of characters may get lost on each line of the output. The system's dump and trace routines, for example, write output records of 120 bytes in length.
>
> To avoid the loss of data, you should load another FCB (forms control buffer) image which disables the indexing feature before requesting the desired printout. For information on FCB loading, see VSE/Advanced Functions System Control Statements.

The conventions for showing the format of commands and statements used in publication VSE/Advanced Functions System Control Statements apply also to this manual.

This publication is divided into four parts:

- **Dumps of Virtual Storage**

  Which describes the various dump functions in general and shows the file and library environment which is needed to store dumps. The methods to request and to print storage dumps which have been stored on tape or in a dump sublibrary are described. The description of how to process the dumps is shown for batch mode only. This part contains also the description of the DOSVSDMP utility and the Stand-alone dump analysis routine IJBXDBUG.

- **SDAID Trace**

  Contains an overview of the SDAID trace program, describes all trace types and the various methods to initialize them. How you can start stop, or terminate the initialized traces is also described in this part.

- **Info/Analysis**

  Info/Analysis is the dump viewing and management facility of VSE/Advanced Functions Version 2 Release 1. This part emphasizes the use of Info/Analaysis in interactive mode using the Interactive System Productivity Facility (ISPF), but gives also batch mode commands reference information.

- **Appendixes**

  Contain a message/subcomponent cross-reference list, a list of all supervisor call codes, a description of the symptom record, various display and list aids such as the LVTOC or the LSERV program, and tells how to use some hardware diagnosis aids.

Please find a summary of the contents of this manual on the next page. A detailed table of contents is given on page xi and on the second page of each chapter.

PART 1. DUMPS OF VIRTUAL STORAGE

- Chapter 1, "Dumps General Description"
- Chapter 2, "Requesting a Dump"
- Chapter 3, "Printing the Stored Dump"
- Chapter 4, "Dump Utilities"

PART 2. SDAID TRACE

- Chapter 5, "SDAID Overview"
- Chapter 6, "SDAID General Description"
- Chapter 7, "Initialize an SDAID Trace in Direct Input Mode"
- Chapter 8, "Initialize an SDAID Trace via a Procedure"
- Chapter 9, "Initialize a Trace in Prompt Input Mode"
- Chapter 10, "Start/Stop and End the Trace"

PART 3. INFO/ANALYSIS

- Chapter 11, "Introduction"
- Chapter 12, "Interacting with Info/Analysis"
- Chapter 13, "Dump Management"
- Chapter 14, "Dump Symptoms"
- Chapter 15, "Dump Viewing"
- Chapter 16, "Dump Offload and Onload"
- Chapter 17, "Interactive Commands"
- Chapter 18, "Example Interactive Sessions"
- Chapter 19, "Info/Analysis in Batch Mode"

PART 4. APPENDIXES

- Appendix A, "Message/Subcomponent Cross-Reference List"
- Appendix B, "Supervisor Call Code (SVC) List"
- Appendix C, "Symptom Record Overview"
- Appendix D, "Other Service Aids"
- Appendix E, "Hardware Service Aids"

Related Publications

You will need the following publications when diagnosing a problem:

- VSE/System Package Diagnosis: Guide, SC33-6182.
- VSE/System Package Messages and Codes, SC33-6181.
- VSE/Advanced Functions, Operation, SC33-6194.
- System /370 Reference Summary, GX20-1850.
- OS/VS,DOS/VSE,VM/370 EREP, GC28-1378
- ACF/VTAM Diagnostic Guide, SC27-0615
- Device Support Facilities, User's Guide and Reference, GC35-0033

Interactive Info/Analysis users should be familiar with how to use dialogs running with the Interactive System Productivity Facility (ISPF).

- Interactive System Productivity Facility Dialog Management Service, SC34-2088.
- Interactive System Productivity Facility Installation and Customization, SC34-2080.
- VSE/Interactive Computing and Control Facility Terminal Users Guide, SC33-6204.

The following publications are also referred to:

- VSE/Advanced Functions, System Management Guide, SC33-6191.
- VSE/Advanced Functions, System Control Statements, SC33-6198.
- VSE/Advanced Functions, Application Programming: Macro User's Guide, SC33-6196.
- VSE/Advanced Functions, Application Programming: Macro Reference, SC33-6197.
- VSE/Online Test Executive Program (OLTEP): Installation and Operation, GC33-6156.
- Using VSE/VSAM Commands and Macros, SC24-5144.
- VSE/Advanced Functions, Planning and Installation, SC33-6193.
- VSE/Advanced Functions, System Utilities, SC33-6100.

- VSE/DITTO, General Information, GH19-6072.

- VSE/Advanced Functions Maintain System History Program Reference, SC33-6199.

For other publications refer to IBM System/370, 30xx and 4300 Processors Bibliography, GC20-0001.

SUMMARY OF AMENDMENTS

This edition, a major revision of the publication VSE/Advanced Functions: Service Aids SC33-6195 contains new information about:

- Virtual Address Extension
- Dump handling with Info/Analysis
- SDAID trace initialization

For a complete overview of functions new with Version 2 Release 1 of VSE/Advanced Functions, refer to VSE/Advanced Functions, Planning and Installation, SC33-6193.

CONTENTS

Part 1. Dumps of Virtual Storage

FIGURES

PART 1. DUMPS OF VIRTUAL STORAGE

You may face system conditions in which you want to know what the contents of your system's storage is. For this, the storage data can be read out, saved in a library or on a tape, or can be printed on SYSLST. This processed storage data is called a **dump**.

```
┌───────────┐
│           │
│           │    ┌────────┐
│           │    │        │ ───> Printer
│  Storage  │ -> │  Dump  │ ───> Tape
│           │    │        │ ───> Library
│           │    └────────┘
│           │
└───────────┘
```

This part of the manual describes how to retrieve a dump and how to use the saved dump for problem determination.

A dump can be written to a dump sublibrary, a tape, or can be printed directly on SYSLST.

It depends on the error situation which of the shown methods you use to retrieve information for problem determination. For example, in case of a system wait or a system loop the Stand-alone dump program would be the appropriate tool to save or print the storage contents.

This part is divided into four sections as shown on the next page.

Dumps General Description Page: 1-1	• Chapter 1, Dumps General Description describes the various types of dumps, shows an overview of the contents of a dump and gives information on the used dump libraries.
Requesting a Dump Page: 2-1	• Chapter 2, Requesting a Dump shows the commands or options to retrieve dumps.
Printing the Stored Dump Page: 3-1	• Chapter 3, Printing the Stored Dump describes the methods to print dumps stored in the dump sublibraries or on a tape.
Dump Utilities Page: 4-1	• Chapter 4, Dump Utilities describes the DOSVSDMP utility and the Stand-alone dump analysis routine IJBXDBUG.

Figure 1-1. Dump Part Overview

CHAPTER 1. DUMPS GENERAL DESCRIPTION

```
 ┌───────────┐    ┌───────────┐    ┌───────────┐
 │   Dumps   │    │           │    │           │
>│  General  │<   │ Requesting│    │Printing the│
>│Description│<   │  a Dump   │    │ Stored Dump│
 │           │    │           │    │           │
 └───────────┘    └───────────┘    └───────────┘
   Chapter 1        Chapter 2        Chapter 3

                  ┌───────────┐
                  │           │
                  │   Dump    │
                  │ Utilities │
                  │           │
                  └───────────┘
                    Chapter 4
```

This chapter describes the various types of dumps and the functions you use to create dumps in general.

A dump can be written to a tape, can be printed on SYSLST or can be saved in a dump sublibrary. How you can define these dump sublibraries is also explained in this chapter.

Contents

TYPES OF DUMPS

You will find a general description of the following types of dumps in this chapter:

- The **ABEND dump** which is initiated by the system (through the system's ABEND handling) or the operator (through the CANCEL command), controlled by the operator.

- The **DUMP command dump**,
- **Stand-alone dump**, and
- **SDAID dump**

which are initiated by the operator.

- The **Macro dump**

which is initiated by the programmer.

Dump Contents Overview

The output of the functions:

DUMP command, ABEND dump, CANCEL command which initiates an ABEND dump, and Stand-alone dump program

contains two major parts.

- The symptom records.
- The data records.

The amount of information which is stored in these dump records depends on the function which requests the dump.

Figure 1-1 gives an overview of a dump, which can reside either in a dump sublibrary or on a dump tape.

```
                ┌─────────────────────────┐
                │ - Environment Information │
SYMPTOM RECORDS │ - Error Symptoms          │
                │ - Control Block and       │
                │   Data Area Descriptors   │
                ├─────────────────────────┤
                │                           │
DATA RECORDS    │ - Dumped Storage          │
                │                           │
                │                           │
                └─────────────────────────┘
```

Figure 1-1. Overview: Dump Contents

The symptom records are built by the component which produces the dump. It contains information to format the dump data later on.

Please find a description of the symptom records in Appendix C, "Symptom Record Overview" on page C-1.

THE ABEND DUMP FUNCTION

The ABEND dump function is internally called when the VSE/Advanced Functions system detects an ABEND condition or when a CANCEL command has been given.

The function is described in this sections under the following headings:

- "ABEND Dump Function, Overview"

More detailed information is shown under:

- "When Is the ABEND Dump Function Activated" on page 1-7.
- "What Are the OPTIONs to Control the ABEND Dump" on page 1-7.
- "OPTIONS to Control the Dump Contents" on page 1-7.
- "Options to Control the Output Destination" on page 1-8.
- "Contents of the ABEND Dump Output" on page 1-9.

ABEND Dump Function, Overview

WHAT IS AN ABEND: ABEND stands for ABnormal END of task. This means that a program (task) is terminated prior to its completion because of an error that could not be resolved by system recovery facilities.

WHAT IS AN ABEND DUMP: The system's ABEND dump function is called by VSE/Advanced Functions system either at the occurrence of an ABEND or else if a CANCEL command has been issued. Once the function has been called, a dump of virtual storage is provided.

In 370 mode, with a program executing in real mode, the real address area allocated to a partition is included. You can control the ABEND dump function via Job Control Options (// OPTION or STDOPT).

Figure 1-2 on page 1-6 shows that:

- The ABEND dump function is activated when an ABEND condition occurs.
- The function is controlled by Job Control OPTIONs submitted by STDOPT, OPTION, or CANCEL statements.
- These OPTIONs define
  - the contents of the dump
  - to which I/O device the dump is moved.

```
 ┌──────────────────┐
 │      ABEND       │
 │    OCCURRENCE    │
 └──────────────────┘
          |
          V
 ┌──────────────────┐
 │      ABEND       │
 │  DUMP FUNCTION   │
 └──────────────────┘
          |
          V
 ┌──────────────────┐
 │     OPTIONS      │
 │   CONTROLLING    │
 └──────────────────┘
     |          |
     V          V
┌───────────┐ ┌─────────────┐
│  OUTPUT   │ │  OUTPUT     │
│ CONTENTS  │ │  DEVICE     │
│───────────│ │─────────────│
│           │ │             │
│= none     │ │= Sublibrary │
│= System or│ │= or SYSLST  │
│= Partition│ └─────────────┘
│   Dump    │
└───────────┘
```

Figure 1-2. Overview: The ABEND Dump Function

Two kinds of ABEND dumps are shown in Figure 1-2 under 'OUTPUT CONTENTS': System dump, Partition dump.

- The **System-Dump** dumps the whole supervisor area and the dump symptoms besides the partition area.

- The **Partition-Dump** includes only selected VSE/Advanced Functions control blocks and the dump symptoms in addition to the partition area.

The output device for the dump data is a dump sublibrary or a printer device assigned to SYSLST.

When Is the ABEND Dump Function Activated

The ABEND dump function is activated when

- A program or task running in one of the system's partitions comes to an **ABnormal END**, and **no AB exit** routine is active. See

  =====> "OPTIONS Controlling the ABEND Dump Function" on page 2-4.

- A **CANCEL command** is issued by the operator for one of the operating system's partitions. See

  =====> "Requesting a Dump by the CANCEL Command" on page 2-5.

If the activation of the ABEND dump function leads to a dump writing operation (depending on the active job control options), the storage contents are dumped

- Before any end-of-job routine is executed.
- Before any of the attached subtasks is terminated.

What Are the OPTIONs to Control the ABEND Dump

Using the job control options shown below you can define whether you want the creation of a dump to be suppressed, or which kind of dump you want to be created, and whether you want the dump to be stored in a dump sublibrary or printed on a particular output device.

[//] **STDOPT** [DUMP=YES|DUMP=PART|DUMP=NO][,SYSDUMP=YES]

[//] **OPTION** [DUMP|PARTDUMP|NODUMP][,SYSDUMP]

<u>OPTIONS to Control the Dump Contents</u>

```
[//] STDOPT [DUMP=YES|DUMP=PART|DUMP=NO]
[//] OPTION [DUMP|PARTDUMP|NODUMP]
```

The options controlling the dump contents can be set with the STDOPT command or the OPTION statement.

STDOPT DUMP=YES Requests a system-dump.

STDOPT DUMP=PART Requests a partition-dump.

STDOPT DUMP=NO Specifies that you want to suppress the dump creation.

Options specified in the STDOPT command are in effect until the next IPL or the next // STDOPT statement, but they may be overridden for a job by an OPTION specification.

OPTION DUMP Requests a system-dump.

OPTION PARTDUMP Requests a partition-dump.

OPTION NODUMP Specifies that you want to suppress the dump output.

A dump option specified in an OPTION statement overrides the dump option specified in an STDOPT command. The option is in effect only for the job to which it belongs.

> **Note:** You will not get any dump output if you use the STXIT PC or STXIT AB macro, even if you include the DUMP or PARTDUMP option.

Options to Control the Output Destination

```
[//] STDOPT SYSDUMP=YES
[//] OPTION SYSDUMP
```

The output of the ABEND dump is either written into the dump sublibrary for the partition or it is printed on SYSLST.

The ABEND dump function writes the output to a **DUMP SUBLIBRARY** if the:

- **Dump Library** (named SYSDUMP) and appropriate **sublibrary** has been created.
- **LIBDEF statement** for the dump sublibrary has been submitted (usually during the ASI procedure for the partition).
- **Job Control option**

  STDOPT SYSDUMP=YES or OPTION SYSDUMP has been specified.

- Associated dump library is **not full**.

If one of the above is not true the dump is printed on **SYSLST**.

> **Note:** The dump is lost if it cannot go to SYSDUMP, and SYSLST has not been assigned.

How the SYSDUMP library can be defined is described under

"How to Establish the Dump Sublibraries" on page 1-20.

For a description of how to print the ABEND dump from a dump sublibrary see

"Print Dump Information" on page 3-16.

Contents of the ABEND Dump Output

The output of the ABEND dump function either in the sublibrary or on SYSLST contains:

- A dump symptom part which is always included.
- A part which is either a system-dump or a partition-dump, depending on the options active at the time the dump was taken.

```
┌──────────────────┐
│   SYMPTOM PART   │
├──────────────────┤
│                  │
│   SYSTEM DUMP    │
│       or         │
│ PARTITION  DUMP  │
│                  │
└──────────────────┘
```

Symptom Part of the ABEND Dump

This part of the output contains

- Control data from the symptom records, like information about the environment or the failure. For a description of the symptom records see Appendix C, "Symptom Record Overview" on page C-1.

System-Dump Contents

The system dump which is produced if OPTION DUMP or STDOPT DUMP=YES is active contains the following information besides the symptom part:

- The ending task PSW, general purpose and floating point registers.
- The entire supervisor area.
- The addresses of the control blocks listed in Figure 1-3 on page 1-11.
- The allocated portion(s) of the system GETVIS area.
- If the error occurred in the SVA, that part of the SVA which holds the phase responsible for the ABEND.
- The partition for which the ABEND dump function is active including areas acquired dynamically within the partition by GETVIS macros in your program.

Partition-Dump Contents

A partition dump is produced when option OPTION PARTDUMP or STDOPT DUMP=PART is active. The dump output includes the following VSE/Advanced Functions system areas besides the symptom part.

- The ending task PSW, general purpose and floating point registers.
- The LOWCORE (low address storage).
- The areas containing VSE/Advanced Functions control blocks listed in Figure 1-3 on page 1-11.
- The partition for which the ABEND dump function is active including areas acquired dynamically within the partition by GETVIS macros in the program.
- If the error occurred in the SVA, that part of the SVA which holds the phase responsible for the ABEND.
- The logical transient area (LTA), if the error causing the dump to be taken occurred in a task owning the LTA.

COMREG	Partition's communication region
SYSCOM	System communication region
PUBTAB	Physical unit block
PUBOWN	PUB owner
PUB2TAB	Physical unit block extension
LUBTAB	Partition's logical unit block
LUBEXT	Partition's LUB table extension
DIBTAB	Partition's disk information block
DIBEXT	Partition's disk information block extension
PIBTAB	Partition information block
PIB2TAB	Partition information block extension
PCB	Partition control block
SYSFIL	System file buffer
TIB	Task information block
TCB	Task control block
LOADLS	Partition's phase load trace table
ICCFVT	ICCF vector table
LPT	Library pointer table
LDT	Library definition table
SDT	Sublibrary definition table
EDT	Extent definition table
DDT	Device definition table

Figure 1-3. VSE/Advanced Functions Control Blocks

THE DUMP COMMAND DUMP

You can request a dump of parts of or of all the virtual storage with the attention routine command DUMP.

Please see the DUMP command description under

=====> "Requesting a Dump by the DUMP Command" on page 2-6.

THE STAND-ALONE DUMP

You can request a dump of the applicable VSE/Advanced Functions virtual storage (a Stand-alone dump) with the Stand-alone dump program. The program can be called from cards, diskette or tape.

A description of how to request this dump is given under

=====> "Requesting a Stand-Alone Dump" on page 2-7.

The following sections give a description of "The Stand-Alone Dump Program" on page 1-12 and the "Execution Dependent Output of the Stand-Alone Dump Program" on page 1-12.

The Stand-Alone Dump Program

If your system entered a hard or soft wait state or is in a continuous loop, no normal system operation is possible. In this case you can invoke the Stand-alone dump program to get information about the problem.

The Stand-alone dump program dumps the applicable VSE/Advanced Functions system's virtual storage. You have to perform an IPL of your VSE/Advanced Functions system after the execution of the Stand-alone dump program has been ended.

Before you can dump the entire system and program storage, the Stand-alone dump program has to be created with the DOSVSDMP utility. See "Generate the Stand-Alone Dump Program" on page 4-3.

The creation of the Stand-alone dump program should be done shortly after system installation in order to have the tape, the diskette or cards available in case of a system error.

Execution Dependent Output of the Stand-Alone Dump Program

The format of the dump output depends on whether

- The dump has been stored on a tape volume.
- The Stand-alone dump program has been executed from cards or diskette which implies that its output is directed to a printer.
- The VSE/Advanced Functions system was executing in 370 mode or ECPS:VSE mode.

Stand-Alone Dump Program Output on Tape

The Stand-alone dump program stores the dump information on the tape from where it has been loaded. The dump information on tape contains

- The symptom record which holds information on the hardware and software environment, error symptoms and control block locators.
- The dump data, which are retrieved pages from processor storage, or from the page data set.
- The last 200 messages from the hard copy file.

You can print the dump output either with the Info/Analysis program or with the DOSVSDMP utility.

A description of how to print the Stand-alone dump program output can be found under "Print Dumps Stored on Tape" on page 3-25.

Stand-Alone Dump Program Output on Printer (Executed from Diskette or Cards)

If you have used a diskette or cards instead of a tape volume to start the Stand-alone dump program from, the program's output is directed to a printer. The contents of the printed output differs according to the mode in which the system is running.

Execution in 370 Mode

The program first prints parts of the symptom record with status information followed by the pages of virtual storage in ascending sequence. The contents of storage locations X'00' through X'18' are not included in the dump output.

Execution in ECPS:VSE Mode

The program prints the first eight pages of virtual storage as follows:

- Locations '0800' through '3FFF'
- Locations '0000' through '07FF'
- Parts of the symptom record and the machine status information.
- The remaining pages are printed in ascending sequence.

  If you did not save the status information, no status information is provided by the program, and the output consists of the pages that are in processor storage at this point in time.

THE SDAID DUMP

The SDAID program can also be used to dump virtual storage. You may use this program for example if you need a dump of a certain part of storage at a defined event.

Please find a short description of this SDAID function under

=====> "Requesting a Dump on Event (SDAID Dump)" on page 2-9.

DUMP REQUESTED BY MACROS

A dump of virtual storage can also be requested through dump macros.

You can find a short description of this kind of requesting a dump under

=====> "Requesting a Dump from a Program" on page 2-10.

INFO/ANALYSIS

Info/Analysis, is a component of VSE/Advanced Functions Version 2 Release 1. It is a tool to:

- Manage the dump files
- Print or display dump information.

With Info/Analysis, you can simplify the task of using dump data to solve software problems. Info/Analysis assists you in this task through the following functions:

- Dump Management - to list the dumps being managed by Info/Analysis, to add or delete dumps from that list, and to delete dumps from the system.

- Dump Symptoms - to display problem failure information collected by the dumping component and by subsequent analysis routines.

- Dump Viewing - to display dump data in hexadecimal and character format, to locate, format, and display control blocks and other dump data that may be pertinent to the problem, to invoke dump analysis routines, and to display the results of those routines.

- Dump Offload - to copy a dump to tape for later retrieval.

- Dump Onload - to copy a dump to a dump sublibrary (a Stand-alone dump for example).

You may use Info/Analysis in either **interactive** (if you have the interactive system productivity facility, ISPF installed) or **batch** mode. All major functions are available in both modes; however, Dump Viewing has more flexibility in interactive mode while it is preferable to use Dump Offload and Dump Onload in batch mode.

The following figure shows an overview of the various modes the Info/Analysis program may be used.

The Info/Analysis Execution Modes

BATCH MODE	INTERACTIVE (ISPF) MODE

Input via:

- **ICCF Terminal**

  Please find an example of a procedure to start Info/Analysis under "Execute Info/Analysis via an ICCF Terminal" on page 1-17.

  Output: ICCF Terminal

- **SYSIN/SYSLOG**

  Find an example of a job to invoke Info/Analysis under Figure 3-1 on page 3-3.

  Output: SYSLST

For more information on Info/Analysis in batch mode refer to:

- Chapter 3, "Printing the Stored Dump" on page 3-1
- "Control Statement Summary" on page 19-20
- Chapter 19, "Info/Analysis in Batch Mode" on page 19-1

Input via:

- **ISPF/ICCF Terminals**

  You initiate Info/Analysis by calling the ISPF start-up procedure. Enter the name of the procedure on the ICCF terminal command line.

  Find an example of such an procedure under

  Figure 12-2 on page 12-6

  Output: ISPF/ICCF Terminal

  For more information on Info/Analysis in interactive mode refer to:

  "Part 3. Info/Analysis."

Execute Info/Analysis via an ICCF Terminal

The following example shows an ICCF procedure to invoke Info/Analysis.

```
/LOAD INFOANA
/OPTION GETVIS=P-400
---------------------------
Note: The partition must have a size of at least 768K bytes.
```

The following example shows how you can start Info/Analysis via an ICCF terminal. It shows the procedure invocation (INFOANA) and the subsequent Info/Analysis prompting messages.

```
--> INFOANA procedure name entered in command mode
    ...
    ...
    BLN1008I INFO/ANALYSIS READY
    BLN1005D ENTER CONTROL STATEMENTS FOR SELECTION LEVEL
    *ENTER DATA
--> SELECT ........   select an Info/Analysis function
```

THE LIBRARY AND FILE ENVIRONMENT

Various files are used to process and evaluate dumps stored either on a tape volume or in a dump sublibrary.

The Library and Files Required to Process Dumps

The libraries and files required to process and use dump information are:

1. The dump sublibraries (in the library SYSDUMP)

2. The dump management file (for Info/Analysis)

3. The external routines file (for Info/Analysis).

The SYSDUMP Sublibraries

The system uses the defined dump sublibraries to store dumps for later processing. These sublibraries are also used to onload dumps which have been stored on tape either by the system's dump functions or by a previous Info/Analysis offload operation.

Dumps can be processed using the print, view, analyze and management functions of the Info/Analysis program once they have been onloaded into a dump sublibrary. How the SYSDUMP library is defined and used is described in the following section. Please find a summary of the contents of this section below.

This section gives more details on the dump library and dump sublibraries under the following headings:

- "What Is the SYSDUMP Library Used for" on page 1-19.

- "How to Establish the Dump Sublibraries" on page 1-20, which discusses:

  - "What Label Information Is Needed for SYSDUMP" on page 1-20.
  - "How to Define the Library" on page 1-20.

- "What Is the Required LIBDEF Statement" on page 1-21.

- "What Are the Required Options for Dump Writing" on page 1-21.

- "How to Identify the Stored Dumps" on page 1-22.

- "How to Handle a Dump Library Full Condition" on page 1-23.

What Is the SYSDUMP Library Used for

The library named SYSDUMP is used to store the various dump types for further processing. It contains one or more dump sublibraries. Each dump sublibrary should be assigned to one partition and may contain one or more dumps. Figure 1-4 gives an overview of the SYSDUMP library concept.

```
                          SYSDUMP LIBRARY
                    ┌──────────────────────┐
                |──>| SYSDUMP.BG.....      | <──┐
                |   |──────────────────────|    |
Dump            |──>| SYSDUMP.F1.....      |    |
sublibraries:   |   |──────────────────────|    |
one per         |──>| SYSDUMP.F2.....      |    |
partition       |   |──────────────────────|    |
(if created)    |──>| ......               |    |
                |   |──────────────────────|    |
                |──>| ......               |    |
                    └──────────────────────┘    |
                                                |
                     DUMP SUBLIBRARY FOR BG     |
                    ┌──────────────────────┐ ─┐ |
Dumps stored    |──>| SYSDUMP.BG.00000020  |  | |
in one dump     |──>| SYSDUMP.BG.DUMP      |  |─┘
sublibrary      |──>|........              |  |
                    └──────────────────────┘ ─┘
```

Figure 1-4. Overview: The SYSDUMP Library Contents

These dump sublibraries are used by the system and by you:

VSE/Advanced Functions stores dumps for later processing from

- ABEND events
- CANCEL commands.

You can use the dump sublibraries to onload dumps which have been stored on tape in order to process them with Info/Analysis functions. You may use the dump sublibraries to store the following information which resides on tape:

- DUMP command dump
- Stand-alone dump
- A dump which has been off-loaded to tape.

How to Establish the Dump Sublibraries

Before dumps can be stored, the dump sublibraries have to be created. The following describes, what job control label information is required for the SYSDUMP library and how the dump sublibraries can be defined via the librarian program LIBR.

The following requirements have to be met if you want to use the dump library and its sublibraries:

1. The DLBL and EXTENT labels for the library SYSDUMP have to be specified.

2. The library SYSDUMP and its sublibraries have to be defined with the LIBR program.

3. LIBDEF statements have to be given.

4. The SYSDUMP option has to be set in order to get ABEND dumps written into the dump sublibraries.

WHAT LABEL INFORMATION IS NEEDED FOR SYSDUMP: Figure 1-5 shows an example of the label and extent information you have to submit if you want to define the dump library SYSDUMP.

The standard label area should be used to store this information.

```
   ... ... ...

// DLBL SYSDUMP,'VSE.DUMP.FILE',99/365,,DSF
// EXTENT SYS010,,1,0,5472,644

   ... ... ...
```

Figure 1-5. Example: Labels for the SYSDUMP Library

> **Note:** IBM recommends securing the dump library. Securing the dump library prevents overwriting of part of the file(s) as a result of a faulty response to an OVERLAPPING EXTENT message. For information about using the access control function, see VSE/Advanced Functions Planning and Installation.

HOW TO DEFINE THE LIBRARY: You define the dump library (normally named SYSDUMP) with the LIBR program shortly after system installation. Figure 1-6 shows an example of such a definition.

```
// JOB DEFINE
// EXEC LIBR
DEFINE L=SYSDUMP
DEFINE S=SYSDUMP.BG -
         SYSDUMP.F1 -
         SYSDUMP.F2 -
         SYSDUMP.Fn REUSE=IMM
/*
/&

-----------------------------
where n = the number of the
          foreground partition
```

Figure 1-6. Example: Defining SYSDUMP with the LIBR Librarian Program

WHAT IS THE REQUIRED LIBDEF STATEMENT: To get the dumps stored into the sublibrary assigned to the partition, the ASI Job Control procedure for each partition should contain a LIBDEF statement as shown in Figure 1-7. In the example given in Figure 1-7 a dump sublibrary is connected to the BG partition.

```
// LIBDEF DUMP,CATALOG=SYSDUMP.BG,PERM
```

Figure 1-7. Example: LIBDEF Statement for a Dump Sublibrary

WHAT ARE THE REQUIRED OPTIONS FOR DUMP WRITING: The system writes the output of an automatically invoked ABEND dump into the dump sublibrary for the partition if you submit either of the following statements:

```
// STDOPT SYSDUMP=YES

// OPTION SYSDUMP
```

With the // STDOPT SYSDUMP statement you request the system to write dumps of the next and all subsequent jobs or job-steps into the dump sublibrary for the particular partition until the SYSDUMP option is deactivated. The STDOPT statement must be given in the BG partition and is active for all partitions.

The // OPTION statement is active only for the duration of the particular job. After EOJ, the option given in a previous STDOPT statement will be active again.

Deactivate Writing Dumps into the Dump Sublibrary for the Partition

The SYSDUMP option is deactivated by:

- // STDOPT SYSDUMP=NO
- // OPTION NOSYSDUMP
- UNBATCH command to deactivate the partition.
- LIBDROP DUMP,PERM

How to Identify the Stored Dumps

Once the dump library and dump sublibraries have been defined, dumps from various sources can be stored there. The dumps stored either automatically or by you via Info/Analysis onload processing have an identifier of the following format:

```
library.partition-id.Onnnnnnn
```

library the dump library name. The Info/Analysis program expects SYSDUMP as the dump library name.

partition-id Sublibrary name, normally the partition identifier, like BG or F3.

Onnnnnnn the dump identifier, up to 8 characters, the first of which must be alphabetic.

O = Alphabetic character. The characters L, M, N, S, X are for system use only.

n = Up to 7 alphameric characters.

If the VSE/Advanced Functions system creates the name, the format of the dump identifier is:

Onnnnnnn

n = integers between 0 and 9 which are maintained by the system automatically with every new store dump operation.

For example:

SYSDUMP.F4.00000002

is the name of a dump residing in the dump sublibrary for the F4 partition of the library SYSDUMP, with the identifier 00000002.

How to Handle a Dump Library Full Condition

Dumps can be stored in the dump sublibraries in two ways:

1. Automatically by the system.
2. Via the Info/Analysis onload process.

If the library becomes full while:

- The system writes dumps into it,
  the whole dump is printed on SYSLST and a dump-library-full information message is issued on SYSLST and SYSLOG.
- Info/Analysis onloads a dump,
  an onload-failed error message is issued and the onload process is terminated.

In both cases the amount of free space of the sublibrary is kept as it was before the dump write operation was started.

You can clear sublibrary space to make room for new dumps by deleting dumps that are no longer required. Use either the Info/Analysis DELETE control statement or, interactively, the delete function of the dump management panel of Info/Analysis. How to delete a dump (batch mode) is described under "Delete a Dump" on page 3-14.

Note: Do not delete a dump with a delete function other than the Info/Analysis delete function.

The Dump Management File

The dump management file BLNDMF contains information about dumps managed by Info/Analysis.

Info/Analysis adds this information either during dump management invocation (Info/Analysis searches the dump sublibraries for new dumps automatically), or when you specify the name of a new dump. For a dump produced as a result of a DUMP attention routine command or for a Stand-alone dump you want to onload, supply a name via the Info/Analysis statement

DUMP NAME (specify the current dump).

Once information about a dump has been added to the dump management file, the Info/Analysis functions can be used to process the dump.

A dump entry remains in the dump management file until the dump is deleted using the Info/Analysis function.

Initializing the Dump Management File

Before you can use the functions of Info/Analysis the dump management file has to be initialized.

This initialization is accomplished by the UTILITY statement of Info/Analysis. The statement is used at system installation time and whenever you want to initialize or recreate the dump management file, for example after you have increased the size of the file, or after the file has been damaged.

For an explanation of how to change the size of the dump management file see "UTILITY - Initialize Dump Management File" on page 19-9.

Figure 1-8 on page 1-25 shows a job example to initialize the dump management file.

Sample Initialization Job

```
// JOB    INIT
// ASSGN  SYSLST,00E
// ASSGN  SYS016,252     " DUMP MANAGEMENT FILE "

 An example of the DLBL and EXTENT  statements is shown in
 Figure 1-10 on page 1-27.

// EXEC   INFOANA,SIZE=400K


SELECT DUMP MANAGEMENT      | use the
UTILITY                     | utility
RETURN                      | function
                            |
SELECT END                  |

/*
/&
```

Figure 1-8. Sample Job: Dump Management File Initialization

The External Routines File

The external routines file with the file name BLNXTRN is created during system installation to store the names of dump analysis exit routines. These routines are used to analyze dumps stored in one of the dump sublibraries. The external routines file contains the name and, optionally a description of each routine available for use with Info/Analysis.

Load the Info/Analysis External Routines File

The names of the external analysis routines which you can call to analyze the stored dumps have to be stored in the file BLNXTRN.

The sample job shown in Figure 1-9 records the name of the analysis routine 'IJBXDBUG' in the Info/Analysis external routines file. IJBXDBUG is the analysis routine for Stand-alone dumps. For the description of the control statements used see VSE/Advanced Functions System Control Statements. The program OBJMAINT is described in the manual VSE/Advanced Functions System Utilities.

```
// JOB LOAD

// ASSGN SYS004,00C                                | defines the
                                                   | input device
                                                   |
// ASSGN SYS005,SYSRES                             | defines the
                                                   | output device
                                                   |
// DLBL UOUT,'INFO.ANALYSIS.EXT.RTNS.FILE',,SD     | defines
                                                   | DLBL/EXTENT
// EXTENT SYS005,,1,0,6117,1                       | for BLNXTRN
                                                   |
// EXEC OBJMAINT                                   | execute OBJMAINT
./ CARD DLM=$$                                     |
./ COPY                                            |
ANEXIT IJBXDBUG   ANALYZE STAND ALONE DUMP         |
$$
/*
/&
```

Figure 1-9. Sample Job: Load External Routines File

What Labels Have to Be Established

Figure 1-10 on page 1-27 shows an example of the DLBL and EXTENT information you submit, if you want to use the functions of the Info/Analysis program. These labels should be stored in the system standard label area.

```
* LABELS FOR THE SYSDUMP LIBRARY,
*           THE DUMP MANAGEMENT FILE AND
*           THE EXTERNAL ROUTINES FILE

   ... ... ...
// DLBL SYSDUMP,'VSE.DUMP.FILE',99/365,,DSF
// EXTENT SYS010,,1,0,5472,644
// DLBL BLNDMF,'INFO.ANALYSIS.DUMP.MGNT.FILE',0,SD
// EXTENT SYS016,volid,1,0,6116,1
// DLBL BLNXTRN,'INFO.ANALYSIS.EXT.RTNS.FILE',0,SD
// EXTENT SYS017,volid,1,0,6117,1
   ... ... ...
----------------------------------------------
volid .... volid of the volume where the dump files
           reside, for example SYSWK1.
```

Figure 1-10. Example: File Labels for Dump Processing

CHAPTER 2. REQUESTING A DUMP

```
┌───────────┐    ┌────────────┐    ┌────────────┐
│   Dumps   │    │            │    │            │
│  General  │   >│ Requesting │<   │Printing the│
│Description│   >│   a Dump   │<   │ Stored Dump│
│           │    │            │    │            │
└───────────┘    └────────────┘    └────────────┘
  Chapter 1        Chapter 2         Chapter 3

                 ┌────────────┐
                 │            │
                 │    Dump    │
                 │  Utilities │
                 │            │
                 └────────────┘
                   Chapter 4
```

VSE/Advanced Funotions offers various functions with which storage areas can be dumped. These functions differ in their output contents, output device, and way of activation. You may use these functions to isolate system program or application program errors.

Contents

Overview of Dump Requests

The table in Figure 2-1 summarizes the dump functions offered by VSE/Advanced Functions. The table may help you to find the dump request function which is the most effective one for your particular error situation.

Initiated by/via	Output Contents	Output Device	Initiated/Controlled by
System (ABEND)	System- or Partition-Dump	Dump Sublib. or SYSLST	OPTIONS to Request the Dump
Operator (Console)	System- or Partition-Dump	Dump Sublib. or SYSLST	CANCEL Command
Operator (Console)	Selected Storage Areas	Tape or Printer	DUMP Command
Operator (Console)	System Storage	Tape or Printer	STAND-ALONE DUMP Program
Programm./Oper. (Defined Event)	Defined Storage Areas	Tape, Printer or Buffer	SDAID Dump Trace
Programmer (Macro)	Macro Dependent	Macro Dependent	MACROS (PDUMP, DUMP, JDUMP)

Figure 2-1. Dump Requesting Functions

Note, that the base structure of most of the dumps is shown under "Dump Contents Overview" on page 1-3.

Each of the dump requesting functions listed in Figure 2-1 is described in the following sections.

OPTIONS Controlling the ABEND Dump Function

[//] **STDOPT** [DUMP=YES|DUMP=PART][,SYSDUMP=YES]

[//] **OPTION** [DUMP|PARTDUMP][,SYSDUMP]

The ABEND dump function is internally called when the VSE/Advanced Functions system detects an ABEND condition (or if a CANCEL command has been given, see the following section).

You can define which kind of dump is to be retrieved by the ABEND dump function and what output device is to be used to store or print the dump.

The **STDOPT** command provides operands to control the ABEND dump function in the following way:

DUMP=YES requests a system-dump.

DUMP=PART requests a partition-dump.

SYSDUMP=YES defines the dump library SYSDUMP as output destination.

DUMP=NO specified with the STDOPT statement requests that no dump is to be taken.

The **// OPTION** statement provides operands to control the ABEND dump function in the following way:

DUMP Requests a system-dump.

PARTDUMP Requests a partition-dump.

SYSDUMP Requests the dump library SYSDUMP as the output destination.

If you want to suppress the dump output, OPTION NODUMP can be used.

> **Note:** If you use the STXIT PC or STXIT AB macro, you will not get any dump output, even if you include the DUMP or PARTDUMP option.

The ABEND dump function is described under

=====> "The ABEND Dump Function" on page 1-5.

Please find the description of the contents of a system or a partition dump under

=====> "Contents of the ABEND Dump Output" on page 1-9.

How a stored ABEND dump can be printed is described under

=====> "Print Dump Information" on page 3-16.

Requesting a Dump by the CANCEL Command

```
CANCEL cuu|{BG|Fn}[,DUMP|,PARTDUMP]
                  [,SYSDUMP]
                  [,FORCE]
```

The CANCEL command, when used as a job control command cancels the execution of the current job in the partition in which the command is given.

When used as an attention routine command the following specifications are possible:

cuu Defines an I/O device for which a pending I/O request is to be canceled.

BG|Fn Defines the partition to be canceled.

DUMP Is the default option; it produces a SYSTEM dump. For the contents of a system dump, please see "System-Dump Contents" on page 1-10.

PARTDUMP Produces a partition dump. The contents of a partition dump is shown under "Partition-Dump Contents" on page 1-10.

SYSDUMP Indicates that the requested dump has to be written to the dump sublibrary allocated to the partition.

If you omit the SYSDUMP operand, the option for SYSDUMP given in the STDOPT statement for the partition or in the OPTION statement for the job will be taken.

FORCE Forces an immediate termination of the tasks running in the defined partition without regard to active critical system functions.

If you omit the FORCE operand in the cancel statement, system functions like librarian directory updates are completed before the cancel function is performed.

You can suppress the generation of a dump with the NODUMP option and you can use the option NOSYSDUMP if you want to print the dump output on SYSLST.

A detailed description of the options for the CANCEL command is given in VSE/Advanced Functions System Control Statements.

How to print a CANCEL command dump (ABEND dump) from a dump sublibrary is described under

======> "Print Dump Information" on page 3-16.

Requesting a Dump by the DUMP Command

DUMP {SUP|partition-id|SVA|BUFFER|[space-id,]address-address},cuu

---

With the attention routine DUMP command particular areas of storage can be dumped. You may use the DUMP command, for example, to dump the buffer of the SDAID trace program selectively.

SUP Produces a dump of the supervisor area and the control registers.

Partition-id Defined in the form of BG or Fn produces a dump of the PSW, the general and floating point registers from the partition save area, and the specified partition area.

SVA Produces a dump of the SVA, including the system GETVIS area and the VPOOL area of the VIO support.

BUFFER Writes the contents of the buffer used by the SDAID program to tape; the command is rejected if any output device other than tape is specified.

space-id Indicates the address space (optional).

address-address Defines the start and end addresses of the storage area that is to be dumped. The PSW and the registers associated with any active partition partially or fully covered by that area are also included in the dump. The addresses must be specified in hexadecimal notation.

cuu Defines the channel and unit number of a 9-track tape drive or a printer.

If cuu is the address of a tape drive, mount an unlabeled tape volume on that drive. The dump output is written without any prior positioning of the tape.

Please find the description of how to print the DUMP command output from tape under

=====> "Print Dumps Stored on Tape" on page 3-25.

Requesting a Stand-Alone Dump

The following steps describe how to invoke the dump process using the Stand-alone dump program and how the program works with various output device types (cards, diskette, or tape).

Do not reset the processor storage by pressing the System Reset key or by causing a system reset operation via the display console before taking the dump.

1. **Press** the stop key

2. If your system operates in

   370 mode

   - **Take a note** of the contents of the storage locations X'0000' through X'0018' using the hardware display feature (IPL of the Stand-alone dump program will overwrite these locations).

   - Perform a **Store Status** operation.

   ECPS:VSE mode

   - Perform a **Machine Save** operation. The status is saved manually by activating the

     – Stop key and
     – Machine Save key.

3. **Ready** the device from which the Stand-alone dump program is to be read into virtual storage (from tape, diskette, or cards).

4. **IPL** the Stand-alone dump program from that device.

The subsequent steps of execution depend on the device type from where your Stand-alone dump program has been loaded.

Executing the Stand-Alone Dump Program from Magnetic Tape

When the IPL of the Stand-alone dump program has finished, no further operator action is required. The dump output is written onto the tape volume from which the program was loaded.

If a program check occurs as a result of incorrect information in the systems page tables, the program dumps only a real dump. It collects only those pages which are in processor storage at that moment, without address translation. The program may or may not issue a completion message on SYSLOG as described in the note below.

Executing the Stand-Alone Dump Program from Diskette or Cards

When the IPL has finished, no further operator action is necessary except making a printer available.

The Stand-alone dump program enters a wait state, if a printer address has not been defined at Stand-alone dump program generation. The program waits until a printer has been made ready.

Incorrect information in the systems page table results in only a real dump being taken.

> **Note:** A message indicating the completion of the requested Stand-alone dump is displayed at the console only if the address of the system's console at the time you IPL the Stand-alone program is the same as the one used during generation of the Stand-alone dump program.

The Stand-alone dump program and its output is described under

=====> "The Stand-Alone Dump" on page 1-11.

The creation of the Stand-alone dump program is described under

=====> "Generate the Stand-Alone Dump Program" on page 4-3.

A description of how to print the Stand-alone dump program output can be found under

=====> "Print Dumps Stored on Tape" on page 3-25.

Requesting a Dump on Event (SDAID Dump)

```
OUTPUT=DUMP
```

You may define that a dump has to be produced whenever a certain trace event occurs. The OUTPUT definition of the SDAID program is used for this purpose.

The following SDAID specifications for dump areas are possible:

- Partition
- Phase
- Area specified by storage addresses
- Area addressed by a register
- Area addressed by a pointer.

For example, a storage alteration trace combined with a dump output definition:

When the SDAID program has been activated a dump of the defined storage area is taken whenever an alteration of the defined storage appears.

The SDAID program is fully described in

=====> "Part 2. SDAID Trace."

Requesting a Dump from a Program

PDUMP - MACRO, JDUMP - MACRO , DUMP - MACRO

VSE/Advanced Functions supports the requesting of a virtual storage dump through dump macros. These macros, for example PDUMP, DUMP, or JDUMP may be issued in any program written in assembler language.

The PDUMP macro provides a dump of the general registers and of the storage area you defined with the macro operands on SYSLST regardless of the active options. If your program issues the macros DUMP or JDUMP, VSE/Advanced Functions dumps the contents of the entire supervisor plus the used part of the system GETVIS area, or, if the options DUMP=NO (NODUMP) or DUMP=PART (PARTDUMP) are active, some supervisor control blocks plus the registers and the contents of the partition that issued the macro.

Detail information on the output device and the output contents of the dump macros and the STXIT macro, are given in <u>VSE/Advanced Functions Application Programming: Macro Reference.</u>

CHAPTER 3. PRINTING THE STORED DUMP

```
┌────────────┐    ┌────────────┐    ┌────────────┐
│   Dumps    │    │            │    │            │
│  General   │    │ Requesting │   >│Printing the│<
│Description │    │   a Dump   │   >│ Stored Dump│<
│            │    │            │    │            │
└────────────┘    └────────────┘    └────────────┘
  Chapter 1         Chapter 2         Chapter 3

                  ┌────────────┐
                  │            │
                  │    Dump    │
                  │ Utilities  │
                  │            │
                  └────────────┘
                    Chapter 4
```

This chapter describes how you can process and print the dumps stored on a tape or in a partition's dump sublibrary. You will find examples for each dump operation.

The methods of evaluating the stored dumps described in this chapter are all related to a batch environment.

For reference information on the Info/Analysis batch commands the Chapter 19, "Info/Analysis in Batch Mode" on page 19-1 may be helpful.

Contents

STANDARD INFO/ANALYSIS BATCH JOB STREAM

Sample Job to Invoke Info/Analysis in Batch Mode

You invoke Info/Analysis by submitting the necessary JCL followed by control statements that request functions. You may submit the job either by entering statements on the console, or by submitting a job to the system input device.

Figure 3-1 shows a sample job to invoke the Info/Analysis program (// EXEC INFOANA). Assume that the dump sublibraries in the library SYSDUMP have already been defined and the label information for the SYSDUMP library resides in the standard label area. Information on the SYSDUMP library can be found under "The SYSDUMP Sublibraries" on page 1-18.

```
// JOB    JCL
// ASSGN  SYSLST,00E
// ASSGN  SYS016,252

 An example of the DLBL and EXTENT  statements is shown in
 Figure 1-10 on page 1-27.

// EXEC   INFOANA,SIZE=400K

 ....           |
 ....           | Info/Analysis
 ....           | control statements
 ....           |

/*
/&
```

Figure 3-1. Sample Job: Invoke Info/Analysis

List all Info/Analysis Batch Control Statements

You can use Info/Analysis to investigate and manage dumps taken to solve software problems in your system. With the HELP statement all of the available Info/Analysis batch commands are printed on SYSLST.

The HELP statement has no operands. You may include HELP at any point in a sequence of control statements, except after a SELECT END statement.

```
// JOB    HELP
// EXEC   INFOANA,SIZE=400K

                  |
HELP              | calls the HELP function
SELECT END        |
                  |
/*
```

Figure 3-2. Sample Job: List Info/Analysis Batch Commands

For more information on using the Info/Analysis in batch mode refer to Chapter 19, "Info/Analysis in Batch Mode" on page 19-1.

SELECT A DUMP AND PROCESS IT

List Names of Dumps Managed by Info/Analysis

To have Info/Analysis print the names of all dumps under its control, use the PRINT DATA statement (DUMP MANAGEMENT). A sample job is shown in Figure 3-3.

```
// JOB LIST

 An example of the job control statements to
 invoke Info/Analysis is shown in Figure 3-1 on page 3-3.

// EXEC INFOANA,SIZE=400K
                         |
SELECT DUMP MANAGEMENT   | calls the 'list
PRINT DATA               | managed dumps'
RETURN                   | function
                         |
SELECT END

/*
/&
```

Figure 3-3. Sample Job: List Managed Dumps

An output example of the PRINT DATA function is shown in Figure 3-4 on page 3-6.

```
   DUMP NAME        ONLINE   DATE/TIME TAKEN   VOLID
SYSDUMP.F3.00000010         84/04/12   08:44:51 T03111
SYSDUMP.F3.DMPC01             TO BE ONLOADED
SYDDUMP.F6.00000007    Y    84/04/12   08:11:06
SYSDUMP.F4.00000006    Y    84/03/28   22:05:16
SYSDUMP.F4.00000003    Y    84/03/17   01:31:21 T02818
SYSDUMP.F5.00000002    Y    84/03/08   15:38:42 T42901
```

Figure 3-4. Example: Dump Management PRINT DATA Output

The column headings represent:

- DUMP NAME - The identifier of the dump.
- ONLINE - An indication (Y, for yes) if the dump is currently stored in the dump sublibrary.
- DATE/TIME TAKEN - The date and time the dump was created or, if the actual date and time are not available, the date and time the dump was identified to Info/Analysis.

  "TO BE ONLOADED" indicates that the dump is not in the dump sublibrary (for example a Stand-alone dump named to the Info/Analysis management, but not yet onloaded).
- VOLID - The identifier of the tape volume to which the dump has been offloaded or from which the dump has been onloaded, if any. If a dump has been offloaded and then onloaded again, Info/Analysis retains the volume id in the dump management file. Consequently, a dump may be in a dump sublibrary ("Y" in ONLINE field) and still have a VOLID.

For more information on using the Info/Analysis in batch mode refer to Chapter 19, "Info/Analysis in Batch Mode" on page 19-1.

Select a Particular Dump to Work with

Specify the name of the dump you wish to process by entering the DUMP NAME statement. You have to specify a dump before you perform any Info/Analysis function except the Dump Management UTILITY or PRINT functions and the HELP function.

DUMP NAME Causes the following operations:

- Create a dump name entry in the dump management file,
  - The dump name entry is created, if this dump name has not been specified to Info/Analysis before.
  - The entry is flagged "TO BE ONLOADED", if the dump has been named but does not exist in the dump sublibrary yet.
- Define the named dump as the current dump; that is, all subsequently selected functions process that dump until you enter another dump name or until it has been deleted.

dumpname Represents the qualified dump name created either automatically or by yourself. You would create a name for instance, if you want to onload a Stand-alone dump from tape.

For example:

SYSDUMP.F3.00000010

defines a dump name which indicates:

- SYSDUMP as the library
- F3 as the sublibrary for the F3 partition
- 00000010 as the dump identifier. If you have to create a dump name (for example if you want to onload a Stand-alone dump or DUMP command dump), the dump identifier may be a one to eight character name where the first character has to be alphabetic.

  Note: For the first character L, M, N, S, and X are reserved for system use only.

Figure 3-5 shows a job to select a dump for further Info/Analysis processing.

```
// JOB    SELECT

 An example of the job control statements to
 invoke Info/Analysis is shown in Figure 3-1 on page 3-3.

// EXEC INFOANA,SIZE=400K
                                |
                                | selects the
DUMP NAME SYSDUMP.F3.00000010   | dump
                                |
SELECT .....                    | selects the
  .                             | function
  .                             |
  .

RETURN
SELECT END

/*
/&
```

Figure 3-5. Sample Job: Select a particular Dump to Work with

For Dumps not Stored in the Dump Sublibrary: Once the dump has been named to the Info/Analysis dump management file, it can be brought into a dump sublibrary using the onload function of Info/Analysis.

Load a Dump into a Dump Sublibrary

Dumps can be stored on tape as output of the following functions:

- DUMP command
- Stand-alone dump program
- Info/Analysis Offload operation

Before you can process these dumps, they have to be onloaded into a dump sublibrary.

The example in Figure 3-6 shows such an onload job; it stores the dump with the name SYSDUMP.BG.DMPL03 in the dump sublibrary assigned to the BG partition.

```
// JOB ONLOAD

// ASSGN SYS009,281

 An example of the job control statements to
 invoke Info/Analysis is shown in Figure 3-1 on page 3-3.

// EXEC INFOANA,SIZE=400K

                                 |
DUMP NAME SYSDUMP.BG.DMPL03      | 1, defines
                                 |    dump name
SELECT DUMP ONLOAD               | 2, calls
                                 |    ONLOAD
                                 |
VOLID T03111 SYS009              | 3, specifies
                                 |    volume and optional
                                 |    logical unit
FILE 2 LAST                      | 4, specifies
                                 |    second
                                 |    file
RETURN                           |
SELECT END                       |
/*
/&
```

Figure 3-6. Sample Job: Onload a Dump from Tape into a Dump Sublibrary

Figure 3-6 on page 3-9 performs the following:

1. Specifies the dump for processing, by name.

   The dump sublibrary is determined by the dump name.

2. Calls the Info/Analysis DUMP ONLOAD function.

3. Specifies the tape volume on which the dump resides.

   The volume name is provided in the list of managed dumps if the dump in question has been offloaded before. See "List Names of Dumps Managed by Info/Analysis" on page 3-5, for the list function.

   If the dump on tape you want to onload has not been offloaded before, the volume id is an identifier of your own choosing. It is used to identify the volume in subsequent dump operations.

4. Defines the file sequence number 2 with the LAST operand in the FILE statement (the dump resides on a multifile tape volume, sequence number 1 is the default value).

During the ONLOAD process Info/Analysis searches for a free tape drive if the tape unit is not defined via the 'ASSGN' and 'VOLID' statements and issues a volume mount request message (MOUNT VOLUME volumename ON UNIT xxx ....).

Info/Analysis indicates the successful execution of the ONLOAD function with a message (DUMP dumpname ONLOADED).

For more information on the statements used in the above sample job, see Chapter 19, "Info/Analysis in Batch Mode" on page 19-1.

Offload a Dump to Tape

Figure 3-7 shows how to offload a dump to a tape, erase the copy in the dump sublibrary and update the Info/Analysis dump management file.

```
// JOB    OFFLOAD

// ASSGN SYS009,280

 An example of the job control statements to
 invoke Info/Analysis is shown in Figure 3-1 on page 3-3.

// EXEC INFOANA,SIZE=400K

                               |
DUMP NAME SYSDUMP.F4.00000003  | defines the
                               | dump name
                               |
SELECT DUMP OFFLOAD            | calls
                               | OFFLOAD
                               |
VOLID T07111 SYS009            | defines the
                               | tape volume
                               | and optionally
                               | the logical unit
RETURN                         |
SELECT END                     |
/*
/&
```

Figure 3-7. Sample Job: Offload a Dump to Tape

The tape mounted at SYS009 is rewound before the OFFLOAD starts.

During the OFFLOAD process Info/Analysis searches for a free tape drive if the tape unit is not defined via the 'ASSGN' and 'VOLID' statements and issues a volume mount request message (MOUNT VOLUME volumename ON UNIT xxx ....).

Info/Analysis indicates the successful execution of the OFFLOAD function with a message (DUMP dumpname OFFLOADED...).

The following operations can be performed with Info/Analysis DUMP OFFLOAD, depending on whether any of the additional statements, ERASE or BYPASS are used.

OFFLOAD operation without any additional statement. (ERASE YES default)

- Copies the dump to tape.
- Deletes the dump from the dump sublibrary.
- Updates the dump management file to reflect the offload operation.

With ERASE NO specified:

- Copies the dump to tape.
- Updates dump management file with offload information.

With BYPASS specified:

- Skips the the copy to tape operation.
- Erases the dump from the dump sublibrary.
- Keeps the entry of the dump in the dump management file.

In comparison to the SELECT DUMP OFFLOAD the SELECT DUMP MANAGEMENT DELETE operation described in the following section, has the following functions:

DELETE operation

- Erases the dump from the dump sublibrary.
- Erases the information about the dump from the dump management file.

Figure 3-8 summarizes the functions of the SELECT DUMP OFFLOAD and the SELECT DUMP MANAGEMENT DELETE operation and shows the differences between these two operations.

Dump written to Tape	Information kept in Dump Management File	Dump erased from Sublibrary	Info/Analysis Function
YES	YES	YES	OFFLOAD without additional Operands
YES	YES	NO	OFFLOAD with ERASE NO specified
NO	YES	YES	OFFLOAD with BYPASS specified
NO	NO	YES	DELETE

Figure 3-8. Summary: SELECT DUMP OFFLOAD and DELETE Operation

Delete a Dump

You can erase a dump, and delete its corresponding information in the Info/Analysis dump management file, by using the DELETE statement of the Info/Analysis dump management function as shown in Figure 3-9. The example shows a job to delete two dumps in one dump management run. Note, that the DUMP NAME statement is valid prior as well as after the SELECT DUMP MANAGEMENT statement.

```
// JOB    DELETE

 An example of the job control statements to
 invoke Info/Analysis is shown in Figure 3-1 on page 3-3.

// EXEC INFOANA,SIZE=400K
                                 |
                                 |
SELECT DUMP MANAGEMENT           | calls
                                 | dump management
DUMP NAME SYSDUMP.F5.00000002    | specifies
                                 | the dump
DELETE                           |
                                 |
DUMP NAME SYSDUMP.F5.00000003    |
DELETE                           |
                                 |
                                 |
RETURN                           |
SELECT END
/*
```

Figure 3-9. Sample Job: Delete Dumps

Info/Analysis indicates the successful execution of the DELETE function with a message (DUMP dumpname DELETED).

A detailed description of the statements used with the delete operation in batch mode is given in Chapter 19, "Info/Analysis in Batch Mode" on page 19-1.

If you want to retain the information about the dump in the dump management file, you can use the SELECT DUMP OFFLOAD operation with the BYPASS operand. Please see Figure 3-8 on page 3-13, which shows a summary of the OFFLOAD and DELETE functions.

Call an Analysis Routine

Analysis routines can be used to analyze the stored dumps. For example, the routine IJBXDBUG (shipped as part of VSE/Advanced Functions) analyzes the output of Stand-alone dumps and adds the analysis information to the dump sublibrary.

You invoke these analysis routines with the CALL statement under the Info/Analysis SELECT DUMP VIEWING function.

You can print the analysis information together with formatted dump areas using the PRINT FORMAT statement under the SELECT DUMP VIEWING function. An example of such an operation is shown under "Print Areas Formatted" on page 3-22.

The example in Figure 3-10 shows a job which

- Selects the dump SYSDUMP.F3.00000010;
- Calls the analysis routine IJBXDBUG.

```
// JOB ANALYSE

 An example of the job control statements to
 invoke Info/Analysis is shown in Figure 3-1 on page 3-3.

// EXEC INFOANA,SIZE=400K
                                |
DUMP NAME SYSDUMP.F3.00000010   | defines the
                                | dump name
                                |
SELECT DUMP VIEWING             | defines the
CALL IJBXDBUG                   | analysis
                                | routine
                                | call
RETURN                          |
SELECT END                      |
/*
```

Figure 3-10. Sample Job: Call the Analysis Routine IJBXDBUG

A description of the analysis routine IJBXDBUG can be found under "The Stand-Alone Dump Analysis Routine IJBXDBUG" on page 4-8.

PRINT DUMP INFORMATION

The following sections describe how Info/Analysis is used to print information from the dumps stored on a tape or in a dump sublibrary.

The contents of a given dump depends on the function which created the dump. But the main form is the same for all dump requesting functions which are able to store dumps on tape or in a dump sublibrary. The functions described in the subsequent sections can be used for any type of dump stored in a dump sublibrary.

Figure 3-11 gives an overview of the various parts of a dump. Info/Analysis can be used to print these parts selectively.

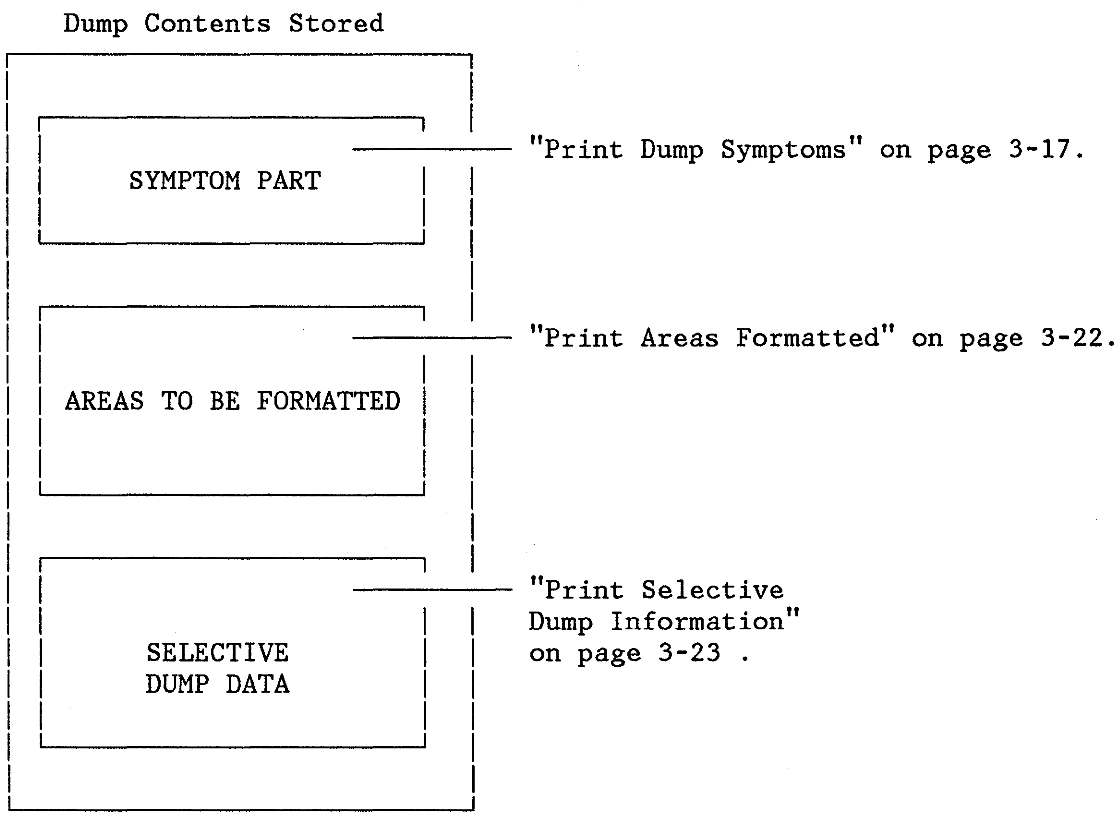


Figure 3-11. Overview: Dump Contents

For a detailed description of using the Info/Analysis in batch mode refer to Chapter 19, "Info/Analysis in Batch Mode" on page 19-1.

Print Dump Symptoms

To print the dump symptoms (part of the symptom record), use the SELECT DUMP SYMPTOMS function of Info/Analysis.

Figure 3-12 shows a sample print dump symptoms job.

```
// JOB PRINT

 An example of the job control statements to
 invoke Info/Analysis is shown in Figure 3-1 on page 3-3.

// EXEC INFOANA,SIZE=400K
                                 |
DUMP NAME SYSDUMP.F3.00000010    | defines the
                                 | dump name
                                 |
SELECT DUMP SYMPTOMS             | calls the
PRINT DATA                       | print operation
                                 |
RETURN                           |

SELECT END
/*
/&
```

Figure 3-12. Sample Job: Print Dump Symptoms

A description of the symptom record can be found in Appendix C, "Symptom Record Overview" on page C-1.

Symptom Part Example

Figure 3-13 shows an output example of a print dump symptoms operation.

```
                         DUMP SYMPTOMS


SYSDUMP.F3.00000011

ENVIRONMENT:
    CPU MODEL ...........4331
    CPU SERIAL ..........020001
    TIME ................08:44:51:00
    DATE ................84/04/12
    SYSTEM ID ...........5666-301
    RELEASE .............H
    FEATURE .............07
    DUMP TYPE ...........SCPREQ
    PROBLEM NUMBER ......

REQUIRED SYMPTOMS:

    AB/S0900
    REGS/0E014
    REGS/0C14E
    MS/0V15I
    RIDS/$IJJTTOP
    ADRS/E794E6


OPTIONAL SYMPTOMS (SDB):

OPTIONAL SYMPTOMS (NON-SDB)
```

Figure 3-13. Example: Output of Print Dump Symptoms

Symptom Part Description

Environment

The environment section of the symptom record describes the environment at the time the dump was taken. This data is provided by the dumping system component.

All of the information in the environment section is self-explanatory except the dump type entry, which may be:

SCPREQ for ABEND dumps

IDUMP for internal VSE/Advanced Functions dump requests

OPRREQ for DUMP command dumps

SADUMP for Stand-alone dumps

Required Symptoms

Required symptoms are those considered essential to your problem analysis effort. They are provided by the dump originating component or by subsequently executed analysis routines.

Each symptom is expressed with a prefix and the specific data connected by a slash.

The possible symptoms under the heading 'Required Symptoms' (if they exist for a problem type) are:

AB/Sxxyy Cancel code

xx ... first cancel code
yy ... second cancel code or 00 if none exists

example: AB/S0900

The following symptoms are additional information to the various cancel codes.

ADRS/address Reflects the absolute address of the instruction following the failing one if the failing address is outside the LTA, SVA, or partition areas (the address is extracted from the PSW).

MS/xxxxx Message number

example: MS/0V15I

OFFS/offset Offset of the instruction following the failing one relative to the phase start address, if the phase resides in the LTA or in the SVA, or relative to the partition begin address (the address is calculated from the PSW).

OPCS/aaaaaa aaaaaa Represents either

SVCnn (nn = SVC code) or
CODEmm (mm = program interruption code)

An entry for OPCS is generated only if an illegal SVC or if a program check occurs. If an illegal SVC occurs, the OPCS entry contains the failing SVC code in decimal. If a program check occurs, the entry contains the /370 mode program interruption code.

For program interruption codes refer to System /370 Reference Summary.

PIDS/comp.id Component identifier

example: PIDS/5666301V2

REGS/xxyyy

- xx is the register number in hex which contains a value, less than and within 4K (K=1024 bytes) of the PSW address at the time of failure. This register contains an address that is close to the point of failure.
- yyy is the difference between the PSW and the register address.

example: REGS/0C14E

0C ...... number of the general purpose register X'C'
14E ..... is the difference between the PSW address minus the contents of register (C).

RIDS/Caaaaaaaa

aaaaaaaa stands for:

- The name of a phase, if the failing instruction is in the SVA or LTA.
- The name of a program if the error occurred in a partition.

example: RIDS/DFHTUB

VALU/Caaaaaaaa

aaaaaaaa represents either:

- phase name (for example, combined with AB/S2200 phase not found)
- SYSnnn (for example, together with AB/S2600 SYSnnn not assigned).

Optional Symptoms (SDB)

SDB stands for structured data base. For information on the symptom record please refer to Appendix C, "Symptom Record Overview" on page C-1.

These symptoms (SDB) may be provided by the function which produced the dump or by subsequently executed analysis routines. These additional symptoms apply to the problem and may be present if the problem recurs. They are in SDB format; for example, the component level may be included in this section.

Optional Symptoms (Non-SDB)

These symptoms (non-SDB) are optionally provided by the dump originating component or by subsequently executed analysis routines. They are free-form symptoms that may be used in problem analysis but do not fit into the SDB format.

Note, that a symptom added by an analysis routine contain a preceding plus sign (+).

Print Areas Formatted

The printed output of a dump is called formatted if selected system information is extracted and printed separately.

To print a formatted dump, use the PRINT FORMAT statement which is part of the Info/Analysis DUMP VIEWING function. function. This function prints selected system areas, using locators and descriptors from the symptom record of the dump. It prints also the output of analysis routines which have been executed before.

The example in Figure 3-14 shows how to print the dump SYSDUMP.F3.00000010 formatted on the device at SYSLST.

```
// JOB PRINT

 An example of the job control statements to
 invoke Info/Analysis is shown in Figure 3-1 on page 3-3.

// EXEC INFOANA,SIZE=400K
                                 |
DUMP NAME SYSDUMP.F3.00000010    | defines the
                                 | dump name
                                 |
SELECT DUMP VIEWING              | calls the
PRINT FORMAT                     | PRINT FORMAT
                                 | function
                                 |
RETURN                           |
SELECT END                       |
/*
/&
```

Figure 3-14. Sample Job: Print a Dump in Formatted Form

Print Selective Dump Information

```
PRINT  {from-addr|from-addr to-addr|
        from-addr END|from-addr FOR length}
```

The PRINT statement of the Info/Analysis SELECT DUMP VIEWING function can be used to print dump information selected by addresses.

All addresses and the length setting are 1- to 4-character hexadecimal values representing valid addresses in the dump. Leading zeros are not required for an address specification.

The specification of

```
PRINT 0 END
```

for example, would cause the whole dump data to be printed.

See the description of the various selections under

======> "PRINT - Print Dump Data" on page 19-11.

The example in Figure 3-15 on page 3-24 shows the:

- Definition of the dump SYSDUMP.F3.00000010 to be processed.
- Definition of the selective print operation for dump data, beginning at the address X'302000' and ending at dump end.

```
// JOB PRINT

 An example of the job control statements to
 invoke Info/Analysis is shown in Figure 3-1 on page 3-3.

// EXEC INFOANA,SIZE=400K
                                 |
DUMP NAME SYSDUMP.F3.00000010    | defines the
                                 | dump name
                                 |
SELECT DUMP VIEWING              | defines the
PRINT 302000 END                 | area to be
                                 | printed
RETURN                           |
SELECT END                       |
/*
```

Figure 3-15. Sample Job: Print Selected Dump Areas

PRINT DUMPS STORED ON TAPE

The subsequent sections describe methods to print dumps that were written to tape, like:

- The Stand-alone dump.
- The DUMP command dump.

The **Info/Analysis program** can be used to format and print such dumps after they have been loaded into a dump sublibrary.

The steps which have to be performed to print a dump stored on tape are described under "Dump Tape Processed and Printed with Info/Analysis" on page 3-26.

In exceptional cases (for example, if the dump sublibrary is too small to hold the entire dump), the DOSVSDMP utility can be used to print the dump in unformatted form. If the DOSVSDMP utility is used, the dump need not be onloaded into a sublibrary. Figure 3-16 shows an overview of the different steps which you have to perform depending on the print method you choose.

Dump Tape Printed in Formatted Form with Info/Analysis

```
┌────────┐           ┌───────────┐         ┌───────┐
│ DUMP   │──────────>│   DUMP    │────────>│PRINTER│
│ TAPE   │  ONLOAD   │SUBLIBRARY │         │       │
└────────┘           └───────────┘         └───────┘
```

Dump Printed in Unformatted Form with DOSVSDMP

```
┌────────┐                                 ┌───────┐
│ DUMP   │────────────────────────────────>│PRINTER│
│ TAPE   │                                 │       │
└────────┘                                 └───────┘
```

Figure 3-16. Overview: Print a Dump Tape

The following sections describe how to print a dump tape with Info/Analysis and also how to use the DOSVSDMP utility for the same purpose. Sample jobs are included and the output is explained.

Dump Tape Processed and Printed with Info/Analysis

This section describes the steps which have to be performed in order to get a printed output of the Stand-alone dump or the DUMP command dump. The dump has to be in one of the dump sublibraries before Info/Analysis can be used to print information from it. Therefore the first operation to be done is to onload the dump into one of the dump sublibraries. Figure 3-17 shows an overview of such an operation.

```
┌──────┐          ┌─────────────┐     ┌───────┐
│      │          │             │     │       │
│ DUMP │────────->│    DUMP     │───->│PRINTER│
│ TAPE │  ONLOAD  │SUBLIBRARIES │     │       │
│      │          │             │     │       │
└──────┘          └─────────────┘     └───────┘
```

Figure 3-17. Overview: Steps to Print a Dump Tape

Steps to Print a Dump from Tape

The following describes in detail the steps which have to be performed.

1. Define the name of the dump.

   The name of the dump determines the dump sublibrary. For example:

   SYSDUMP.BG.DUMPSA2

   defines the sublibrary BG as the target library for the subsequent onload process.

2. Select the Info/Analysis onload function.

3. Define the tape volume (VOLID).

4. Mount the dump tape on the device, which Info/Analysis tells you during processing, or which you defined with the VOLID statement.

5. When the dump has been stored in the desired dump sublibrary, Info/Analysis can print the dump.

Stand-Alone Dump Printed with Info/Analysis

Following is an example of a typical Stand-alone dump print job and its output.

Sample Job to Print a Stand-Alone Dump

The example shown in Figure 3-18 defines a job which

- Onloads a Stand-alone dump with the user defined name SYSDUMP.BG.DUMPSA2.
- Prints the following on the device assigned to SYSLST:
  - The symptoms of the dump.
  - The formatted areas of the dump.

```
// JOB    PRINT

// ASSGN SYS009,281

 An example of the job control statements to
 invoke Info/Analysis is shown in Figure 3-1 on page 3-3.

// EXEC INFOANA,SIZE=400K
                                 |
DUMP NAME SYSDUMP.BG.DUMPSA2     | 1, defines the
                                 |    dump name
                                 |
SELECT DUMP ONLOAD               | 2, selects
                                 |    ONLOAD
VOLID T03111 SYS009              | 3, defines the
                                 |    tape volume
RETURN                           |
                                 |
SELECT DUMP SYMPTOMS             | 4, calls the
PRINT DATA                       |    DUMP SYMPTOMS
                                 |    function
RETURN                           |
                                 |
SELECT DUMP VIEWING              | 5, calls the
PRINT FORMAT                     |    DUMP VIEWING
                                 |    function
RETURN                           |
                                 |
SELECT END                       |
/*
/&
```

Figure 3-18. Sample Job: Print a Stand-Alone Dump

You can also print the dump data using

```
SELECT DUMP VIEWING
PRINT 0 END
```

which does no formatting of the dump. See also "Print Selective Dump Information" on page 3-23.

Printed Output of the Stand-Alone Dump

Figure 3-19 and Figure 3-20 on page 3-29 show the output of the sample job shown in Figure 3-18 on page 3-27. Figure 3-19 shows an example of the dump symptoms. A list of the formatted dump areas which are printed is given in Figure 3-20 on page 3-29.

OUTPUT OF THE DUMP SYMPTOMS, PRINT DATA OPERATION

```
ENVIRONMENT:
    CPU MODEL ...........4331
    CPU SERIAL ..........020001
    TIME ................08:44:51:00
    DATE ................84/04/12
    SYSTEM ID ...........5666-301
    RELEASE .............H
    FEATURE .............07
    DUMP TYPE ...........SADUMP
    PROBLEM NUMBER ......
  ....
  ..
    MACHINE = 370
  ..
    MODE = PAGING
```

Figure 3-19. Sample: Symptom Part of the Stand-Alone Dump Output (PRINT DATA)

SUMMARY OF THE DUMP VIEWING, PRINT FORMAT OPERATION OUTPUT

The following Stand-alone dump information is selected and printed.

PSW	at time of failure
FREGS	Floating point registers
GREGS	General purpose registers
CREGS	Control registers
MESSAGE	Error messages and the last 200 messages from the hard copy file
Analysis Routine Output (if available)	
LOWCORE	Low address storage
SYSCOM	System communication region
COMREG	Partition communication region for each partition
PIBTAB	Partition information block
PIB2TAB	Partition inform. block extension
LUBTAB	Logical unit block for each partition
PUBTAB	Physical unit block
PUB2TAB	Physical unit block extension
ERBLOC	Error recovery block
CHQTABLE	Channel queue table
CHNTAB	Channel control table
SAVAREA	Partition save areas for each partition
TIBATAB	Task inform. block address table
PCB	Partition control block
TIB	Task information block
TCB	Task control block
LPT	Library pointer table
LDT	Library definition table
SDT	Sublibrary definition table
EDT	Extent definition table
DDT	Device definition table

Figure 3-20. Summary: Formatted Areas of the Stand-Alone Dump (PRINT FORMAT)

DUMP Command Dump Printed with Info/Analysis

This section gives an example of how to print a DUMP command dump that is on tape; it also shows the output of the sample job.

Sample Job to Print a DUMP Command Dump

The example shown in Figure 3-21 defines a job to print the dump symptoms and the formatted dump areas of a DUMP command dump named SYSDUMP.BG.DUMPC02 on the device assigned to SYSLST.

An example of the output of this job is shown under Figure 3-22 on page 3-31 and Figure 3-23 on page 3-32.

```
// JOB PRINT

// ASSGN SYS009,280

 An example of the job control statements to
 invoke Info/Analysis is shown in Figure 3-1 on page 3-3.

// EXEC INFOANA,SIZE=400K
                                |
DUMP NAME SYSDUMP.BG.DUMPC02    | 1, defines the
                                |    dump name
                                |
SELECT DUMP ONLOAD              | 2, selects
                                |    ONLOAD
VOLID T03111 SYS009             | 3, defines the
                                |    tape volume
RETURN                          |
                                |
SELECT DUMP SYMPTOMS            | 4, defines the
PRINT DATA                      |    print operation
                                |
RETURN                          |
                                |
SELECT DUMP VIEWING             | 5, defines the
PRINT FORMAT                    |    print operation
                                |
RETURN                          |
                                |
SELECT END                      |

/*
/&
```

Figure 3-21. Sample Job: Print the Output of a DUMP Command

The statements used in the previous example are further explained in Chapter 19, "Info/Analysis in Batch Mode" on page 19-1.

Output of the DUMP Command Dump Printed by Info/Analysis

The output of the DUMP command dump printed by Info/Analysis consists of two parts:

- The dump symptom part (example shown in Figure 3-22)
- The formatted dump areas (summary of the areas shown in Figure 3-23 on page 3-32).

Figure 3-21 on page 3-30 shows the job with which these two dump parts can be produced.

OUTPUT OF THE DUMP SYMPTOMS, PRINT DATA OPERATION

```
ENVIRONMENT:
    CPU MODEL ...........4331
    CPU SERIAL ..........020001
    TIME ................08:44:51:00
    DATE ................84/04/12
    SYSTEM ID ...........5666-301
    RELEASE .............H
    FEATURE .............07
    DUMP TYPE ...........OPRREQ
    PROBLEM NUMBER ......


    DUMP 4400-8FFF,280 | which is the command,
                       | that requested the
                       | dump
```

Figure 3-22. Sample: Symptom Part of the DUMP Command Dump

OUTPUT OF THE DUMP VIEWING, PRINT FORMAT OPERATION

Figure 3-23 gives a list of the dump areas printed.

PSW	Program status word at time of failure
GREGS	General purpose registers at time of failure
FREGS	Floating point registers
LOADLS	Phase load list of the partition
Partition	Storage within the partition boundaries

Figure 3-23. Summary: Areas to be Printed with DUMP VIEWING, PRINT FORMAT

Dump Tape Printed with DOSVSDMP

How to print the tapes produced by the Stand-alone dump program and the DUMP command in unformatted form is discussed in this section. Normally Info/Analysis is used to process and print dump tapes. In exceptional cases the use of the DOSVSDMP utility may be necessary, for example, if none of your dump sublibraries is big enough to hold the Stand-alone dump.

The output of a Stand-alone dump taken on a central processor with ECPS:VSE can be printed by DOSVSDMP under a VSE system in 370 mode and vice versa.

The printed output of the DOSVSDMP utility contains for both, DUMP command tape or Stand-alone dump tape, the following:

- Symptom record.
- Unformatted dump data.

Sample DOSVSDMP Print Set-up

To print a dump from tape using the DOSVSDMP utility, invoke DOSVSDMP by submitting the control statements shown in Figure 3-24 on page 3-34.

The utility prompts you by messages for further control information which you enter at SYSLOG.

```
// JOB DOSVSDMP
// EXEC DOSVSDMP
```

DOSVSDMP prompts you by messages at SYSLOG to define the operation you want to perform, with:

```
┌──────────────── Prompt message ──────────────────┐
│ xx 4G01D SELECT ONE OF THE FOLLOWING FUNCTIONS   │
│ ...........................................      │
│ 1 CREATE STAND ALONE DUMP PROGRAM                │
│ 2 PRINT DUMP TAPE                                │
│ 3 PRINT SDAID TAPE                               │
│ R END DOSVSDMP PROCESSING                        │
└──────────────────────────────────────────────────┘
```

Enter **2** to invoke DOSVSDMP Print Dump Tape processing.

The DOSVSDMP utility response is:

```
┌──────────────── Prompt message ──────────────────┐
│ xx 4G02D SPECIFY THE TYPE OF THE DUMP            │
│ .....................................            │
│ 1 STAND ALONE DUMP ON TAPE                       │
│ 2 DUMP COMMAND ON TAPE                           │
│ R END DOSVSDMP PROCESSING                        │
└──────────────────────────────────────────────────┘
```

(ENTER ONE OPTION ONLY)

Enter **1** to print the Stand-alone dump or enter **2** to print the DUMP command dump on SYSLST.

The DOSVSDMP utility response is:

```
┌─────────────────── Prompt message ─────────────────────┐
│ xx 4G04D SPECIFY INPUT TAPE IN THE FORM SYSNNN OR CUU  │
│......................................................  │
└────────────────────────────────────────────────────────┘
```

Enter 280, for example, if the dump tape is mounted on the tape drive 280.

Figure 3-24 (Part 1 of 2). Sample: Dump Tape Printed with DOSVSDMP

The DOSVSDMP utility starts printing
on the device at SYSLST if the tape contains
a **Stand-alone dump**.

After print completion,
control is returned to Job Control.

If the dump tape contains a **DUMP command dump**
the DOSVSDMP utility responds with:

```
------------------ Prompt message ------------------
| xx 4G30D SPECIFY FILENUMBER ON TAPE              |
|.....................................             |
----------------------------------------------------
```

Enter 2, for example, if the second file contains the
DUMP command output you want to print.

Now the DOSVSDMP utility starts printing the
DUMP command tape on SYSLST.
After print completion, control is returned to
Job Control.

Figure 3-24 (Part 2 of 2). Sample: Dump Tape Printed with DOSVSDMP

CHAPTER 4. DUMP UTILITIES

```
 ┌───────────┐   ┌────────────┐   ┌────────────┐
 │   Dumps   │   │            │   │            │
 │  General  │   │ Requesting │   │Printing the│
 │Description│   │   a Dump   │   │ Stored Dump│
 │           │   │            │   │            │
 └───────────┘   └────────────┘   └────────────┘
   Chapter 1       Chapter 2        Chapter 3

                 ┌────────────┐
                 │            │
                >│    Dump    │<
                >│  Utilities │<
                 │            │
                 └────────────┘
                   Chapter 4
```

This chapter describes the functions of utilities that are used in problem determination. The utilities described in this chapter are:

- The DOSVSDMP Utility.
- The Stand-Alone Dump Analysis Routine IJBXDBUG.

Contents

THE DOSVSDMP UTILITY

The DOSVSDMP utility is used to create the Stand-alone dump program with which the applicable virtual storage of the entire VSE/Advanced Functions system can be dumped. The utility can also be used to print the output of the SDAID program, the DUMP command, or of the stand-alone dump program, from tape.

Functions of the DOSVSDMP Utility: The DOSVSDMP utility includes the following functions:

- "Generate the Stand-Alone Dump Program."
- "Dump Tape Printed with DOSVSDMP" on page 3-33,

  which describes how a DUMP command dump or a Stand-alone dump stored on tape can be printed.
- "Print the SDAID Produced Tape" on page 4-6.

Generate the Stand-Alone Dump Program

The Stand-alone dump program is mainly used in case of a hard or soft wait or if a system loop occurred. You can generate the Stand-alone dump program to reside on a **magnetic tape**, if you want to have a quick save of the dumped information for later processing.

For processing the tape see "Print Dumps Stored on Tape" on page 3-25.

The Stand-alone dump program can also be generated to reside on **diskette** or in **cards** (no dump information saved for later retrieval since output goes to a printer).

To generate a Stand-alone dump program, invoke DOSVSDMP by entering

```
// EXEC DOSVSDMP
```

The program, once it receives control, prompts you for further control information as shown in Figure 4-1 on page 4-4.

```
Prompt message
xx 4G01D SELECT ONE OF THE FOLLOWING FUNCTIONS
..............................................
1 CREATE STAND ALONE DUMP PROGRAM
2 PRINT DUMP TAPE
3 PRINT SDAID TAPE
R END DOSVSDMP PROCESSING
```

Enter **1** to invoke dump program generation.

The DOSVSDMP utility responds with

```
Prompt message
xx 4G04D SPECIFY OUTPUT DEVICE IN THE FORM SYSNNN OR CUU
.........................................................
```

The device defined with SYSNNN or CUU can be a:

- Tape device
- Card punch
- Diskette device

Figure 4-1. Sample: Stand-Alone Dump Program Generation

Generating the Program on Tape

If the selected device is a tape drive, it must be either 9-track or 7-track with data converter.

Note: Neither the utility DOSVSDMP nor the generated Stand-alone dump program supports the IBM 8809 tape drive in streaming mode.

The completion message

```
xx 4G09I DUMP PROGRAM HAS BEEN CREATED
..........................................
```

indicates the successful generation of the dump program.

You need not regenerate the Stand-alone dump program when a tape has been used. The dump program remains useable for all subsequent Stand-alone dump requests.

What kind of dump program is generated depends on the mode of system operation. For example, a Stand-alone dump program generated under 370 mode, can be used to dump also an address space of a VSE system operating in ECPS:VSE mode.

Generating the Program on Diskette or Cards

If you use a diskette or a card unit you will be prompted for a printer address.

```
┌─────────────── Prompt message ──────────────┐
|xx 4G06D SPECIFY PRINTER ADDRESS IN FORM CUU|
|............................................|
└─────────────────────────────────────────────┘
```

You may respond with

- the unit address of that printer which you want to use for the dump output at Stand-alone dump program execution time.
- end/enter if you want to choose the printer address at execution time.

Note, that a Stand-alone dump program generated under 370 mode which resides on a diskette or on cards cannot be used in E/mode and vice versa.

If you do not specify a printer address, the generated Stand-alone dump program will enter a wait state until the operator starts the printer device which is to be used to print the Stand-alone dump.

DOSVSDMP generates the Stand-alone dump program in the form of punched cards or on the diskette, whichever applies. The message

```
┌──────────────────────────────────────────┐
|xx 4G09I DUMP PROGRAM HAS BEEN CREATED    |
|....................................      |
└──────────────────────────────────────────┘
```

indicates successful generation of the program.

The description of how the Stand-alone dump program is executed can be found under "The Stand-Alone Dump Program" on page 1-12.

Print the SDAID Produced Tape

You may define that the SDAID trace information is recorded onto a tape reel. DOSVSDMP can be used to retrieve this information from tape and to print it on SYSLST.

This is done by responding to DOSVSDMP prompts as shown in the Figure 4-2.

When the utility gets control, it prompts you for further definitions via SYSLOG.

SDAID Tape Print Set-up Example

```
// JOB SDAID
// EXEC DOSVSDMP
```

DOSVSDMP prompts you to define the operation you want to perform:

```
------------------- Prompt message -------------------------
xx 4G01D SELECT ONE OF THE FOLLOWING FUNCTIONS
............................................
1 CREATE STAND ALONE DUMP PROGRAM
2 PRINT DUMP TAPE
3 PRINT SDAID TAPE
R END DOSVSDMP PROCESSING
```

Enter **3** to invoke DOSVSDMP Print SDAID Tape processing.

The DOSVSDMP utility responds is:

```
---------------------- Prompt message ----------------------
xx 4G04D SPECIFY INPUT TAPE IN THE FORM SYSNNN OR CUU
.....................................................
```

Enter 280, for example, if the SDAID output tape is mounted on the device 280.

DOSVSDMP prints the tape on the device assigned to SYSLST. After print completion, control is returned to Job Control.

Figure 4-2. Sample Job: SDAID Tape Printed with DOSVSDMP.

For the description of how to print the SDAID buffer that was dumped with the DUMP BUFFER,cuu command, see "Print Dumps Stored on Tape" on page 3-25.

THE STAND-ALONE DUMP ANALYSIS ROUTINE IJBXDBUG

The analysis routine IJBXDBUG analyses the output of a stand-alone dump onloaded to a dump sublibrary.

When IJBXDBUG receives control from Info/Analysis, it requests portions of dump data using Info/Analysis dump access routines. During analysis of the dump data, information from the dump is extracted and written back to the dump sublibrary for further Info/Analysis operations.

Activating the Routine

The routine name IJBXDBUG must be contained in the Info/Analysis external routines file before you can call the routine.

The statements

```
SELECT DUMP VIEWING
CALL IJBXDBUG
```

of Info/Analysis start execution of the analysis routine.

Figure 3-10 on page 3-15 illustrates the activation of an analysis routine.

How to start an analysis routine interactively is described under "Invoking an Analysis Routine" on page 15-29.

Output of the Routine

The output of the analysis routine may contain the following:

- General information.
- Specific information.

While general analysis information is provided in every dump, specific analysis information only concerns particular error situations. The output of the analysis routine is stored in the dump sublibrary. It can be printed together with formatted dump areas with the operation:

```
DUMP NAME ...........
SELECT DUMP VIEWING
PRINT FORMAT
```

General Analysis Information

The general analysis information, which is provided for each dump, is not dependent on certain error conditions.

HEADER ENTRY

This contains data as follows:

Service level identifier
Supervisor ID
Supervisor name
Date the dump was taken
Dump type
System status
Current task
Owner of LTA and transient name (if active)
DOC screen image buffer (if applicable)
ASYNOC reply status (if applicable)

ADDRESS VALIDATION

When an address has been located or calculated during the analysis process of the dump data, it is validated. IJBXDBUG checks whether the address is

- Within the range of high and low limits of the affected areas:
  - Supervisor
  - Partition
  - SVA.

When an address is found to be invalid, the address and information about the expected contents of the address are added to the dump.

An address validation entry might look like this:

```
INVALID ADDRESS FF0002 ENCOUNTERED DURING ANALYSIS.
ADDRESS OF: PUB TABLE FROM BG COMREG
```

Specific Analysis Information

The data which is selected, analysed, and finally stored in the dump sublibrary depends on the error situation. The following error conditions can be recognized by the IJBXDBUG routine:

- Hard Waits
  - WAITFFA
  - WAITFFB
  - WAITFFF
  - WAITFF9

- WAITFFE
- WAITFF4
- WAITFF8
- WAITFD0
- Other Hard Waits

• Soft Waits or System Running (Loop)

HARD WAIT DUMP ENTRIES

For all dumps that indicate a hard wait, the routine supplies

• Wait state code, and
• General purpose registers at the time the failure occurred

Besides this information, the following data, depending on the specific hard wait code, is provided:

For <u>WAITFFA, WAITFFB, or WAITFFF</u>

• Type of program check
• Program check address
• Instruction at the program check address
• Overwritten instruction information (if applicable)
• Name of transient which program-checked and the displacement within the transient (if applicable)
• Transient areas checked are LTA, PTA, DOC, and RTA
• Name of the SVA phase which program-checked and the displacement within the phase (if applicable)
• Registers at the time of failure
• 64 bytes of data pointed to by each register

For <u>WAITFF9 or WAITFFE</u>

• Last device to which a sense was issued
• Sense data address
• Sense data
• Registers at the time of failure
• 64 bytes of data pointed to by each register.

For <u>WAITFF4</u>

• Error recovery phase which has not been found
• Registers at the time of the failure
• 64 bytes of data pointed to by each register

For <u>WAITFF8</u>

• CRT phase which has not been found
• Registers at the time of the failure
• 64 bytes of data pointed to by each register

For WAITFDO

- IPL cancel code
- Registers at the time of the failure
- 64 bytes of data pointed to by each register

For other Hard Waits

- Registers at the time of the failure
- 48 bytes of data pointed to by each register

SOFT WAITS OR SYSTEM RUNNING

On all dumps indicating a soft wait or a system running condition, the status of all active devices (non-telecommunication) and active tasks is supplied.

Information is provided for active devices, excluding local telecommunication devices.

A device is active if:

- It is flagged busy in the PUB table
- It has a channel queue entry queued to the PUB table.

A task is active if it is not unbatched, stopped, or flagged not active in the TIB (Task Information Block).

Tasks of VSE/POWER are classified as not active if VSE/POWER has flagged the task as waiting for work.

The following information is provided:

Device Status Information

- Device address
- Device type
- Task-id of first channel queue entry
- I/O request status (I/O started or not started)
- Reason I/O not started (CSW stored, intervention, etc.)
- CSW from channel queue entry if interrupt has been presented
- A list of additional tasks with I/O queued for this device
- The device, that last presented an interrupt to the system

Task Status Information

- Task name
- Serviced task name
- Main task name for subtask
- Status
- What the task is waiting for
- Task information (LTA active, ICCF pseudo partition, EOT active, etc.)
- Subsystems running in the partition

Output Examples

Examples of IJBXDBUG output are given:

- For a hard wait X'FFF' - See Figure 4-3.
- For a soft wait - See Figure 4-4 on page 4-13.
- For a system loop - See Figure 4-5 on page 4-14.

```
                      VSE/AF 5666-301 V2 R2.1 ECLEV=0000

DATE DUMP WAS TAKEN:  02/25/85                    DUMP TYPE:  SADUMP
SUPERVISOR ID:  E-R2.1-03/10-11.53                SUPERVISOR NAME: $$A$SUPE
SYSTEM STATUS:  HARD WAIT                         HARD WAIT CODE:  FFF
CURRENT TASK:  BG MAIN TASK                       PHASE:  PAYROLL1

LTA ACTIVE, OWNER:  AR TASK                       PHASE NAME:  $$BOSVLT

PROGRAM CHECK TYPE:  0004 PROTECTION EXCEPTION
ADDRESS OF PROGRAM CHECK:  000A82
PROGRAM CHECK INSTRUCTION:  D2404973C8D8

COMPARING FAILING INSTRUCTION AGAINST CURRENT RUNNING SYSTEM
INDICATES POSSIBLE OVERLAY, INSTRUCTION CURRENTLY:  47F010C2D207

ASYNOC REPLIES OUTSTANDING:  NONE



REGISTERS AT THE TIME OF THE FAILURE:

REG  CONTENTS       DATA POINTED TO BY THIS REGISTER
---  --------  --- +0-------+4-------+8-------+C-------
 0   00000290  +00  D50A0000 28145400 28145400 28145400  *N...............*
               +10  D2034560 78900000 80F0F0F0 3245DD41  *K..-.....000....*
               +20  05054110 D8905880 F00047F0 C1000700  *....Q...0..0A...*
               +30  05111110 11101110 F0000000 000000C1  *........0..0...A*
                   +0-------+4-------+8-------+C-------
 1   000001FC  +00  N/A..... C150D777 30005400 90000000  *....A.P.........*
               +10  4040C1E3 D3C1D5E3 C1404040 B9984400  *  ATLANTA    ....*
               +20  980F9000 D4075400 28148800 FFFF980F  *....M...........*
               +30  08009000 00000000 00000000 00000000  *................*
                   +0-------+4-------+8-------+C-------
 2   00FFFFFF  +00  DATA NOT AVAILABLE
```

Figure 4-3. Example: WAITFFF Analysis Report

```
                 VSE/AF 5666-301 V2 R2.1 ECLEV=0000

DATE DUMP WAS TAKEN:  02/25/85                  DUMP TYPE:  SADUMP
SUPERVISOR ID:  E-R2.1-03/10-11.53              SUPERVISOR NAME:  $$A$SUPE
SYSTEM STATUS:  SOFTWAIT
CURRENT TASK:  CRT FOR BG                       PHASE: $$BOCRTA

LTA ACTIVE, OWNER:  AR TASK                     PHASE NAME:  $$BOSVLT

ASYNOC REPLIES OUTSTANDING:  FB-011


DEVICE ANALYSIS FOR ACTIVE NON TP DEVICES ONLY:

DEV  TYPE  TSK  I/O REQUEST STATUS AND INFORMATION
---  ----  ---  ---------------------------------------------------------------
009  1052  FB   I/O NOT STARTED, CSW STORED
                OTHER TASKS QUEUED:  BG,ASY,T00,F3
                CHANNEL QUEUE CHAIN ERROR.
01C  2540  F1   I/O NOT STARTED, INTERVENTION REQUIRED
                CSW: C0191250 02000045
131  3330  N/A  LAST I/O INTERRUPT WAS FROM THIS DEVICE


TASK ANALYSIS FOR ACTIVE TASKS ONLY:

TASK NAME     STATUS    TASK INFORMATION
------------  --------  ----------------------------------------------------------
SNS FOR BG    AVRBND    WAIT, AVR PROCESSING
BG MAIN TASK  READY     READY TO RUN
F1 MAIN TASK  WAITBND   WAIT FOR I/O OR ECB POST    CCB/ECB ADDRESS: 1911D8
```

Figure 4-4. Example: Soft Wait Analysis Report

```
                    VSE/AF 5666-301 V2 R2.1 ECLEV=0000

DATE DUMP WAS TAKEN:  02/14/85                 DUMP TYPE:  SADUMP
SUPERVISOR ID:  VSE.370.SUP3                   SUPERVISOR NAME:  $$A$SUPE
SYSTEM STATUS:  RUNNING                        PSW:  01000F0F 0027D01C
CURRENT TASK:  BG MAIN TASK                    BASE PHASE:  GSXDEBUG

AREA POINTED TO BY PSW:  SVA                   PHASE NAME:  $IJBLBR

SCANNING SVA/$IJBLBR , APAR IDENTIFIERS WERE FOUND.
APAR NUMBERS:  DY15017 DY20012

ASYNOC REPLIES OUTSTANDING:  F3-003 F2-002


DEVICE ANALYSIS FOR ACTIVE NON TP DEVICES ONLY:

DEV  TYPE  TSK  I/O REQUEST STATUS AND INFORMATION
---  ----  ---  ------------------------------------------------------------
009  1052  F7   I/O NOT STARTED, REASON UNKNOWN
                OTHER TASKS QUEUED:  SNS,FA,ASY
                CHANNEL QUEUE CHAIN ERROR
00E  1403  F4   I/O NOT STARTED, INTERVENTION REQUIRED
02A  2540  FB   I/O STARTED, AWAITING DEVICE END
100  3330  T37  I/O NOT STARTED, CSW STORED
                CSW: 00005980 001400007
104  3340  F2   I/O STARTED, AWAITING INTERRUPT
                OTHER TASKS QUEUED:  T38,F5
                I/O STARTED ON ALTERNATE CHANNEL
204  3375  F8   I/O STARTED, INTERRUPT RECEIVED
                CSW: 00105A98 0C40009E
                RAS RETRY INDICATOR ON FOR THIS DEVICE
260  3330  N/A  LAST I/O INTERRUPT WAS FROM THIS DEVICE
```

Figure 4-5 (Part 1 of 2). Example: System Loop Analysis Report

```
TASK ANALYSIS FOR ACTIVE TASKS ONLY:

TASK NAME     STATUS     TASK INFORMATION
------------  --------   ------------------------------------------------------
SNS FOR F6    WAITBND    WAIT FOR I/O OR ECB POST    CCB/ECB ADDRESS: 001212
ASY TASK      WAITBND    WAIT FOR I/O OR ECB POST    CCB/ECB ADDRESS: 002212
BG MAIN TASK  READY      READY TO RUN
F1 MAIN TASK  RURBND     WAIT FOR LOCKED RESOURCE
                         RESOURCE: PAYROLL.FILE      OWNER: BG
                         SVC RETRY INDICATOR ON      SVC: 6E (HEX)
F2 MAIN TASK  WAITBND    WAIT FOR I/O OR ECB POST    CCB/ECB ADDRESS: 1BF628
                         SUB SYSTEMS IN THIS PARTITION: ICCF CICS
F3 MAIN TASK  READY      READY TO RUN
                         VSAM AUTO CLOSE ACTIVE FOR THIS TASK
F4 MAIN TASK  CHQBND     WAIT FOR CHANNEL QUEUE
F7 MAIN TASK  WAITBND    WAIT FOR I/O OR ECB POST    CCB/ECB ADDRESS: 5BF620
                         TASK IS LTA OWNER
F8 MAIN TASK  WAITBND    WAIT FOR I/O OR ECB POST    CCB/ECB ADDRESS: 474628
                         TERMINATOR ACTIVE FOR TASK
F9 MAIN TASK  LTABND     WAITING FOR LTA
FA MAIN TASK  WAITBND    WAIT FOR I/O OR ECB POST    CCB/ECB ADDRESS: 333628
                         JOB CONTROL ACTIVE IN THIS PARTITION
T37 F2 SUB    WAITBND    WAIT FOR I/O OR ECB POST    CCB/ECB ADDRESS: 222628
                         ICCF PSEUDO PARTITION
T38 F2 SUB    READY      READY TO RUN
                         ICCF PSEUDO PARTITION
```

Figure 4-5 (Part 2 of 2). Example: System Loop Analysis Report

PART 2. SDAID TRACE

In order to isolate a problem in a system or in an application program, you need to know the exact sequence of execution steps which have been performed. To find out about the execution steps performed, you have to trace single events. VSE/Advanced Functions offers a program which helps you in tracing specific events in your system. This program is called SDAID.

It provides the possibility to initialize various trace types in order to get the most useful information to analyze particular events in the system.

You may initialize these SDAID traces via promptings in the attention routine (AR), via direct input statements from the AR or a partition, or via job control procedures.

This part of the book describes each of the methods to initialize a trace and how the initialized trace is started/stopped or ended. Trace output examples are shown for each of the trace types.

If the SDAID program is new for you, then please use

========> Chapter 5, "SDAID Overview" on page 5-1

as an introduction.

If you are familiar with the SDAID program conventions, please use the

=======> Summary in Figure 5-2 on page 5-7.

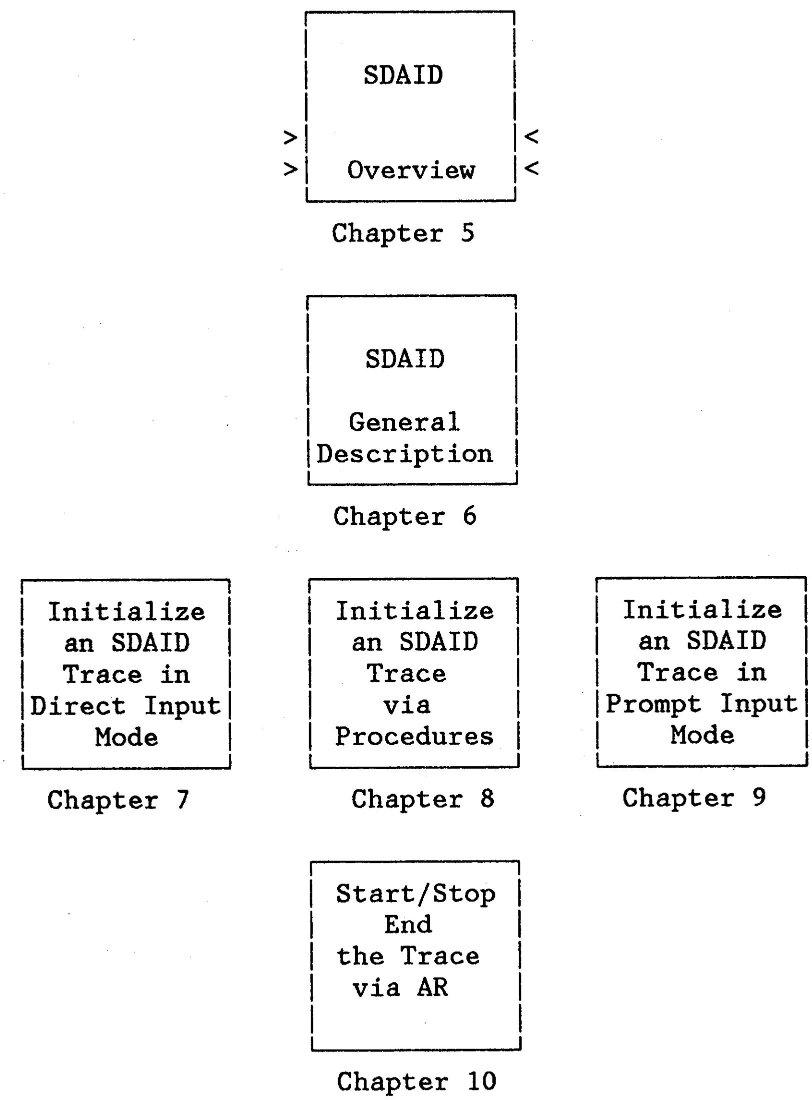
This Summary shows all trace types with references to their format descriptions for the various initialization methods.

An overview of the chapters in this part is shown on the next page.

CONTENTS OF PART 2: SDAID TRACE

SDAID Overview	• Chapter 5, SDAID Overview shows the different ways to initialize SDAID traces in general. Talks about performance and storage requirements.
SDAID Trace General Description	• Chapter 6, SDAID General Description describes the various trace types and gives trace output examples.
Initialize an SDAID Trace Direct Input Mode	• Chapter 7, Initialize an SDAID Trace in Direct Input Mode tells you, how to initialize a certain SDAID trace via direct input mode statements.
Initialize an SDAID Trace via Procedures	• Chapter 8, Initialize an SDAID Trace via a Procedure tells you, how to initialize an SDAID trace with only one job control procedure statement.
Initialize an SDAID Trace Prompt Input Mode	• Chapter 9, Initialize a Trace in Prompt Input Mode tells you, how to initialize a trace interactively in question/answer technique.
Start/Stop End the Trace via AR	• Chapter 10, Start/Stop and End the Trace shows you how to start, stop and end an initialized trace under normal and under exceptional conditions.

CHAPTER 5. SDAID OVERVIEW



This chapter gives an overview of the SDAID program and the various ways to initialize a trace; it includes considerations on the performance and the environment.

Contents

The SDAID Session

Basically you will do two things:

1. Initialize a trace.

2. Start, stop, or terminate the initialized trace.

How to Initialize an SDAID Trace

You initialize a trace with mainly four statement types which have to be submitted to the SDAID program:

1. The SDAID statement to start the initialization process.

2. The OUTDEV specification to define the output device for the trace data.

3. The TRACE statements to define all necessary information for the trace, like the trace type and the area to be traced.

4. A statement which signals the end of the initialization process (/* or READY).

You submit the SDAID statements with one of the following methods:

- Direct input mode in the attention routine or partition.

- Job control procedures in a partition.

- Prompts in the attention routine (AR).

Initialization in Direct Input Mode

In direct input mode the SDAID information is entered in form of commands to the attention routine or as SYSIN statements in a partition.

The SDAID program identifies the mode of initialization via the format of the TRACE and OUTDEV statement. In direct input mode these statements must contain at least one operand.

The following examples show two initialization jobs, one entered in a partition the other one entered via SYSIN.

Example of a trace initialization in direct input mode in the attention routine:

```
sdaid
AR 4C05I PROCESSING OF 'SDAID'  COMMAND SUCCESSFUL.
AR 1I40I READY
outdev tape=280
AR 4C05I PROCESSING OF 'OUTDEV'  COMMAND SUCCESSFUL.
AR 1I40I READY
trace sio unit=009
AR 4C05I PROCESSING OF 'TRACE'  COMMAND SUCCESSFUL.
trace io unit=009 output=ccw
AR 4C05I PROCESSING OF 'TRACE'  COMMAND SUCCESSFUL.
ready
AR 4C05I PROCESSING OF 'READY'  COMMAND SUCCESSFUL.
AR 1I40I READY
```

Example of a trace initialization via direct input mode statements read in from SYSIN:

```
  // EXEC SDAID
  OUTDEV TAPE=280
  TRACE SIO UNIT=009
  TRACE IO UNIT=009 OUTPUT=CCW
  /*
```

The direct input mode is described in
========> Chapter 7, "Initialize an SDAID Trace in Direct Input Mode" on page 7-1.

Initialization via Job Control Procedures

The easiest way to initialize a trace is to use catalogued procedures.
An example of such a trace procedure statement is shown below.

```
// EXEC PROC=SDIO,UNIT=009,TAPE=280
```

The initialization via procedures is described in

========> Chapter 8, "Initialize an SDAID Trace via a Procedure" on page 8-1.

Initialization via Prompts in the Attention Routine

You start the initialization process with the attention routine command 'SDAID'. The necessary trace definitions are given in response to promptings after you entered the TRACE or OUTDEV statement without an operand.

You enter the prompt mode whenever you define these two commands without an operand.

Example of a trace initialization via prompts in the attention routine:

```
sdaid □
AR 4C05I PROCESSING OF 'SDAID'  COMMAND SUCCESSFUL
outdev □
AR 4C08D SPECIFY OUTPUT DEVICE.+
tape □
  ...

□  indicates the ENTER key pressed
```

Note that you enter the prompt mode also if you specify direct input mode statements combined with prompt mode statements like a question mark (? requests the help function of SDAID). The example below shows, how you can combine the two input modes. You would be prompted after the question mark has been processed.

```
TRACE SIO AREA=BG ?
- direct input-->|< prompt mode
```

The prompt input mode is described in

========> Chapter 9, "Initialize a Trace in Prompt Input Mode" on page 9-1.

AR Commands to Start, Stop and End an Initialized Trace

You can start, stop, or end an initialized trace via attention routine (AR) commands. The table below shows an overview of these commands. A more detailed description about stopping, starting, and ending a trace is given in Chapter 10, "Start/Stop and End the Trace" on page 10-1.

STRTSD	Starts SDAID execution; may follow READY or STOPSD.
STOPSD	Interrupts SDAID execution; allowed only after STRTSD.
ENDSD	Ends SDAID session; releases all system resources used by SDAID at any time.

Figure 5-1. AR Commands to Start/Stop and End an Initialized Trace

Trace Type Summary

Find the trace type which you want to initialize in the following trace command summary. You will find page references to the description of the trace type and to the format of the trace initialization statements for the various initialization methods. Localize the section according to the initialization method you choose.

The page references to the available descriptions are under the following headings:

Des Page reference to the description of the trace type

Dir Page reference to the format description for initialization in direct input mode.

Prc Page reference to the format description for the initialization via procedures.

Prp Page reference to the description for the initialization in prompt mode.

Trace Type	Provides a Trace of:	Des	Dir	Prc	Prp
BRANCH	instructions which caused a branch	6-8	7-15	8-8	9-21
BUFFER	of the trace buffer full condition	6-8	7-16	-	9-21
CANCEL	program (main task) cancel or EOJ	6-9	7-17	-	9-21
EXTERNAL	external interrupts	6-9	7-18	-	9-21
INSTRUCTION	selected or all instruction(s) execution	6-11	7-19	8-9	9-22
IO	I/O interrupts	6-12	7-22	8-10	9-23
MONITORCALL	MC instructions	6-13	7-23	-	9-23
PAGING	page faults	6-14	7-24	-	9-24
PGMCHECK	program checks	6-15	7-25	8-12	9-24
PGMLOAD	phase load requests, or actual load	6-16	7-27	8-11	9-25
REGISTER	alteration of general purpose registers	6-17	7-29	8-13	9-26
SIO	SIO instructions	6-18	7-32	8-10	9-28
STORAGE	storage alterations	6-20	7-33	8-14	9-28

Figure 5-2 (Part 1 of 2). Trace Type Summary

Trace Type	Provides a Trace of:	Des	Dir	Prc	Prp
SVC	executed supervisor calls	6-21	7-35	8-16	9-29
VTAMBU	usage of ACF/VTAM buffers	6-21	7-37	-	9-30
VTAMIO	VTAM I/O operations	6-23	7-38	-	9-30

Figure 5-2 (Part 2 of 2). Trace Type Summary

Trace Output

The trace output, an **event record,** is supplied for each occurrence of a traced event, according to your set-up instructions.

You may request the event records to be written to a line printer, onto magnetic tape, or into a wraparound buffer. How to define the output device is described together with each type of initialization process.

Sample event records are shown for each trace type under ========> "The Various TRACE Types" on page 6-6.

Performance Considerations

System Performance Degradation

The tracing of events with SDAID may affect overall system performance. This may especially affect time dependent programs (such as programs controlling MICR equipment or doing input/output via telecommunication lines).

When you invoke SDAID in the AR, console input is blocked during processing of each SDAID command until the final READY command has been processed successfully.

System Performance Degradation Caused by PER Traces

The following SDAID traces use the program event recording feature (PER) of the /370 hardware:

- branch trace (BRANCH)
- instruction trace (INSTRUCTION)
- register alter trace (REGISTER)
- storage alter trace (STORAGE).

For the instruction fetch and storage alter trace the feature allows the SDAID to limit the trace address range via the control registers 11 and 12. For this, the use of the address specification (ADDress= in direct input mode for example) helps to achieve better performance.

No address limitation via the program event recording (PER) feature is possible for the branch and register alteration trace. That means that every branch or register alteration taken in the partition causes an interrupt to the SDAID.

To avoid this time consuming situation the following method may be used as a substitute for the branch trace:

Use the instruction trace with branch defined and limit the address range to a particular amount of storage.

Performance Degradation if LTA and/or SVA is Defined

The system performance degradation caused by the SDAID is less, the smaller the defined address range is. Therefore, you should avoid the use of additional trace ranges by defining LTA and or SVA. The worst performance situation would arise if you define these two areas with the storage alter or instruction trace. The definition of these two areas would expand the region from where the SDAID program is interrupted from the LTA start address to the SVA end address, which takes considerable processing time.

SDAID Space Requirements

SPACE REQUIREMENTS DURING INITIALIZATION IN THE AR

The SDAID set-up phases are loaded into the system GETVIS space. The SDAID set-up phases require approximately 100K bytes (K equals 1,024) of virtual storage. When initialization is complete (the READY command is processed successfully), that GETVIS space is released.

SPACE REQUIREMENTS DURING INITIALIZATION IN A PARTITION

Beside the GETVIS space of 100K bytes the phase SDAID (called via EXEC SDAID) requires approximately 16K bytes of partition virtual storage. This is significantly less than the minimum VSE partition size. Therefore SDAID will run in any foreground or background partition.

SPACE REQUIREMENTS FOR SDAID EXECUTION

When the READY command is entered, SDAID allocates and fixes a certain number of pages in processor storage for SDAID execution. The amount of storage required for SDAID execution depends on the combination of trace operations that you request and on the size of the output buffer (specified in the OUTDEV command).

For simple applications (OUTPUT parameter not used) SDAID requires approximately 20K bytes. More complex operations require approximately 48K bytes.
If TAPE=cuu is specified, but the buffer parameter is not used, a default buffer size of 3K bytes is used.

If a buffer is specified in the OUTDEV command, the size of that buffer has to be added to the space requirement. Which means, that you may run out of storage if you define too large a buffer.

SPACE REQUIREMENTS FOR SDAID EXECUTION, SUMMARY

Basic requirement for SDAID execution:	12K
Additional requirements	
Per specified trace:	2K
If BUFFER=nn is specified:	2K + buffer size
If OUTPUT is used in the trace commands:	12K
If OUTDEV is a printer:	8K
If OUTDEV is a printer and OUTPUT is defined:	16K

For an SDAID execution in E- or VM mode, the system loads the SDAID execution phases into the system GETVIS area.

In /370 mode, an area between the dynamic supervisor area and the begin address of the private user partitions is allocated for SDAID. The size of the SDAID area allocated by default is 48K bytes.

The size of the SDAID area may be increased during IPL using the SDSIZE parameter of the SYS command. You would do this for example if you plan to define a large buffer size.

During execution, SDAID fixes a certain number of pages in processor storage. As a result, the number of page frames available to VSE for the execution of programs in virtual mode is reduced, which may affect overall system performance.

Number of Traces per Session

In prompt and direct input mode: The number of TRACE commands that you can submit per session depends on the types of the specified traces and the requested trace options. For each TRACE command, SDAID builds at least one TRACE command control block; for some it builds two such blocks or even more as shown below. The program can build (and use) a maximum of ten TRACE command control blocks per session. The number of blocks per TRACE command is:

Type of trace	PHase=phasename not specified	PHase=phasename specified
PGMLOAD	2	2
PAGING	2	4
all others	1	3

If the traces that you requested require more than ten trace control blocks, the program ignores the TRACE command that was submitted last and informs you about this action with a message.

Via procedures: Only one procedure statement is possible which can create one or more TRACE commands.

CHAPTER 6. SDAID GENERAL DESCRIPTION

```
                 ┌────────────┐
                 │   SDAID    │
                 │            │
                 │  Overview  │
                 └────────────┘
                   Chapter 5

                 ┌────────────┐
                 │   SDAID    │
                 │            │
                >│  General   │<
                >│Description │<
                 └────────────┘
                   Chapter 6

┌────────────┐   ┌────────────┐   ┌────────────┐
│ Initialize │   │ Initialize │   │ Initialize │
│  an SDAID  │   │  an SDAID  │   │  an SDAID  │
│  Trace in  │   │   Trace    │   │  Trace in  │
│Direct Input│   │    via     │   │Prompt Input│
│    Mode    │   │ Procedures │   │    Mode    │
└────────────┘   └────────────┘   └────────────┘
  Chapter 7        Chapter 8        Chapter 9

                 ┌────────────┐
                 │ Start/Stop │
                 │    End     │
                 │ the Trace  │
                 │   via AR   │
                 └────────────┘
                   Chapter 10
```

This chapter contains a general description of all SDAID initialization definitions, and output examples for all trace types.

The format of the commands and definition examples are shown under the descriptions for the various initialization methods.

Contents

DEFINE THE OUTPUT DEVICE

The following output destinations can be defined when you initialize an SDAID trace:

- Printer
- Tape
- Wrap-around buffer
- Wrap-around buffer and printer
- Wrap-around buffer and tape

You define the output destination for the event records in prompt and direct input mode via the 'OUTDEV' statement. If you use a procedure to initialize a trace, the 'BUFFER=', 'Tape=', or 'Printer=' specifications are used to accomplish an OUTDEV definition.

The OUTDEV command is required with all initialization methods.

Printer Defined as Output Destination

If a printer is defined for the output, SDAID writes the event records on a line printer at the time the particular events occur.

If any program in the system provides output directly to the printer, it will be mixed with the SDAID output. You can avoid this by selecting a different printer, or by using the OUTDEV buffer option, as described below.

Tape Defined as Output Destination

If you define a magnetic tape as output device, the SDAID acquires a buffer of 3K bytes and writes event records to the available tape volume in blocks of 3K bytes whenever the buffer becomes full or a 'STOPSD' or 'ENDSD' command is being processed.

SDAID writes the buffers successively on tape and one trace entry can extend over several buffers. This is the reason why, if you rewind the tape after one or more buffers have been written, it seems that a part of a trace entry is lost.

Buffer Defined as Output Destination

```
                  ┌──────────> Tape or Printer
                  │
        on defined Event
                  │
               ┌─────────┐
  Trace Event  │         │
 ─────────────>│ Buffer  │
   Records     │         │
               └─────────┘
                  │
           DUMP Command
                  │
                  └──────────> Tape
```

Figure 6-1. Overview, Trace Events into a Buffer

A buffer in storage may be defined to store the trace event records. During tracing, SDAID stores one event record after the other. When the buffer becomes full, SDAID wraps around and continues to write event records at the beginning of the buffer overwriting previously stored records. A buffer used in the described way is also called a <u>wrap around buffer</u>.

If the contents of the buffer is written to an output device, SDAID starts with the oldest trace record that has not been overwritten and ends writing with the most recent one. The buffer is immediately reset after it has been written to the output device. Note that one trace entry may spread over several buffers. If you rewind the tape after one or more buffers have been written, it may seem as if a part of the trace entry is lost.

You **define the point of processing** when the buffer has to be printed or written to tape in prompt and direct input mode via the OUTPUT definition for a certain trace event (OUTPut=BU), or if you use a procedure to initialize a trace, via the 'BUFFOUT=' keyword operand (BUFFOUT=CANCEL for example).

The information contained in the buffer can also be retrieved with the attention routine DUMP command. Please find further information under "Print the SDAID Produced Tape" on page 4-6.

You **define the size** of the buffer in number of blocks of 1K bytes.

The possible buffer size which you can define depends on the device type of the buffer output device.

If you define a tape device, the possible size may vary from 3K to 32K bytes.

You can request a buffer size from 3K to 99K bytes in case a printer device or no output device is definition.

> **Note:** The buffer is a part of the SDAID area. A definition of a buffer may cause a bottleneck in SDAID storage. Please find more detailed information under "Space Requirements for SDAID Execution" on page 5-10.

Steps to Define a Wrap-around Buffer

Consider, for example, you want to collect the event records of an instruction trace into a 6K bytes buffer as long as no program check occurs. The buffer contents should be written onto a tape when a program check interruption occurs.

Perform the following steps:

- Define the buffer and the tape device with the OUTDEV direct input and prompt mode command or the procedure statement in the form of: 'BUFFER=6 TAPE=280'
- Define the instruction trace.
- Define the PGMCHECK trace with OUTPUT=BUFFER (or define BUFFOUT=PGMC in the procedure statement).

Please find the formats of the output definitions under the following sections according to the input mode used.

- =====> "Define the Output Device in Direct Input Mode" on page 7-9.

- =====> "BUffer=, Printer=, Tape= Keyword Operands" on page 8-18.

- =====> "Output Device Definition in Prompt Mode: OUTDEV Command" on page 9-16.

THE VARIOUS TRACE TYPES

SDAID offers you various trace types so that you get the information most suitable to solve a problem in your computing environment.

This section describes all SDAID trace types with their SDAID default values and shows trace event record examples. The shown output may be written into a buffer or on a tape or printer device according to your output device specification.

Note that, if you operate in 370 mode, the event records include the space-id in front of the appropriate partition-id.

Summary of Trace Types

Please find the command, and a short description in the table below. The page references in this table help you to find the description and an output example of each trace type.

Trace Type	Provides a Trace of:	Page
BRANCH	Successfully executed branch instructions	6-8
BUFFER	Output the trace buffer when it is full	6-8
CANCEL	Program (main task) cancel or EOJ	6-9
EXTERNAL	External interrupts	6-9
INSTRUCTION	Selected or all instruction(s) execution	6-11
IO	I/O interrupts	6-12
MONITORCALL	MC instructions	6-13
PAGING	Page faults	6-14
PGMCHECK	Program checks	6-15
PGMLOAD	Phase load requests, or actual load	6-16
REGISTER	Contents or alterations of 1 to 8 registers	6-17
SIO	SIO instructions	6-18
STORAGE	Storage alterations	6-20

Figure 6-2 (Part 1 of 2). Trace Type Summary

Trace Type	Provides a Trace of:	Page
SVC	Executed supervisor calls	6-21
VTAMBU	Usage of ACF/VTAM buffers	6-21
VTAMIO	VTAM I/O operations	6-23

Figure 6-2 (Part 2 of 2). Trace Type Summary

BRANCH Trace

A branch-instruction trace provides an event record for every branch taken, within the defined range of the trace. The area definition is required in this trace statement.

An example of the output is shown in Figure 6-3.

BRANCH Trace Output Example

```
    ┌──event identifier
    │        ┌────── partition–id of the active task     ┌─first four bytes of
    │        │                                           │ old PSW
    │        │                                           │
  BRANCH 47  1 BG 102A4B ==>102A4C  47F0C240          PSW=470D0000  PHASEx+00004E
         │   │    │           │      │                                        │
         │   │    │           │      └── instruction                          │
         │   │    │           └── address of instr    offset if tracing ──┘
         │   │    │               being branched to    in phase is defined
         │   │    └── address of branch instruction
         │   └── space–id (370–mode only)
         └── op code of instruction causing the event
```

Figure 6-3. BRANCH Trace Event Record

BUFFER Trace

The buffer trace dumps the contents of the SDAID wrap-around buffer to the output device (printer or tape) when the buffer is full.

The buffer trace can be used only if you have also 'Printer' or 'Tape' specified in the OUTDEV command or in a procedure.

The buffer trace output is the collection of all trace records contained in the buffer when a buffer overflow occurs. These trace records are written sequentially.

CANCEL Trace

This trace provides an event record when the main (or only) task of the program defined with the AREA definition either is canceled or reaches EOJ.

You may use this trace type combined with additional output definitions to get more reasonable information at the time of a cancel or EOJ condition.

For example, use the cancel trace type to get the buffer or areas of interest together with the cancel event record recorded.

An example of the cancel trace output is shown in Figure 6-4.

CANCEL Trace Output Example

```
  ┌── event identifier
  │     ┌────── partition-id of the active task      ┌ first four bytes of
  │     │                                            │ old PSW
  │     │                                            │
 CAN   1 F4            CODE=24                     PSW=040C2000
       │                  │
       │                  └── cancel code
       └──space-id (370-mode only)
```

Figure 6-4. CANCEL Trace Event Record

EXTERNAL Trace

The external trace provides information concerning the occurrences of external interrupts such as pressing the external interrupt key. You may define one or more of the following external interrupt types:

- KEY (manual external interrupt key)
- SIGNAL (external signal 2-7, malfunction alert, emergency signal, external call)
- TIMER (interval timer, TOD clock sync check, clock comparator, CPU timer)

SDAID Default Value

If you do not define the type of interrupt, all external interrupts are traced.

The format of a printed external-interrupt trace event record is shown in Figure 6-5.

EXTERNAL Interrupt Trace Output Example

```
  ┌── event identifier
  │    ┌──── Partition-id of the active task    ┌─ first four bytes of
  │    │                                        │  old PSW
  │    │                                        │
 EXT   1 F4 00369C ==>000A96      CODE=0040     PSW=070E2000
       │    │        │               │
       │    │        │               └── interrupt code
       │    │        └── instruction address in new PSW before
       │    │            modification by SDAID
       │    └── address of instruction last executed
       └── space-id (370 mode only)
```

Figure 6-5. EXTERNAL Interrupt Trace Event Record

INSTRUCTION Trace

An instruction trace provides information for instructions executed in the area defined for the trace. You may select certain types of instructions to be traced by defining the instruction operation codes or all types by defining an asterisk (*). Moreover, the trace can be defined to record all branch instructions, without regard to whether the branch is taken or not.

Note the difference from the branch trace, which only records the branches actually taken.

The format of a printed event record for an instruction trace is shown in Figure 6-6.

INSTRUCTION Trace Output Example

```
 ┌── event identifier
 │       ┌────── partition-id of the active task   ┌ first four bytes of
 │       │                                         │ old PSW
 │       │                                         │
INST  45 1 BG 1B7CD2 ==>1B7D0C  45D0730C          PSW=470D1000 $IJBSDMP+003ADA
      │  │     │          │        │                                       │
      │  │     │          │        └── instruction   offset within ────────┘
      │  │     │          └── target address of      the defined phase
      │  │     │              branch instruction     (PH=$IJBSDMP)
      │  │     └── instruction address
      │  └── space-id (370 mode only)
      └── op code
                                                       ┌── PSW status part
                                                       │
INST  0A 1 BG 1B7D0C ==>00916E  0A00              PSW=040C0000 $IJBSDMP+003B14
                            │
                            └── target address of
                                SVC instruction

INST  91 1 BG 1B7D0E            91801002          PSW=470D3000 $IJBSDMP+003B16
```

Figure 6-6. INSTRUCTION Execution Event Record

The example has been created by the following direct input mode trace statement:

TRACE INST=* AR=BG PH=<IJBSDMP

IO Trace (I/O Interrupt)

The IO trace collects information about I/O interrupts of all I/O devices of the whole VSE/Advanced Functions system.

You may limit the I/O operations to be traced to a partition or to the supervisor. Another limitation is to define a particular unit, control unit, or channel to be traced.

SDAID Default Value

If you do not define a partition or the supervisor, all tasks of the system are traced. All I/O devices are traced if you do not specify any of the I/O definitions.

The format of an I/O-interrupt event record is shown in Figure 6-7.

Note: Due to the fact that no tables are available in the supervisor for TP status modifier commands, the output of TP channel programs (e.g.VTAM) may be incomplete.

IO Interrupt Trace Output Example

```
┌── event identifier
│          ┌──────────── partition-id of partition active at IO interrupt time
│          │IO interrupt time.
IO   0130 1 AR 003698 ==>00095C        CSW=00003F480C000000
      │   │        │        │
      │   │        │        └── address in new PSW before modification by SDAID
      │   │        └── address of interrupted partition and instruction
      │   │
      │   └────── space-id (370mode only)
      │
      └── device address
```

Figure 6-7. I/O-Interrupt Trace Event Record

MONITORCALL Trace

The monitor call trace provides information about monitor call instruction executions.

You may define all (defined via an asterisk (*)) or up to eight mc (monitor classes) in hexadecimal notation of the MC instructions to be traced. An event record is provided when an executed MC instruction has a monitor class which matches any of the specified classes.

You may specify any valid monitor class; however, SDAID ignores a specification of class 2.

The format of an MC instruction trace event record is shown in Figure 6-8.

MONITORCALL Trace Output Example

```
 ┌── event identifier
 │          ┌────── partition-id of the active task      ┌─ PSW status part
 │          │                                            │
MON   05 1 F4 176098 ==>0057F8     CODE=00000005     PSW=072D3000  PHASEx+00004E
       │ │      │        │                │                              │
       │ │      │        │                └── monitor code               │
       │ │      │        └── address of instruction getting control      │
       │ │      └── address of instruction causing event                 │
       │ └── space-id (370 mode only)offset                        if tracing
       └── monitor class                                    in PHASEx is defined
```

Figure 6-8. MONITORCALL Trace Event Record

PAGING Trace

The paging trace provides information about page fault events. An event record is written for every occurrence of such an event within the specified trace range.

The format of a paging event record is shown in Figure 6-9.

PAGING Trace Output Example

```
 ┌─── event identifier                    ┌─── address causing page fault
 │         ┌─── partition-id              │
 │         │                              │
PAGING   1 F4  176BA2    40402000   LOC   =185800        PSW=072D300
         │       │          │                             │
         │       │          │                             └─ first 4 bytes of
         │       │          │                                old PSW
         │       │          └─── instruction causing page fault
         │       │
         │       └─── address of instruction causing page fault
         │
         └─── space-id (370 mode only)
```

Figure 6-9. PAGING Trace Event Records

PGMCheck Trace (Program Check)

The program check trace provides information on the occurrence of program check interrupts.

You may limit the trace operation by defining certain program interruption codes. Up to 16 program interruption codes of a value lower than X'40' in hexadecimal notation may be defined.

SDAID writes an event record only if the interrupt code returned by the system matches one of the specified interrupt codes.

If you do not want to limit the trace recording to a specific interrupt code, define an asterisk (*) to trace all program checks except the page fault interrupts.

For a discussion of program interrupt codes, refer to the applicable Principles of Operation manual.

The format of a program-check event record is shown in Figure 6-10.

PGMCHECK Trace Output Example

```
 ┌─── event identifier
 │          ┌────── partition-id of the active task      ┌─ PSW status part
 │          │                                            │
PGMCHK 07 1 BG 1012AC ==>0057F8  4F320000     PSW=471D0000  PHASEx+00004E
       │  │       │          │         │                               │
       │  │       │          │         │      offset if tracing in ───┘
       │  │       │          │         │      PHASEx is defined
       │  │       │          │         └─── instruction causing the event
       │  │       │          └─── address in new PSW before change by SDAID
       │  │       └─── address of instruction causing the event
       │  └─── space-id (370-mode only)
       └─── program interrupt code
```

Figure 6-10. Program Check Trace Event Record

PGMLOAD (Fetch/Load) Trace

The program load trace provides information about all program load events.

Such a program load event can be one of the following:

- Phase load request
- Phase fetch request
- Actual fetch/load operation

You may limit the trace recording if you define that only the fetch/load request or only the actual fetch/load operation is to be traced.

Moreover, you can limit the trace data collection to the load events of a certain phase.

SDAID Default Values

If you do not define the kind of the fetch/load request, both the request and its handling is traced. All phases are traced if you do not define a specific phase.

The format of a program load event record is shown in Figure 6-11.

PGMLOAD Trace Output Example

```
          ┌── activity indicator
          │           ┌── address of load instr
          │           │      ┌── load-SVC     ┌─── register number
          │           │      │                │   ┌─── contents of register
          │           │      │                │   │
   PGML   REQ 1 F3 192078 SVC=04  $IJBSLA   R00=00000000 R01=00192648   PSW =nn
                                  └── phase name
   PGML   HDL 1 F3                $IJBSLA   LP= 001E1000 EP= 001E1000 SVA LIB=IJSYSRS
SLIB=SYSLIB
      │         │ │                              │              │      │  │
      │         │ └─── partition-id              │              │      │  │
      │         └─── space-id (370-mode only)    │              │      │  │
      └─── event identifier       load address ──┘              │      │  │
                                         entry point address────┘      │  │
                                  indicates phase is contained in SVA ─┘  │
                                  library and sublibrary name  ───────────┘
```

Figure 6-11. PGMLOAD Trace Event Records

REGISTER Alteration Trace

The register alteration trace provides information about instructions which alter defined registers. At least one register number must be defined. Up to eight general registers may be selected for tracing. The register number has to be defined in hexadecimal notation.

You may limit the trace operation by defining a certain pattern. The pattern can have the length of up to four bytes and is defined in hexadecimal notation. Only those instructions are recorded which alter the contents of the named register(s) so that the resulting value is equal to the value specified by the pattern.

You may also specify a pattern range with a pattern pair defining a lower and an upper limit (value1 value2). In this case all those instructions are traced which alter the named register(s) to a value which is equal to or within the limits of that pattern range.

Note, that, if the first value is higher than the second one, the actual defined range starts with the higher value1, extends up to the address FFFFFFFF, continues with address 00000000 and ends with the value defined in value1.

SDAID Default Values

You may omit the pattern definition; this causes that each alteration of the contents of the defined register to be traced.

The format of a printed standard event record for a register trace is shown in Figure 6-12 on page 6-18.

REGISTER Alteration Trace Output Example

```
┌── event identifier
│    ┌── op code
│    │ ┌── space-id (370-mode only)
│    │ │ ┌── partition-id
│    │ │ │    ┌── instruction address
│    │ │ │    │                 ┌── instruction        PSW status part─┐
│    │ │ │    │                 │       ┌─ defined pattern              │
│    │ │ │    │                 │       │                               │
REG  41 1 BG 100A8C           41A00FFF  PATT=00100078 001859E3   PSW=470D0000
  GR 0-7 00000003 001001A4 00000380 00100917 00001194 00000003 00000006 00101A1
  GR 8-F 00101684 00109060 00000FFF 001000E0 00102800 00103800 00000330 8010082

         └────────────────general register contents─────────────────────────┘
```

Figure 6-12. Register Alter Trace Event Record

SIO Instruction Trace

The SIO instruction trace provides a trace event record for every executed start I/O (SIO) or start I/O fast release (SIOF) instruction.

You may limit the trace operation by defining selected tasks or by defining a unit, a control unit, or a channel address. A partition or the supervisor may be defined as the traced tasks.

SDAID Default Value

If you do not define a partition or the supervisor, all tasks of the system are traced. All I/O devices are traced if you do not specify any of the I/O definitions.

The format of a printed standard SIO instruction event record is shown in Figure 6-13.

Note: Due to the fact that no tables are available in the supervisor for TP status modifier commands, the output of TP channel programs (e.g.VTAME) may be incomplete.

SIO Trace Output Example

Trace Event Record before SIOF Instruction

```
┌── event identifier
│         ┌── space-id (370-mode only)
│         │
SIOF-1 0130 1 AR
       │    │
       │    └─── partition owning the issued start I/O
       └─── device address
```

Trace Event Record after SIOF Instruction

```
SIOF-2 0130 1 AR    CC=1  CSW=00027B20 2C000000
                    │     │
                    │     └── CSW contents if CC is other than 0
                    │
                    └── condition code
```

Figure 6-13. SIO (Start I/O) Trace Event Record

STORAGE Alteration Trace

The storage alteration trace writes an event record whenever a program alters the contents of a defined storage area. Storage alterations caused by I/O operations are not recorded.

With the AREA specification you define those tasks which you want to watch. You would usually define ALL for the AREA definition to have all tasks of the system observed. WHERE the investigated area is, is specified via the ADDRESS definition.

You may limit the trace operation by defining a certain storage pattern. If you define such a pattern a trace event record is written only when a storage area is set to the specified value. Specify the pattern in hexadecimal notation. The pattern can be up to four bytes long.

If you specify an odd number of digits, a zero is inserted to the left of the first specified hexadecimal digit.

In order to get meaningful information, define the area to be investigated via the ADDRESS definition in the length of the defined pattern.

SDAID Default Value

You may omit the pattern definition. This causes that each alteration of the defined storage area is traced.

An example of an storage alteration trace event record is shown in Figure 6-14.

STORAGE Alteration Trace Output Example

```
  ┌─── event identifier
  │      ┌─── space-id (370-mode only)
  │      │ ┌─── partition-id                          PSW status part ┐
  │      │ │                                                          │
STOR   F3 1 BG 100FA4   F384B277B202 STAD=1010F1  PATT=F0F0   PSW=471D0030
       │                 │                │           │
       │                 │                │           └───specified pattern
       │                 │                └─── address of target field
       │                 └─── instruction in hex
       └─── op code
```

Figure 6-14. Storage-Alter Trace Event Record

SVC Trace (Supervisor Call)

The supervisor call trace provides information about one, several, or all SVC instructions executed in the defined area.

You have to define at least one or may define up to 16 different SVC codes or an asterisk (*) specifying that all supervisor call instructions are to be traced. Specify the SVC code in hexadecimal notation.

For a list of valid SVCs, refer to Appendix B, "Supervisor Call Code (SVC) List" on page B-1.

A typical SVC trace event record is shown in Figure 6-15.

SVC Trace Output Example

```
┌─── event identifier
│     ┌─── space-id (370-mode only)
│     │ ┌────── partition-id of the active task       PSW status part ─┐
│     │ │                                                                │
SVC 07 1 BG 1000BE ==>00A36  R00=00100078 R01=00101C99 R15=00170FFF PSW=471D0030
    │       │         │        │
    │       │         │        └─── contents of registers 0, 1, 15
    │       │         └─── address of supervisor instruction receiving control
    │       └─── address of instruction causing the event
    └─── supervisor call interrupt code (SVC)
```

Figure 6-15. SVC Trace Event Record

VTAMBU Trace (VTAM Buffer)

The VTAM buffer trace provides an event record each time ACF/VTAM uses one of the buffers in its buffer pool.

The format of a printed standard VTAMBU event record is shown in Figure 6-16 on page 6-22.

Note: For the VTAMBU trace you must condition ACF/VTAM either by entering an ACF/VTAM MODIFY command or by specifying the TRACE operation when you start ACF/VTAM.

For information on conditioning ACF/VTAM, see the ACF/VTAM Diagnostic Guide publication.

At the end of an SDAID session, end ACF/VTAM conditioning with the appropriate MODIFY command.

VTAMBU VTAMIO Trace Output Example

```
   ┌── (A) ┌── partition-id
VT-IO  0130 1 BG 003230 ==>0006AC  CSW =000040 000039A80C000000    PSW=030F2000
            └── space-id (370-mode only)
   ┌── (B)
VT-SVC   31 1 BG 10012E ==>000B08      R00=00100A1C R01=00101F0F R15=00180FFF
   ┌── (C)
VT-SIO 0130 1 BG             CC=0
                                    ┌── sequence number of trace record
VTAM  BUFFER POOL USE  SEQ.NO = 00000001  DAY 097   TIME 19:47:10
      VF IN USE=00005 MAX ALLOC=00005 MAX WAIT=0000 EXPAND=0000 MAX AVAIL=0000 CUR AVAIL=0000
      VP IN USE=00089 MAX ALLOC=00098 MAX WAIT=0000 EXPAND=0000 MAX AVAIL=0000 CUR AVAIL=0000
      │
      │         (D)             (E)          (F)         (G)           (H)            (I)
      └── pool-ID; for details see ACF/VTAM Debugging Guide
```

(A) For description see Figure 6-7 on page 6-12 (I/O interrupt trace).

(B) For description see Figure 6-15 on page 6-21 (SVC trace).

(C) For description see Figure 6-13 on page 6-19 (SIO trace).

(D) Number of buffers (pages for VF and VP) in use when buffer usage was recorded.

(E) Maximum number of buffers (pages for VF and VP) at any point in time up to the point buffer usage was recorded.

(F) Maximum number of requests for buffers (pages for VF and VP) that were queued at any point in time up to the point buffer usage was recorded.

(G) Number of times the buffer (page) pool was expanded up to the point buffer usage was recorded.

(H) Number of buffers (pages for VF and VP) that were in the pool when buffer usage was recorded.

(I) Maximum number of buffers (pages for VF and VP) that were in the pool at any point in time up to the point buffer usage was recorded.

Figure 6-16. VTAMIO/VTAMBU Trace Record

VTAMIO Trace

The VTAMIO trace combines the following trace types:

- SVCs with codes X'31' and X'35'
- SIO (or SIOF) instructions
- I/O interrupts

Please find a description of the traces involved under the various trace type descriptions (SVC trace, SIO trace and I/O trace).

SDAID Default Value

All I/O devices are traced if you do not specify any of the I/O definitions.

The format of the printed VTAMIO event records is shown in Figure 6-16 on page 6-22.

DEFINE THE TASKS TO BE TRACED: AREA DEFINITION

AREA=
partition-id|SUPVR|ALL|space-id

You define the tasks for which processing is traced via the AREA definition. The following task definitions are possible:

- partition-id
- SUPVR
- ALL
- space-id.

Only one of these definitions is possible.

partition-id defines that the activities of the tasks running with this partition-id are observed. The following partition-id's may be defined:

BG|BGR|Fn|FnR

If, for example, BG is specified, the steps executed by the BG main task and by all attached subtasks are traced. AREA=BG does not necessarily mean that tracing is restricted to the BG space.

SUPVR defines, that the activities of the supervisor are traced.

ALL defines, that the activities of all tasks in the VSE/Advanced Functions system are traced.

space-id defines that the tasks of the defined space are to be traced (in /370 mode). If no partition is allocated in this address space, a message is issued (4C14D). The following space-id's can be specified:

1 2 3 R

1 2 3.. stand for the spaces 1 to 3
R ..... stands for the real address space

Notes

- The AREA specification has to be set for all TRACE definitions, except for the traces: BUFFER, EXTERNAL, IO, SIO, and VTAMBU.
- For IO and SIO traces, AREA=ALL|BG|Fn|SUPVR may be specified and AREA=ALL is the default value.
- For the VTAMIO trace, only AREA=BG|Fn can be specified. BG or Fn denotes the partition where VTAM is active. In that case only the I/O activities of that partition are traced.

DEFINE THE STORAGE TO BE TRACED: OFFSET, ADDRESS, PHASE

Whereas the AREA definition allows you to specify the tasks that you want to observe, the storage definition determines the storage region that is to be investigated. The storage definition has to be specified in accordance to the task definition. For example, if you specified a partition-id for AREA, the definitions OFFSET,ADDRESS and PHASE can be used.

The following sections tell you the valid storage definitions in accordance with the various task specifications.

Storage Definition when Partition-Id Defines the Tasks

You may use one of the following storage area definitions if the tasks to be traced are defined by a partition-id (for example, by defining AREA=BG).

```
AREA=partition-id

[OFFSET=][LTA|SVA|(LTA SVA)]                       |
[ADDRESS=][LTA|SVA|(LTA SVA)]                      |
[PHASE= [OFFSET=0:*]][LTA|SVA|(LTA SVA)]

 OFFSET=0:*   .........      default
```

OFFSET= limits tracing to a certain address range via offsets relative to the partition start address. OFFSET=0:* is the default specification when AREA is define by a partition-id. See "OFFSET Definition" on page 6-29.

ADDRESS= limits tracing to a certain address range within the storage allocated to VSE/Advanced Functions. See "ADDRESS Definition" on page 6-28.

PHASE= limits tracing to the area of a certain phase residing in the partition or in the SVA.

When you specify the PHASE= operand the operand OFFSET= can be used in addition. In this case, OFFSET= defines a trace area within the phase. If you omit the OFFSET= definition the whole storage area of the phase is traced. See "PHASE Definition" on page 6-29.

LTA SVA in addition to the areas defined by OFFSET, ADDRESS or PHASE, this specifications allow tracing also in the logical transient and shared virtual area. You may enter

LTA, SVA or both of them. See "LTA and SVA Operands" on page 6-31.

SDAID Default Values

If you do not specify OFFSET, ADDRESS, or PHASE, the whole area allocated to the defined partition is traced. (OFFSET=0:* is the resulting definition in direct input mode notation).

If you do not specify an offset range after the PHASE definition, 0:* is assumed.

Storage Definition when the Supervisor is Traced

You may use one of the following storage area definitions if the task to be traced is the supervisor. For example if AREA=SUPVR is defined (direct input mode definition).

```
AREA=SUPVR

[OFFSET=|ADDRESS=|PTA]

 OFFSET=0:* ........  default
```

OFFSET= defines an offset range within the area allocated to the supervisor. See "OFFSET Definition" on page 6-29.

ADDRESS= defines an address range in the storage allocated to VSE/Advanced Functions. See "ADDRESS Definition" on page 6-28.

PTA defines that only the supervisor activities in the PTA are traced. PTA stands for physical transient area.

SDAID Default Value

If you do not define OFFSET, ADDRESS, or PTA, the whole area allocated to the supervisor is traced (OFFSET=0:*).

Storage Definition when Space-id or ALL is Defined

```
AREA=space-id|ALL

[ADDRESS=]

 ADDRESS=0:* ........  default
```

ADDRESS= defines an address range in the storage allocated to VSE/Advanced Functions. See "ADDRESS Definition."

SDAID Default Value

If you do not define an address range, the whole area defined with AREA= is traced (ADDRESS=0:*).

ADDRESS Definition

```
ADDRESS=
addr1:addr2|addr1:*|0:*
```

You can limit the trace to an area within the storage allocated to VSE/Advanced Functions with the ADDRESS definition.

addr1:addr2 defines a trace address range in hexadecimal notation. 'addr1' and 'addr2' can be any address in the virtual storage defined for VSE/Advanced Functions.

'addr2' must be higher than or equal to addr1.

addr1:* defines a trace address range starting with 'addr1' up to the end of the VSE/Advanced Function storage.

0:* defines that the whole storage allocated to the VSE/Advanced Functions is defined as trace storage area.

OFFSET Definition

```
OFFSET=
reladdr1:reladdr2|reladdr1:*|0:*
```

You can limit the trace to a storage area within the defined partition, supervisor, or phase with the OFFSET definition.

reladdr1:reladdr2 defines a trace address range in hexadecimal notation.

'reladdr1' can be any relative address within the partition, supervisor or phase defined by the AREA operand.

'reladdr2' must be higher than or equal to reladdr1.

reladdr1:* defines a trace address range starting with 'reladdr1' up to the end of the partition, the supervisor or the phase.

0:* defines that the whole storage allocated to the partition, the supervisor or the phase is defined as trace storage area.

If AREA=BGR or AREA=FnR is specified, the OFFSET operand refers to the partition allocations in the real address space.

PHASE Definition

```
PHASE=
phase-name [OFFSET=]
```

phase-name defines the trace storage area via a phase name. Only the storage area occupied by that phase is traced.

You may use the OFFSET operand to limit the trace area within the defined phase. See "OFFSET Definition" above.

The SDAID program has to localize the defined phase if it does not reside in the SVA. Therefore the trace has to be started (STRTSD command) before the phase is loaded into the partition. Note that some phases which are located via LPT hardpointers cannot be used to limit the trace operation.

LTA and SVA Operands

LTA|SVA|(LTA SVA)

LTA defines that the logical transient area is added to the storage area to be investigated.

SVA defines that the area allocated to the shared virtual area is added to the storage area to be investigated.

Note:

Use these two specifications only in exceptional cases. The keywords LTA and SVA may degrade the performance for the following trace types:

- Branch
- Register
- Instruction
- Storage

Please find more detailed information about the performance under "Performance Degradation if LTA and/or SVA is Defined" on page 5-9.

You can avoid the performance degradation when you define the areas explicitly for example via the ADDRESS definition.

DEFINE ADDITIONAL TRACE OUTPUT

OUTPUT=

definition|(definition1 definition2)

You may specify additional trace output with the OUTPUT definition. This additional trace information is recorded together with the trace event records. For example, you may define that a dump of defined control blocks or address ranges be recorded in addition to the trace event record.

You may select one or more definitions for a specific trace type.

Please find a summary of all OUTPUT definitions in Figure 6-17 on page 6-33. The page numbers in the third column refer you to the description of the OUTPUT definitions in this chapter.

Definition	Records/prints in addition:	Page
BUFFER	Contents of SDAID output buffer	6-34
CCB	CCB or IORB (TRACE=IO, SIO, or VTAMIO only)	6-34
CCW	CCW's (TRACE=IO,SIO, or VTAMIO only)	6-34
CCWD=nnnn	CCW's plus nnnn bytes of data (TRACE=IO, SIO, or	6-34
COMREG	Partition communication region	6-36
CREG	Control registers	6-34
DUMP	Virtual or real storage	6-37
FREG	Floating point registers	6-40
GREG	General purpose registers	6-40
IOTAB	PUB, LUB, ERBLOC, CHANQ	6-41
LTA	Logical transient area	6-41
LOWCORE	Processor storage from zero to X'BC'	6-42
PTA	Physical transient area	6-42
PIB	Partition information block	6-43
PTAB	Partition related control blocks	6-43
SUPVR	Supervisor, GREGs and CREGs	6-43
SYSCOM	System communication region	6-44
TOD	Time-of-Day clock	6-44
TTAB	Task related control blocks	6-45

Figure 6-17. OUTPUT Definition Summary

Record the Trace Buffer

```
OUTPUT=
       BUFFER
```

BUFFER writes the contents of the SDAID buffer to the device specified with the OUTDEV statement. The buffer is written immediately after the associated event has been recorded in the buffer.

Note: Specifying BUFFER as one of a number of output options may result in the original event record(s) getting lost due to the wraparound recording technique used.

Record CCB or IORB

```
OUTPUT=
       CCB
```

CCB records or prints the contents of either the CCB or the IORB (input/output request block) plus the TOD (time of day clock). This output option is meaningful only with an IO or SIO trace request.

Record CCW

```
OUTPUT=
    CCW|(CCWD=nnnn)
```

CCW records or prints the available channel program (Channel Command Word chain) when the trace type is SIO. In case of an IO trace, only the CCWs which refer to transferred data are recorded or printed.

The output contains also the first 12 and the last four bytes of the associated data, the CCB, and the TOD (time of day clock).

Specifying this output option for an event other than IO or SIO is not meaningful.

CCWD=nnnnn (CCW plus data) records or prints nnnn bytes of the transferred data, the CCB and the TOD clock in addition to the information processed with the CCW specification. The number nnnn may be any (decimal) number between 32 and 65535.

The most meaningful trace type to be combined with this output option is the IO trace.

Please find an example of the output produced with this option in Figure 6-18 on page 6-36.

You may define either CCW or CCWD=nnnnn.

IO Trace Output Example with OUTPut=(CCWD=256)

The example shown in Figure 6-18 on page 6-36 contains two IO trace event records (OUTP=(CCWD=256) defined).

The first record indicates a channel end, the second a device end interrupt. The amount of data defined by the residual count in the applicable CCW is shown.

```
                                                   ┌── channel end
                                                   │
IO  05A0  AR 0083B6=>009256  CCW =000069F8 08000000          PSW=070C0000
    TOD = 85.135 18.32.735
    CCB =0069F8  00000500 00000004 000069E8 200069F8
                                        ┌> 4 bytes data ─┐
                                        │                │
    CCW =0069E8  01006C49 80000004 DATA=4211D440 <───────┘
    CCW =0069F0  01007161 20000051
                                        │
    ┌─────── 51 bytes data ─────────────┘
    V
    ───CCW DATA───
    007161 1DE0F1F7 40C1D940 F0F1F540 F1C9F4F0 C94040D9 C5C1C4E8 40404040 40404040
    007181 40404040 40404040 40404040 40404040 40404040 40404040 40404040 40404040
    0071A1 TO 0071B1  SUPPRESSED  LINE(S) SAME AS ABOVE ...

                                                   ┌── device end
                                                   │
IO  09A0  AR 0083B6=>009256  CCW =00000000 04000000          PSW=070C0000
    TOD = 85.135 18.33.271
    CCB =0069F8  00000500 00000004 000069E8 200069F8
```

Figure 6-18. Output of OUTPut=(CCWD=256)

Record the Partition Communication Region

```
OUTPUT=
       COMREG
```

COMREG records or prints the contents of the partition communication region.

Record the Control Registers

```
OUTPUT=
        CREG
```

CREG records or prints the contents of all control registers together with the trace event record(s).

Dump Virtual or Real Storage

```
OUTPUT=
        DUMP partition-id [OFFSET=]              |
             PHASE [OFFSET=]                     |
             ADDRESS=                            |
             REG=reg:bytes                       |
             PTR=reg:offset DMP=offset:length
```

DUMP records or prints the contents of virtual or real storage.

You may request up to ten different dumps.

You have to specify one or more of the dump area specifications as shown below.

partition-id (BG, F1, F2, and so on or, in 370 mode, also BGR, F1R, F2R, and so on.) The OFFSET operand may be defined in addition to the partition-id to limit the dump area in the partition. See "OFFSET Definition" on page 6-29.

PHASE limits the dump to the phase that is specified in the applicable area definition. The OFFSET operand may be defined with the PHASE operand to limit the dump area in the phase.

See "PHASE Definition" on page 6-29.

ADDRESS defines a dump range by a pair of addresses. For example, if you want to dump the contents of four bytes starting at storage location 0080 (hexadecimal). The definition in direct input mode would look like this:

```
ADD=0080:0083
```

See "ADDRESS Definition" on page 6-28.

REG=reg:bytes if the starting address of the dump is specified by a register, and the number of bytes to be dumped is specified by a hex value.

PTR=reg:offset DMP=offset:length if the dump area is located via a register plus an offset which addresses a pointer. The dump area itself is determined by the definition

```
DMP=offset:length
```

which defines the dump start address relative to that pointer and the dump length.

The offsets and the dump length are specified in hexadecimal notation. The following example and Figure 6-19 on page 6-39 explain such a dump area definition.

For example:

The contents of register E is used to locate a control block. The fullword at relative offset X'10' in this control block is used as a pointer to another control block or data area.

Starting at offset X'200' an area of X'100' bytes is dumped.

Direct Input Mode Format

OUTPUT=(DUMP PTR=E:10 DMP=200:100)

Figure 6-19 shows, how the area to be dumped is localized, when you use the example shown above.

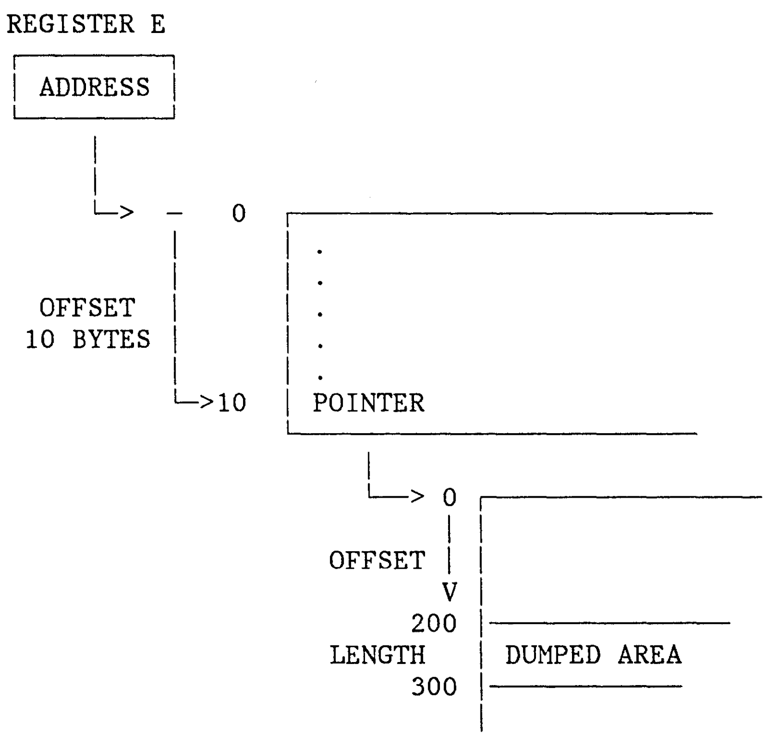


Figure 6-19. Overview: Defining the Area to be Dumped

Record Floating Point Registers

```
OUTPUT=
      FREG
```

FREG records or prints the contents of all floating point registers. Figure 6-20 shows an example of a printout of the contents of these registers.

INSTRUCTION Trace Output Example with OUTPUT=FREG

```
                                ┌── instruction that triggered the output

   INST  58 1 BG 1010B2        5870817C                         PSW=470D1000
         FP 0–7 00000000 00000000 00000000 00000000 00000000 00000000 64E2D100
```

Figure 6-20. Printout of Floating-Point Registers

Record General Purpose Registers

```
OUTPUT=
      GREG
```

GREG records or prints the contents of all general purpose registers.

Figure 6-21 on page 6-41 shows an example of a printout of the contents of the general purpose registers.

Trace Output Example with OUTPUT=GREG

```
                                     ┌──instruction that triggered output
                                     │
 INST 05   1 BG 1010B6          0577                         PSW=470D1000
  GR 0-7  80400C00 02021B40 80000000 FFFFFFFF 00000000 00000000 00000000 00000000
  GR 8-F  00000000 0000000F 00100000 001A76FF 00000000 00000000 8F000000 00000200
```

Figure 6-21. Printout of General Registers

Record PUB, LUB, ERBLOC, CHANQ

```
OUTPUT=
       IOTAB
```

IOTAB records the contents of the following I/O tables: PUB, LUB, ERBLOC, CHANQ.

Record the Logical Transient Area

```
OUTPUT=
       LTA
```

LTA records the contents of the LTA on occurrence of the associated event.

Dump Processor Storage from X'00'to X'BC'

OUTPUT=
LOWCORE

LOWCORE (Lowcore contents) records or prints the contents of the first 188 bytes of processor storage (X'00' to X'BC').

See Figure 6-22 for a sample output.

Trace Output Example with OUTPUT=LOWCORE

```
        address of COMREG for active partition-----------┐
                                   ┌---- PGMCHK old PSW        |
                                   |                           |
 PGMCHK 02 1 BG 1012BA ==>005A30|  020B0000                    |  PSW=471D0000
        ---LOCORE----              |                           |
        000000 400FA248 00000000 |    1000 000FB1FA 00000000 00000330 00000000 00000000 .......
        000020 47100000 00100228 040C2000 00003886 00000000 00000000 020C1000 000FB1FA .....
        000040 00000000 00000000 000488B0 00000000 FEE67800 00000000 040C0000 000008EE .....
```

Figure 6-22. Printout of Low Address Storage. (bytes X'0' to X'60')

Record the Physical Transient Area

OUTPUT=
PTA

PTA records or prints the contents of the physical transient area on occurrence of the associated event.

Record the Partition Information Block

```
OUTPUT=
       PIB
```

PIB records or prints the contents of partition information blocks that are associated with the main task of the partition for which the specified trace operation is active.

Record Partition Related Control Blocks

```
OUTPUT=
       PTAB
```

PTAB (partition-related control blocks) records or prints the contents of the tasking control blocks (PIB, PCB) for all partitions.

Record the Supervisor Area

```
OUTPUT=
       SUPVR
```

SUPVR records or prints the contents of the storage area used by the supervisor.

Record System Communication Region

```
OUTPUT=
       SYSCOM
```

SYSCOM (system communication region) records or prints the contents of the system communication region.

Record Time-of-Day Clock

```
OUTPUT=
       TOD
```

TOD records or prints the setting of the time-of-day clock each time the associated event occurs.

Figure 6-23 shows an example of this option.

<u>Trace Output Example with OUTPUT=TOD</u>

```
                                    ┌── instruction that triggered output

   PGMCHK 02 1 BG 1012BA ==>005A30  020B0000                  PSW=471D0000
          TOD = 84.101 13.54.02.763
```

Figure 6-23. Program-Check Event with Time of Day

Record Task Related Control Blocks

```
OUTPUT=
       TTAB
```

TTAB (task-related control blocks) records or prints the contents of the tasking control blocks for all tasks (TIB, TCB, PIB, PCB).

DEFINE TRACE OPTIONS

```
OPTION=
definition|(definition1 definition2)
```

You may define one or more of the following options:

HALT,NOJCL,TERMINATE,OCCURRENCE=occ1:occ2

HALT stops the processing of the system when the defined trace event occurs.

The SDAID puts the system into a wait state with address X'00EEEE' in the address portion of the wait PSW. This wait on event enables you to perform some debugging work, for example to display registers or selected storage areas.

How to Get out of this WAIT

- Give an external interrupt.

  The trace remains initialized and the system stops at the next trace event occurrence.

- Alter storage location 0 to a value of X'FF', then give an external interrupt.

  The trace remains active but the HALT option is canceled.

NOJCL suppresses tracing of Job Control Phases. If the keyword NOJCL is omitted, the user program and the job control statements are also traced.

TERMinate allows you to terminate SDAID output at the occurrence of the specified event. You may start the trace output again if you issue the STOPSD and the STRTSD attention routine commands. Note, that you should issue the attention routine command ENDSD if you want to end the trace operation and release all system resources which SDAID has used.

OCCURRENCE=occ1:occ2 specifies the number of associated events to be traced. For example, the specification 1:20 defines that tracing starts with the first occurrence of the specified trace event and ends with occurrence 20.

DEFINE THE TRACED I/O DEVICES

UNIT=cuu|CU=cu|CHANNEL=c

UNIT=cuu specifies one or up to 8 device addresses.

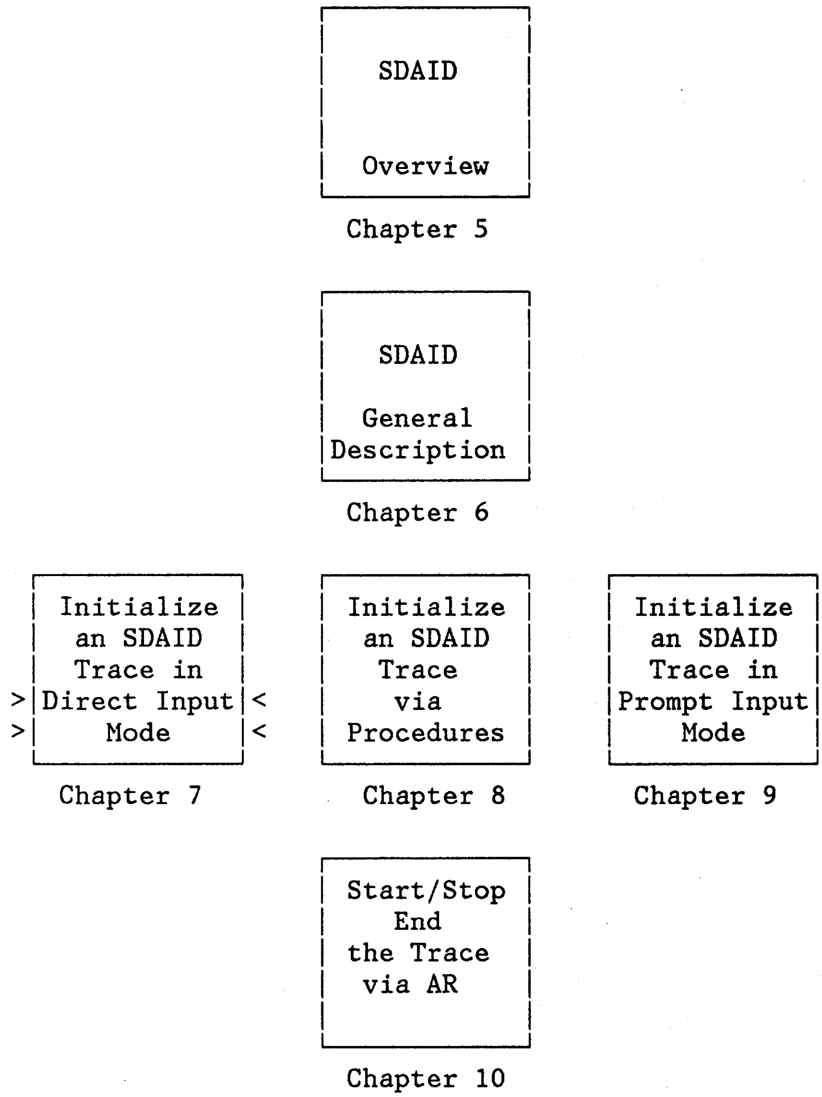
CU=cu specifies one or up to 16 control unit addresses.

CHANNEL=c specifies one channel address or a list of up to 16 addresses.

Notes

- You may define these operands with the TRACE statements for the IO trace, the SIO trace, and the VTAMIO trace.
- The parameters UNIT, CHANNEL, and CU are optional. If you do not specify the UNIT statement, all devices are traced.
- The parameters UNIT, CHANNEL, and CU are mutually exclusive (in the same TRACE command).

CHAPTER 7. INITIALIZE AN SDAID TRACE IN DIRECT INPUT MODE



This chapter describes how you can initialize a SDAID trace via statements read in from a SYSIN device or the attention routine (AR).
Direct input means that you enter the trace specifications directly via control statements. Beside the trace initialization in prompt mode or via procedures the direct input mode can be used to set up SDAID traces. You can use the direct input mode of SDAID to enter complete SDAID commands avoiding the time consuming prompt input mode.

Contents

HOW TO INITIALIZE AN SDAID TRACE IN DIRECT INPUT MODE

An SDAID trace initialization in direct input mode is based on the following statements:

SDAID Statement	Page
[//] EXEC SDAID or SDAID	7-8
OUTDEV specification	7-9
TRACE type and additions	7-13
/*, READY or EOB	7-8

Figure 7-1. Input Statement Summary

The Various SDAID Statements

You may initialize traces via the attention routine either in direct input mode or in prompt mode. The SDAID program may switch from one input mode to the other according to your command input. However, the statements described in this section belong to direct input mode only.

COMMANDS ENTERED VIA THE ATTENTION ROUTINE:

The **SDAID program is started** in the attention routine via the command

```
SDAID
```

You **determine the direct input mode** by defining at least one keyword operand together with the commands TRACE and OUTDEV.

For example the OUTDEV command together with the keyword operand P=E.

```
OUTDEV P=E
```

The SDAID program **switches to prompt input mode** when you omit all possible keyword operands or you use a prompt input mode statement. For example if you request the prompt mode help function with the question sign (?).

```
TRACE INST=* AR=BG OUTP=?
```

COMMANDS ENTERED IN A PARTITION:

You may **invoke the SDAID program** in a partition via the statement

```
// EXEC SDAID
```

This statement and the SDAID trace initialization commands may be entered from a console, or from a SYSIN device.

You **specify the output device** of the trace with the OUTDEV statement. Only one output device specification is possible at one time in the system. Any subsequent OUTDEV statement overwrites the existing one.

The TRACE statement contains the **definition of the trace type** and additional trace keyword operands. You can enter up to ten TRACE statements.

You **end the trace initialization** with the READY statement. If you entered the statements via SYSIN, the end of data (/*) or EOB in case of console input are treated as READY statements and end the initialization process. When the READY statement has been read in, no further OUTDEV or TRACE statement can be entered.

You can **flush the SDAID set-up** during initialization with the ENDSD or CANCEL statement. Two initialization examples are shown in Figure 7-2 on page 7-5. One entered directly via the attention routine, the other read in via a SYSIN device.

In both examples the same SDAID trace is initialized.

```
Direct Input Set-up in the Attention Routine

-> sdaid
   AR 015 4C05I PROCESSING OF 'SDAID' COMMAND SUCCESSFUL.
   AR 015 1I40I READY
-> outdev t=280
   AR 015 4C05I PROCESSING OF 'OUTDEV' COMMAND SUCCESSFUL.
   AR 015 1I40I READY
-> trace branch ar=bg off=0:200 outp=(dump add=40000:40100)
   AR 015 4C05I PROCESSING OF 'TRACE' COMMAND SUCCESSFUL.
   AR 015 1I40I READY
-> trace inst=(d207 95 9103) ar=bg outp=greg
   AR 015 4C05I PROCESSING OF 'TRACE' COMMAND SUCCESSFUL.
   AR 015 1I40I READY
-> ready
   AR 015 4C05I PROCESSING OF 'READY' COMMAND SUCCESSFUL.
   AR 015 1I40I READY


Direct Input Set-up under Control of a Partition

 // EXEC SDAID
 OUTDEV T=280
 TRACE BRANCH AR=BG OFF=0:200 -
              OUTP=(GREG DUMP ADD=40000:40100)
 TRACE INST=(D207 95 9103) AR=BG OUTP=GREG
 /*
```

Figure 7-2. Trace Initialization Examples (Direct Input Mode)

Statement Format Considerations

- The various operands may be separated by a comma or by at least one blank.

- Enter all operands in the order explained.

- Most keywords may be abbreviated. The possible abbreviation is shown through lower case letters. For example the branch trace statement description looks like this:

  TRACE BRanch .....

  Thus, you may submit the branch trace type specification as any of the following:

  BR BRA BRAN BRANC BRANCH

- Mandatory operands are highlighted.

- Optional operands are enclosed in square brackets '[]'

- Operands separated by a '|' denote choices one of which must be selected.

  Figure 7-3 shows for example that:

  - You may enter either only one SVC code,
    for example: SVC=0,
    or several codes enclosed in brackets,
    for example: SVC=(0 6 32),
    or you can define, that all SVC instructions should be traced:
    for example: SVC=*.

  - The AREA operand must be specified, it is mandatory in the SVC trace statement.
  - The other additional operands OUTPUT and OPTION are optional.

```
TRACE SVC=svc|(svc1 svc2 ..)|* ARea=
                                [OUTPut=]
                                [OPTion=]
```

Figure 7-3. Statement Format Example

- Command continuation is allowed. It is specified by a trailing minus sign ('-').

- Comments may be specified via SYSIN together with SDAID statements or as separate comment lines.

  A '/*' sign specifies the begin of a comment. All text from the '/*' sign up to the end of the line is treated as a comment. '/*' must not start in column 1.

  The example given in Figure 7-4 on page 7-7 shows a TRACE statement in which continuations (-) and comments (/*) are used.

```
/* BRANCH TRACE SPECIFICATIONS
TRACE BR AR=BG -                        /* PARTITION=BG
        OUTP=(GREG -                    /* INCLUDE REGS
             DUMP -                     /* INCLUDE DUMP
             ADD=400000:410000)
```

Figure 7-4. SYSIN Statement Format Example

START SDAID TRACE INITIALIZATION

[//] **EXEC SDAID | SDAID**

You start the initialization process with the statement 'SDAID'.

If you want to set-up the trace from the AR, type only 'sdaid'.

If you want to initialize the trace in a partition via SYSLOG, enter 'exec sdaid' in that particular partition. Submit 'exec sdaid' or '// exec sdaid' if you use SYSRDR as input device.

END THE SDAID TRACE INITIALIZATION

READY	**In the Attention Routine**
/*	**In a Partition via SYSIPT**
EOB	**In a Partition via Console**

You end the initialization process with the statements 'READY' or '/*'. Now you can start the initialized trace with the AR command STRTSD.

You find more information about starting and stopping a trace in Chapter 10, "Start/Stop and End the Trace" on page 10-1.

DEFINE THE OUTPUT DEVICE IN DIRECT INPUT MODE

OUTDEV [BUffer=nn][Tape=cuu|Printer=cuu]

You define the destination of your trace output with the OUTDEV statement.

Enter the OUTDEV statement with an operand to get into the direct input mode. Otherwise, if you entered the SDAID statement from the attention routine, the SDAID program would prompt you for the necessary information (prompt mode).

Note: An OUTDEV statement in a partition must contain at least one operand (prompt mode is not possible in a partition).

Output Device Definition

You can define the following output destinations with the OUTDEV statement:

- Printer
- Tape
- Wrap-around buffer
- Wrap-around buffer and printer
- Wrap-around buffer and tape

Find the appropriate OUTDEV and TRACE statement for your trace initialization in Figure 7-5 on page 7-10.

Device	Buffer	Output when:	SDAID Statements	Note
Printer	no	immediately	OUTDEV P=cuu	1
Tape	yes	3K buffer full	OUTDEV T=cuu	2
-	yes	-	OUTDEV BU=nn	3
Printer	yes	certain event occurs	OUTDEV BU=nn P=cuu TRACE type OUTP=BU	4
Tape	yes	certain event occurs	OUTDEV BU=nn T=cuu TRACE type OUTP=BU	5

Figure 7-5. OUTDEV Summary

nn..... stands for the wrap-around buffer size in units of K bytes.
If you do not define an output device or you use a printer device, nn may have a value of 3 up to the maximum of 99.
If you use a tape device, nn may have a value of 3 up to the maximum of 32.

cuu ... stands for the unit address.
You may abbreviate it in the following way:
00E, 0E, E

type .. represents the type of trace event which forces the output operation.

Notes

[1] No buffer is allocated. The event records are printed on the printer with the device address cuu.

[2] The event records are written into a 3K bytes buffer. This buffer is written to a tape mounted on the device cuu when it is full or when an ENDSD or STOPSD command is issued.

[3] A trace defined with this OUTDEV statement writes the event records into a wrap-around buffer. You can retrieve the trace records only with the attention routine command:

```
DUMP BUFFER,cuu
```

DOSVSDMP can be used to print the tape. Please find further information under "Print the SDAID Produced Tape" on page 4-6.

[4] A trace defined with this OUTDEV and TRACE statement prints the contents of the buffer when the event (type), defined with the TRACE statement, occurs.

[5] The defined wrap-around buffer is written to tape (cuu) when the trace event (type) occurs.

Note: The buffer is a part of the SDAID area. A definition of a buffer may cause a bottleneck in storage. Please find more detailed information under "Space Requirements for SDAID Execution" on page 5-10.

OUTDEV Statement Example:

Assume you want to make SDAID tracing into a 6K bytes buffer. (The definition of the recorded events is shown as TRACE ...).

- The buffer is written to a tape when an EOJ condition is encountered (TRACE CA OUTP=BU ...).
- The tape is mounted on unit address 280 (OUTDEV .... T=280).
- The buffer size should be 6K bytes to get a reasonable amount of event records (OUTDEV BU=6 ...).
- The SDAID statements are entered in direct input mode via SYSIN.

```
// EXEC SDAID
OUTDEV BU=6 T=280
TRACE .......
TRACE CA OUTP=BU,OPT=TERM
/*
```

Figure 7-6. OUTDEV Statement Example

Initialization Example:

- Assume you want to make SDAID tracing instructions into a 6K bytes buffer as long as no program check comes up in the BG partition.
- The buffer should be written to a tape when the program check interrupt is received. The tape is mounted on unit address 280.
- The buffer size should be 6K bytes to get a reasonable amount of recorded event records.
- The SDAID statements are entered in direct input mode via SYSIN.

```
// EXEC SDAID
OUTDEV BU=6 T=280
TRACE INST=* AR=BG
TRACE PGMC=* OUTP=BU,OPT=TERM
/*
```

Figure 7-7. Initialize a Trace Example

THE TRACE STATEMENT

The TRACE statement is used to define the trace types you need to get the most reasonable information about errors in your computing environment.
This section describes the format of all SDAID trace type initializations in direct input mode and shows trace statement and trace initialization examples.

Find the appropriate trace type in Figure 7-8. The possible abbreviations are shown through lower case letters.

Summary of Trace Types

Trace Type	Provides a Trace of:	Page
BRanch	Successfully executed branch instructions	7-15
BUffer	Output the trace buffer when it is full	7-16
CAncel	Program (main task) cancel or EOJ	7-17
EXTernal	External interrupts	7-18
INSTruction	Selected or all instruction(s) execution	7-19
IO	I/O interrupts	7-21
MONitorcall	MC instructions	7-22
PAGing	Page faults	7-23
PGMCheck	Program checks	7-24
PGMLoad	Phase load requests, or actual load	7-26
REGister	Contents or alterations of 1 to 8 registers	7-28
SIO	SIO instructions	7-31
STorage	Storage alterations	7-32
SVC	Executed supervisor calls	7-34
VTAMBU	Usage of ACF/VTAM buffers	7-36

Figure 7-8 (Part 1 of 2). Trace Type Summary

Trace Type	Provides a Trace of:	Page
VTAMIO	VTAM I/O operations	7-37

Figure 7-8 (Part 2 of 2). Trace Type Summary

Besides the type of the trace and some definitions which belong to the trace type, other keyword operands like AREA, OUTPUT or OPTION may be defined to limit the trace or to produce additional trace output.

These other operands are grouped together and their format is shown under "Additional Definitions" on page 7-38.

BRanch Trace

TRACE BRanch ARea= [-]
[OUTPut=] [-]
[OPTion=]

See for ARea= page 7-39, OUTPut= page 7-44, OPTion= page 7-48

Find the description of the trace and an example of the output under "BRANCH Trace" on page 6-8.

Initialization Example

The following items are met by the trace set up shown below:

- Use the SYSRDR device to set up the trace.
- Trace all branches taken.
- The branch instructions under the control of the BG partition are to be traced.
- The whole system is to be observed to record the instructions executed for the BG partition.
- Output the trace data to the tape at device address 280.
- Do not trace the job control branch instructions.

```
// EXEC SDAID
OUTDEV T=280
TRACE BR AR=BG ADDR=0:* -
         OPT=NOJCL
/*
```

Figure 7-9. Example: Branch Trace Initialization

BUffer Trace

TRACE BUffer [OPTion=]

For OPTion see page 7-48

Find the description of the trace under "BUFFER Trace" on page 6-8.

The BUffer trace output is the collection of all trace records contained in the buffer when a buffer overflow occurs.

Initialization Example

The following items are met by the trace set-up shown below:

- Use the SYSRDR device to set up the trace.
- Trace all successfully executed branch instructions.
- The branch instructions of the BG tasks are to be traced.
- Observe the whole storage.
- Collect the trace data in a 6K byte buffer.
- Output the buffer whenever it is full.
- Write the output to the tape on device address 280.
- Do not trace the job control branch instructions.

```
// EXEC SDAID
OUTDEV BU=6 T=280
TRACE BR AR=BG ADDR=0:* -
        OPT=NOJCL
TRACE BU
/*
```

Figure 7-10. Example: Buffer Trace Initialization

CAncel Trace

TRACE CAncel ARea= [-]
[OUTPut=] [-]
[OPTion=]

See for ARea= page 7-39, OUTPut= page 7-44, OPTion= page 7-48

Find the description of the trace and an example of the output under "CANCEL Trace" on page 6-9.

Statement Example

The trace statement shown initializes a CAncel trace which writes an event record and the buffer whenever a cancel or EOJ condition occurs in the BG partition.

```
TRACE CA AR=BG -
         OUTP=BU
```

Figure 7-11. Example: CAncel Trace Statement

Initialization Example

The following items are met by the trace set-up shown:

- Use the SYSRDR device to set up the trace.
- Trace cancel and EOJ conditions.
- Print the area between BG relative address X'78' and X'1000' together with the event record.
- Output the trace record on the printer at device address 00E.

```
// EXEC SDAID
OUTDEV P=E
TRACE CA AR=BG -
         OUTP=(DUMP BG OFF=78:1000)
/*
```

Figure 7-12. Example: CAncel Trace Initialization

EXTernal Trace

```
TRACE EXTernal [KEY][TIMER][SIGNAL]  [-]
               [OUTPut=]   [-]
               [OPTion=]
```

For OUTPut= see page 7-44, OPTion= page 7-48

If you specify more than one external interrupt type, separate them by one or more blanks or by a comma (with or without blanks) and enclose them in brackets.

Find the description of the trace and an example of the output under "EXTERNAL Trace" on page 6-9.

Statement Example

The example shows an external interrupt trace. KEY is defined as external interrupt type. This interrupt is used to have the wrap-around buffer and the TOD clock recorded or printed together with the external trace event record.

```
TRACE EXT KEY -
      OUTP=(TOD,BU)
```

Figure 7-13. Example: EXTernal Interrupt Trace Statement

Initialization Example

The following items are met by the trace set-up shown below:

- Use the SYSRDR device to set up the trace.
- Trace all signal and timer interrupts.
- Output the trace data to the tape on device address 280.
- Add the TOD clock to the trace data output.

```
// EXEC SDAID
OUTDEV T=280
TRACE EXT (SIGNAL TIMER) -
      OUTP=TOD
/*
```

Figure 7-14. Example: EXTernal Trace Initialization

TRACE INSTruction=opcode|(opcode1 opcode2 ..)|*|BRanch ARea= [-]
[OUTPut=] [-]
[OPTion=]

See for ARea= page 7-39, OUTPut= page 7-44, OPTion= page 7-48

opcode (one to eight) entered as either one-byte or two-byte hexadecimal instruction codes.

If you specify more than one operation code, separate them by one or more blanks or by a comma (with or without blanks) and enclose them in brackets.

*** (asterisk)** requests a trace of all executed instructions.

BRanch requests that all types of branch instructions be recorded.

Find the description of the trace and an example of the output under "INSTRUCTION Trace" on page 6-11.

Statement Examples

- Trace BG task CLC instructions
- Trace all BG task instructions
- Trace BG task branch type instructions
- Trace selected BG task instructions.

```
TRACE INST=D7 AR=BG
TRACE INSTR=* AR=BG
TRACE INST=BR AR=BG
TRACE INST=(92 D204) AR=BG
```

Figure 7-15. Example: INSTruction Trace Statements

Initialization Example

The following items are met by the trace set-up shown below:

- Use the SYSRDR device to set up the trace.
- Collect all trace data into a wrap-around buffer.
- The MVC instructions of BG tasks are to be traced.
- Write the buffer to the tape on device address 280 when a program check occurs in the BG partition.
- Do not trace the job control MVC instructions.

```
// EXEC SDAID
OUTDEV BU=4 T=280
TRACE INST=D2 AR=BG -
         OPT=NOJCL
TRACE PGMC=* AR=BG -
         OUTP=BU
/*
```

Figure 7-16. Example: INSTruction Trace Initialization

I/O Interrupt Trace

```
TRACE IO [ARea=BG|Fn|SUPvr|ALL] [-]
         [UNit=|CHannel=|CU=]  [-]
         [OUTPut=]  [-]
         [OPTion=]
```

For UNit= see page 7-43 OUTPut= page 7-44, OPTion= page 7-48

Find the description of the trace and an example of the output under "IO Trace (I/O Interrupt)" on page 6-12.

Statement Examples

The shown TRACE statement examples define additional trace output (GREG CCWD=512 and CCW) to the normal IO trace event records.

```
TRACE IO  UNIT=(130 133) -
          OUTP=(GREG CCWD=512)

TRACE IO  CU=28 OUTP=CCW
```

Figure 7-17. Example: IO Trace Statement

Initialization Example

The following items are met by the trace set-up shown below:

- Use the SYSRDR device to set up the trace.
- Trace all I/O interrupts from device with the address 320.
- Output the trace data to the tape on device address 280.
- Add the CCW plus up to 512 bytes of transferred data to the trace event record.

```
// EXEC SDAID
OUTDEV T=280
TRACE IO -
      UN=320 -
      OUTP=CCWD=512
/*
```

Figure 7-18. Example: IO Trace Initialization

TRACE MONitorcall=mc|(mc1 mc2 ..)|* ARea= [-]
[OUTPut=] [-]
[OPTion=]

See for ARea= page 7-39, OUTPut= page 7-44, OPTion= page 7-48

---

mc (monitor classes) defines the monitor class of the MC instructions to be traced. Only the monitor call instructions with the defined class are traced. Up to eight classes may be defined.

The monitor classes must be specified as one-digit hexadecimal values. If you specify two or more classes, they must be enclosed in brackets and separated by one or more blanks, or by a comma with or without one or more blanks.

You may specify any valid monitor class; however, SDAID ignores a specification of class 2.

*** (asterisk)** provides an event record for any execution of an MC instruction (except an MC instruction with class 2 specified) within the range of the trace operation.

Find the description of the trace and an example of the output under "MONITORCALL Trace" on page 6-13.

Statement Examples

• Trace all class 3 monitor calls.	TRACE MON=3
• Trace all monitor calls.	TRACE MON=*
• Trace all class 3 and 4 MCs.	TRACE MON=(3 4)

Figure 7-19. Example: MC Trace Statements

PAGing Trace

```
TRACE PAGing ARea=       [-]
             [OUTPut=]   [-]
             [OPTion=]
```

See for ARea= page 7-39, OUTPut= page 7-44, OPTion= page 7-48

Find the description of the trace and an example of the output under "PAGING Trace" on page 6-14.

Statement Example

The example defines a trace to record information about all paging events in the F2 partition.

```
TRACE PAG AR=F2
```

Figure 7-20. Example: PAGing Trace Statement

PGMCheck Trace

TRACE PGMCheck=pgmc|(pgmc1 pgmc2 ..)|* ARea= [-]
[OUTPut=] [-]
[OPTion=]

See for ARea= page 7-39, OUTPut= page 7-44, OPTion= page 7-48

---

pgmc At least one program interruption code (up to 16) must be specified in hexadecimal notation; leading zeros may be omitted.

If you specify more than one program interruption code, they must be enclosed in brackets and separated by one or more blanks, or by a comma with one or more blanks.

***** (asterisk) requests a trace of all valid program interrupt codes with a value less than X'40' and except the page fault interrupt.

Find the description of the trace and an example of the output under "PGMCheck Trace (Program Check)" on page 6-15.

Statement Examples:

- Trace BG task program check addressing exceptions
- Trace all program checks of BG tasks
- Trace the program checks of BG tasks
  with interruption codes
  1 ... operation exception
  A ... decimal overflow exc.
  11... page translation exc.

```
TRACE PGMC=5 AR=BG

TRACE PGMC=* AR=BG

TRACE PGMC=(1 A 11) AR=BG
```

Figure 7-21. Example: PGMCheck Trace Statement

Initialization Example

The following items are met by the trace set up shown below:

- Use the SYSRDR device to set up the trace.
- The MVC instructions (D2) of the F2 partition are to be traced.
- Collect the trace event records in a 4K bytes wrap-around buffer.
- Write the buffer to the tape on device address 280 when a program check interrupt with interruption code 'operation exception (0001)' occurs in the F2 partition.
- Do not trace the instructions executed during job control processing.

```
// EXEC SDAID
OUTDEV BU=4 T=280
TRACE INST=D2 AR=F2 -
         OPT=NOJCL
TRACE PGMC=1 AR=F2 -
         OUTP=BU
/*
```

Figure 7-22. Example: Program Check Trace Initialization

Program Load Trace (Fetch/Load Trace)

TRACE PGMLoad [ALL|REQ|HDL] [PH=phase] **ARea=** [-]
[OUTPut=] [-]
[OPTion=]

See for ARea= page 7-39, OUTPut= page 7-44, OPTion= page 7-48

REQ defines, that an event record for each request for fetching/loading a phase is to be written.

HDL defines that an event record is to be written each time a phase fetch/load request is handled; that is, when a requested phase is actually loaded into storage for execution.

ALL combines REQ and HDL.

PH=phase defines the phase whose program load events should be traced.

Find the description of the trace and an example of the output under "PGMLOAD (Fetch/Load) Trace" on page 6-16.

Statement Examples

- Trace all program load events for BG tasks.
- Trace all BG task program load events for the phase PROGR1.

```
TRACE PGML AR=BG
TRACE PGML PH=PROGR1 AR=F2
```

Figure 7-23. Example: Program Load Trace Statements

Initialization Example

The following items are met by the trace set up shown below:

- Use the SYSRDR device to set up the trace.
- Trace all fetch/load executions in the BG partition.
- Record the trace data in a 6K bytes wrap around buffer.
- Output the trace data to the tape on device address 280 when a program check operation exception (interrupt code 1) occurs in the BG partition.
- Add a dump of the BG area between relative address 1000 to 2000 to both event records.
- Do not trace the job control activities.

```
// EXEC SDAID
OUTDEV BU=6 T=280
TRACE PGML HDL AR=BG -
      OUTP=(DUMP BG OFF=1000:2000)
      OPT=NOJCL
TRACE PGMC=1 AR=BG -
      OUTP=(DUMP BG OFF=1000:2000 -
      BUFFER) -
      OPT=NOJCL
/*
```

Figure 7-24. Example: PGMLoad (fetch/load) Trace Initialization

Register Alteration Trace

```
TRACE REGister=reg|(reg1 reg2 .. reg8) [PATTern=pp|(pp nn)]  [-]
                  ARea=             [-]
                  [OUTPut=]         [-]
                  [OPTion=]
```

See for ARea= page 7-39, OUTPut= page 7-44, OPTion= page 7-48

reg defines one to eight general purpose register numbers. Define the register number in hexadecimal notation.

If you specify more then one register number, these numbers must be separated by one or more blanks, or by a comma with or without one or more blanks and must be enclosed in brackets.

```
Sample register number definitions: 9
                                    (8 9,a, f)
```

reg PATTern=pp defines a register and certain register contents. **pp** defines a hexadecimal pattern of up to four bytes.

(reg1 reg2 ...) PATTern=pp defines the same trace as shown above but with the tracing of up to eight defined registers.

reg PATTern=(pp nn) defines a register and a certain register contents range (pp to nn) all in hexadecimal notation. **(pp nn)** defines a pattern range between X'pp' up to X'nn'.

The two values must be separated by one or more blanks, or by a comma with or without one or more blanks and must be enclosed in brackets.

(reg1 reg2 ...) PATTern=(pp nn) defines the same trace as shown above but with the tracing of up to eight defined registers.

Sample pattern specifications:

```
3e8        (the same as 000003e8)
(0 7d0)    (the same as 00000000, 000007d0)
```

Find the description of the trace and an example of the output under "REGISTER Alteration Trace" on page 6-17.

Statement Examples

The examples shown below define the following functions for the BG partition (AR=BG).

[1] Trace instructions which alter register 12 (X'C').

[2] Trace all instructions which alter registers 1 or 12 (X'C').

[3] Trace all instructions which alter register 15 (X'F') to a pattern of 0000000C.

[4] Trace all instructions which alter register 2 to a value which is equal to or which is within the limits 0000000C-0000000F.

[5] Trace all instructions which alter register 2 or 15 to a value which is equal to or which is within the limits of 0000000C-0000000F.

```
(1) TRACE REG=C AR=BG

(2) TRACE REG=(1 C) AR=BG

(3) TRACE REG=F PATT=0000000C -
                AR=BG

(4) TRACE REG=2 -
      PATT=(0000000C 0000000F) -
      AR=BG

(5) TRACE REG=(2 F) -
      PATT=(0000000C 0000000F) -
      AR=BG
```

Figure 7-25. Example: REGister Trace Statements

Initialization Example

The following items are met by the trace set up shown below:

- Use the SYSRDR device to set up the trace.
- Trace BG instructions which alter register 15 (X'F').
- Record the trace data in a 3K bytes wrap around buffer.
- Output the trace data together with a program check event record to the tape on device address 280 when a program check operation exception (interrupt code 1) occurs in the BG partition.
- Do not trace the register alterations of the job control program.

```
// EXEC SDAID
OUTDEV BU=3 T=280
TRACE REG=F AR=BG -
            OPT=NOJCL
TRACE PGMC=1 AR=BG -
            OUTP=BUFFER
/*
```

Figure 7-26. Example: REGister Trace Initialization

SIO Instruction Trace

```
TRACE SIO [ARea=BG|Fn|SUPvr|ALL]  [-]
          [UNit=|CHannel=|CU=]  [-]
          [OUTPut=]   [-]
          [OPTion=]
```

For UNit= see page 7-39, OUTPut= page 7-44, OPTion= page 7-48

ARea=BG|Fn|SUPvr|ALL defines those tasks via an area, whose SIO instructions are to be traced. Only the specifications shown above are possible. If you omit the ARea operand, all SIO instructions executed in the VSE/Advanced Functions system will be traced.

UNit,CHannel,CU limits the trace to SIO instructions for a certain unit, channel, or control unit. If you omit these operands no device address limitation is used.

Find the description of the trace and an example of the output under "SIO Instruction Trace" on page 6-18.

<u>Statement Examples</u>

The TRACE statement examples shown below define the following functions:

[1] Trace each SIO instruction of F4 tasks for the device with the device address 009. Add a dump of the CCB to each SIO trace event record.

[2] Trace all SIO instructions which concern the channels 2 and 3. Add the time of day entry (TOD) to each SIO trace event record. Please find an example of such a TOD entry under "Trace Output Example with OUTPUT=TOD" on page 6-44.

```
(1)  TRACE SIO AR=F4 UNIT=009 -
               OUTPUT=CCB

(2)  TRACE SIO CHANNEL=(2 3) -
               OUTP=TOD
```

Figure 7-27. Example: Start I/O Trace Initialization

Storage Alteration Trace

```
TRACE STorage [PATTern=xxxxxxxx] ARea= ADDress=  [-]
                                   [OUTPut=]  [-]
                                   [OPTion=]
```

See for ARea= page 7-39, OUTPut= page 7-44, OPTion= page 7-48   ADDress= see 7-41

xxxxxxxx defines a hexadecimal storage pattern up to four bytes long.

If you specify an odd number of digits, a zero is inserted to the left of the first specified hexadecimal digit.

ARea= defines those tasks whose alteration activities you want to trace. Usually this trace is used to catch the modifying tasks. Therefore use ARea=ALL to have all tasks of the system watched.

ADDress= defines the observed storage area.

Please do not use the definitions OFFset or PHase.

Find the description of the trace and an example of the output under "STORAGE Alteration Trace" on page 6-20.

Statement Example

The example records all instructions which alter the contents of two bytes starting with storage location X'65674' to the pattern X'D205'.

```
TRACE ST PATT=D205 -
         AR=ALL -
         ADD=65674:65675
```

Figure 7-28. Example: Storage Alteration Trace Statement

Initialization Example

The following items are met by the trace set up shown below:

- Use the SYSRDR device to set up the trace.
- Trace all BG instructions excluding the job control instructions.
- Record the event records in a 3K bytes wrap around buffer.
- Write the buffer together with a storage alter trace event record to the tape on device address 280 when the storage area with the address 40100 to 40101 is altered to X'FFFF'.
- Observe all tasks of your system, in respect to altering the storage X'40100'-X'40101' to X'FFFF'.

```
// EXEC SDAID
OUTDEV BU=3 T=280
TRACE INST=* AR=BG -
             OUTP=GREG -
             OPT=NOJCL
TRACE ST PATT=FFFF -
             AR=ALL -
             ADD=40100:40101 -
             OUTP=BU
/*
```

Figure 7-29. Example: Storage Alteration Trace Initialization

Supervisor Call Trace

```
TRACE SVC=svc|(svc1 svc2 ..)|* ARea=  [-]
                               [OUTPut=]    [-]
                               [OPTion=]
```

See for ARea= page 7-39, OUTPut= page 7-44, OPTion= page 7-48

svc defines a certain Supervisor Call Code. You may define up to 16 different SVC codes. Specify the SVC code in hexadecimal notation.

If you specify more than one SVC code, the codes must be enclosed in brackets and separated by one or more blanks, or by a comma with or without one or more blanks.

***** (asterisk) defines that all SVC instructions are to be traced.

Find the description of the trace and an example of the output under "SVC Trace (Supervisor Call)" on page 6-21.

Statement Examples

- Trace all BG task SVCs
- Trace BG task set timer SVCs (X'A')
- Trace BG task WAITM and STXIT AB SVCs

```
TRACE SVC=* AR=BG

TRACE SVC=A AR=BG

TRACE SVC=(1D 25) AR=BG
```

Figure 7-30. Example: SVC Trace Statement

Initialization Examples

The following items are met by the trace set up shown below:

- Use the SYSRDR device to set up the trace.
- Trace all supervisor call instructions.
- The SVC instructions from the BG partition are to be traced.
- Output the trace data to the tape on device address 280.
- Do not trace the job control branch instructions.

```
// EXEC SDAID
OUTDEV T=280
TRACE SVC=* AR=BG -
          OPT=NOJCL
/*
```

Figure 7-31. Example: SVC Trace Initialization

VTAMBU Buffer Trace

```
TRACE VTAMBU [OUTPut=]  [-]
             [OPTion=]
```

For OUTPut= page see 7-44, OPTion= page 7-48

Find the description of the trace and an example of the output under "VTAMBU Trace (VTAM Buffer)" on page 6-21.

Statement Examples

The example defines a VTAM buffer trace.

```
TRACE VTAMBU
```

Figure 7-32. Example: VTAMBU Trace Statement

VTAMIO Trace

```
TRACE VTAMIO ARea=BG|Fn    [-]
             [UNit=|CHannel=|CU=] [-]
             [OUTPut=]            [-]
             [OPTion=]
```

For UNit= see page 7-43, OUTPut= page 7-44, OPTion= page 7-48

A VTAMIO trace requires an area definition. Define the operands as shown above (BG|Fn).

Find the description of the trace and an example of the output under "VTAMIO Trace" on page 6-23.

Statement Example

The example defines a VTAMIO trace for the F3 tasks.

```
TRACE VTAMIO AREA=F3
```

Figure 7-33. Example: VTAMIO Trace Statement

Initialization Example

The following items are met by the trace set up shown below:

- Use the SYSRDR device to set up the trace.
- Trace all VTAM I/O operations concerning unit at address 020 and F3 tasks.
- Output the trace data to the tape on device address 280.

```
// EXEC SDAID
OUTDEV T=280
TRACE VTAMIO AR=F3 -
             UN=020
/*
```

Figure 7-34. Example: VTAMIO Trace Initialization

ADDITIONAL DEFINITIONS

```
ARea=,ADDRess=,OFFset=,PHase=,LTA SVA,OPTion=,OUTPut=,UNit=,CHannel=,CU=
```

The following section describes definitions which may follow the trace type specification in the TRACE statement.

The table below shows a list of all additional definitions, an overview of their function, and a page reference to their format description and examples in this chapter. The various definitions are described in detail under:

- "Define the Tasks to be Traced: AREA Definition" on page 6-24.
- "Define the Storage to be Traced: OFFSET, ADDRESS, PHASE" on page 6-26.
- "Define Additional Trace Output" on page 6-32.
- "Define Trace Options" on page 6-45.
- "Define the Traced I/O Devices" on page 6-46.

Operand	Function	Page
AREA	Limit tracing to a certain system area	7-39
ADDRESS	Limit tracing to a certain address range	7-41
OFFset	Limit tracing in a partition or phase area	7-41
PHase	Limit tracing to a certain phase	7-42
LTA SVA	Define additional trace area	7-42
UNit	Define the device address	7-43
CHannel	Define the channel address	7-43
CU	Define the control unit address	7-43
OUTPut	Define additional trace output	7-44
OPTion	Define additional trace options	7-48

Figure 7-35. Additional Definitions Summary

ARea Definition

```
ARea=partition-id [ADDress=][LTA|SVA|(LTA SVA)]              |
    =partition-id [OFFset=][LTA|SVA|(LTA SVA)]               |
    =partition-id [PHase= [OFFset=]][LTA|SVA|(LTA SVA)]|
    =SUPvr [OFFset=|ADDress=|PTA]                            |
    =ALL [ADDress=]                                          |
    =space-id [ADDress=]
```

The figure above shows the complete direct input mode format of the ARea definition. The ARea definition specifies those tasks whose activities should be traced via the following specifications:

- partition-id,
- SUPvr,
- ALL or space-id.

Only one ARea definition is possible.

The following sections show what other definitions you can combine with the various ARea specifications (for example ARea=ALL may be combined with ADDress= only)

Limit Tracing via AREA=Partition-Id

The possible storage area definitions together with ARea=partition-id, are:

```
ARea=partition-id [OFFset=][LTA|SVA|(LTA SVA)]       |
    =partition-id [ADDress=][LTA|SVA|(LTA SVA)]      |
    =partition-id [PHase= [OFFset=]][LTA|SVA|(LTA SVA)]
```

OFFset= See "OFFset Definition" on page 7-41.

ADDress= See "ADDress Definition" on page 7-41.

PHase= See "PHase Definition" on page 7-42.

LTA SVA In addition to the specifications OFFset, ADDress, or PHase, LTA, SVA or both enclosed in brackets may be defined. See "LTA SVA Operands" on page 7-42.

DEFAULT VALUE: If you use ARea=partition-id without an additional specification, OFFset=0:* is assumed. OFFset=0:* defines the whole partition as trace area.

Limit Tracing with ARea=SUPvr

The possible storage area definitions together with ARea=SUPvr are:

```
ARea=SUPvr [OFFset=|ADDress=|PTA]
```

OFFset= See "OFFset Definition" on page 7-41.

ADDress= See "ADDress Definition" on page 7-41.

PTA defines the physical transient area as the traced storage area.

DEFAULT VALUE: OFFset=0:* is active if no other specification is defined with ARea=SUPvr. The whole storage area defined to the supervisor is traced.

Limit Tracing with ARea=ALL

The possible storage area definition together with ARea=ALL is:

```
ARea=ALL [ADDress=]
```

ADDress= See "ADDress Definition" on page 7-41.

DEFAULT VALUE: ADDress=0:* is defined if you omit the ADDress keyword operand.

Limit Tracing with ARea=space-id

The possible storage area definition together with ARea=space-id is:

```
ARea=space-id [ADDress=]
```

ADDress= See ADDress Definition.

DEFAULT VALUE: ADDress=0:* is defined if you omit the ADDress keyword operand.

ADDress Definition

ADDress=addr1:addr2|addr1:*|0:*

You can limit the trace to a VSE/Advanced Functions system storage area with the ADDress definition.

addr1:addr2 Defines a trace address range in hexadecimal notation in any virtual storage defined to VSE/Advanced Functions.

For example:

```
ADD=203000:*
```

DEFAULT VALUE: If you omit the ADDress specification with the space-id definition 0:* is assumed, which means that all storage is traced.

OFFset Definition

OFFset=reladdr1:reladdr2|reladdr1:*|0:*

You can limit the trace to a storage area within the defined partition, supervisor or phase with the OFFset definition.

For example:

```
OFF=200:*
```

DEFAULT VALUE: If you omit the OFFset definition, 0:* is assumed.

PHase Definition

```
PHase=phase-name [OFFset=]
```

With the PHase definition the traced storage area is defined by the area occupied by that phase.

phase-name For example:

```
PH=PROGRAM1
```

DEVAULT VALUE: You may limit the traced storage area within the defined phase with the OFFset keyword operand. See "OFFset Definition" on page 7-41.

For example:

```
PH=PROG OFF=20:400
```

The example above initializes a trace which is active only in the phase with the name PROG between program address X'20' to X'400'.

LTA SVA Operands

```
LTA|SVA|(LTA SVA)
```

Additional trace areas may be specified by the operands LTA and SVA. If you specify both operands, enclose them in brackets.

> **Note:** Please see the performance considerations under "Performance Degradation if LTA and/or SVA is Defined" on page 5-9.

You can avoid the performance degradation when you define the areas explicitly, for example via the ADDress definition.

UNit, CU and CHannel Definitions

```
UNit=cuu|CU=cu|CHannel=c
```

UNit=cuu

For example:

```
UN=280
UN=(280 310)      note the brackets if you
                  specify more than one address.
UN=e              same as 00e
```

CU=cu

For example:

```
CU=28
CU=00
CU=(28 31)        note the brackets if you
                  specify more than one address.
```

CHannel=c

For example:

```
CH=2
CH=(2 3)
```

Only one of the parameters UNit, CHannel, or CU can be specified in the same TRACE command.

DEFAULT VALUE: If none of the I/O parameters is specified, all devices are traced.

OUTPut Definition

```
OUTPut=definition|(definition1 definition2)
```

You may take one or more definitions together with one TRACE statement. If you enter more than one OUTPut definition in direct input mode, enclose them in brackets.

Please find a summary of all definitions which you can specify with

OUTPut=....

in the table below. This table contains the format and a short description of the data which is recorded together with the trace event record. For those output definitions which allow additional definitions a page reference to the information contained in this chapter is shown.

Definition	What it records/prints in addition:	Page
BUffer	Contents of SDAID output buffer	-
CCB	CCB or IORB (TRACE=IO, SIO, or VTAMIO only)	-
CCW	CCW's (TRACE=IO,SIO, or VTAMIO only)	7-45
CCWD=nnnn	CCW's plus nnnn bytes of data (TRACE=IO, SIO, or VTAMIO only)	7-45
COMReg	Partition communication region	-
CREG	Control registers	-
DUMP	Virtual or real storage	7-46
FREG	Floating point registers	-
GREG	General purpose registers	-
IOTab	PUB, LUB, ERBLOC, ERRQ, CHANQ	-

Figure 7-36 (Part 1 of 2). OUTPut Definition Summary

Definition	What it records/prints in addition:	Page
LTA	Logical transient area	-
LOwcore	Processor storage from zero to X'BC'	-
PTA	Physical transient area	-
PIB	Partition information block	-
PTAB	Partition related control blocks	-
SUPvr	Supervisor plus GREG and CREG	-
SYSCom	System communication region	-
TOD	Time-of-Day clock	-
TTAB	Task related control blocks	-

Figure 7-36 (Part 2 of 2). OUTPut Definition Summary

Please note that a description of all output definitions is given under "Define Additional Trace Output" on page 6-32.

Record CCW

OUTPut=CCW|(CCWD=nnnn)

CCW (channel command word) records|prints the available channel program (CCW chain) plus the CCB and the TOD clock when the trace type is SIO.

In case of an IO trace only the CCWs which refer to transferred data are recorded or printed.

Specifying this output option for an event other than IO or SIO is not meaningful.

CCWD=nnnn (CCW plus data) records|prints nnnn bytes of the transferred data, the CCB and the TOD clock in addition to the information processed with the CCW specification.

The number nnnn may be any (decimal) number between 32 and 65535.

The most meaningful trace type to be combined with this output option is the IO trace.

Please find an example of the output produced with this option in Figure 6-18 on page 6-36.

You may define either CCW or CCD=nnnn.

Dump Virtual or Real Storage

```
OUTPut=DUMP partition-id [OFFset=]              |
            PHase [OFFset=]                     |
            ADDress=                            |
            REG=reg:bytes                       |
            PTR=reg:offset DMP=offset:length
```

DUMP records|prints the contents of virtual or real storage.

You may request up to ten different dumps.

You have to specify one or more of the dump area specifications as shown below.

Partition-id For example, dump the storage beginning with offset X'0' up to X'78' of the real BG partition (BGR, 370 mode).

```
OUTP=(DUMP BGR OFF=0:78)
```

PHase For example, dump the area starting with relative address X'40' up to relative address X'60' in the phase defined via the 'PHase=' keyword operand.

```
OUTP=(DUMP PH OFF=40:60)
```

ADDress For example, dump the contents of two bytes starting on storage location 0080 (hexadecimal). The definition in direct input mode looks like this:

```
OUTP=(DUMP ADD=80:81)
```

REG=reg:bytes For example, dump 16 bytes of storage pointed to by register 15.

```
OUTP=(DUMP REG=F:10)
```

PTR=reg:offset DMP=offset:length For example, dump a four-byte field which is located in a table with an offset of X'20' bytes. The table address is stored in storage pointed to by register 6 plus displacement X'100'. OUTP=(DUMP PTR=6:100 DMP=20:4)

Trace Statement Example: Dump an Area in a Phase

The following items are met by the trace set up shown below:

- Trace program check interrupts
- Traced tasks: BG partition main and subtasks
- Traced storage area: phase1 storage area
- Additional trace output:
  - general registers (GREG)
  - Dump of X'400' bytes of phase1 area starting at relative address 0 (DUMP PH OFF=0:400).
  - low-core (LOWC)

```
TRACE PGMC=* -
      AR=BG -
      PH=PHASE1 -
      OUTP=(GREG DUMP PH -
           OFF=0:400 LOWC)
```

Figure 7-37. Example: OUTPut=DUMP

OPTion Definition

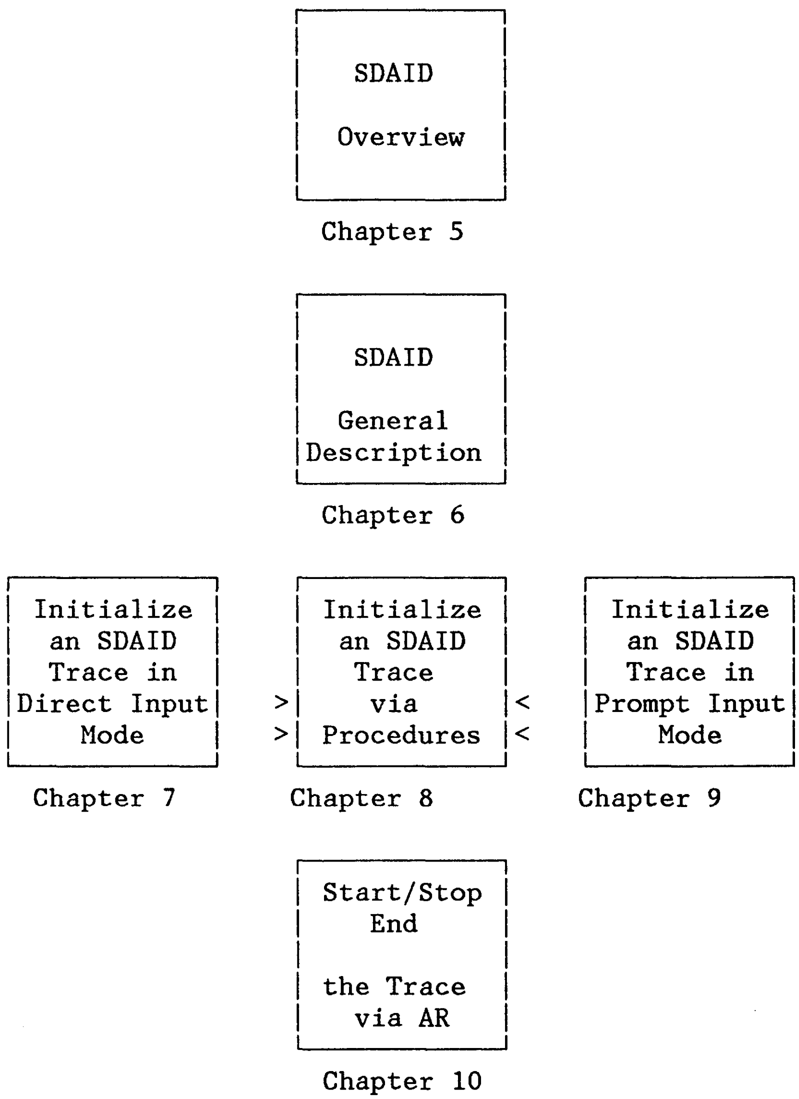
```
OPTion=definition|(definition1 definition2)

        HALT
        NOJCL
        TERMinate
        OCCurrence=occ1:occ2
```

Please find the description of the OPTion definitions under "Define Trace Options" on page 6-45.

OCCurrence Examples:

OPT=OCC=1:1	trace only the first occurrence of the event
OPT=OCC=1:*	trace all occurrences of the event (this is the default value)
OPT=OCCUR=5:12	trace selected occurrences (5 to 12) of the specified event

CHAPTER 8. INITIALIZE AN SDAID TRACE VIA A PROCEDURE



This chapter describes how you initialize SDAID traces by using just one job control (JCL) procedure statement. VSE/Advanced Functions offers a set of predefined JCL procedures to initialize SDAID traces under control of a partition; delivered to you as members of the system library IJSYSRS.

The most frequently used SDAID functions are covered by these JCL procedures. The JCL procedures contain reasonable default values to ease the SDAID trace initialization process. You may easily define your own procedures tailored to the requirements of your installation or to a special debugging problem.

Contents

INTRODUCTION

Besides the direct input mode and prompt mode trace initialization a third initialization method is available under VSE/Advanced Functions, the initialization via cataloged procedures. These cataloged procedures contain direct input mode command skeletons. You activate the initialization via the job control procedure statement.

```
// EXEC PROC=trace-type,specification,specification,...
```

The specifications in the procedure statement are translated to SDAID direct input mode jobs.

Each procedure statement contains the name of the procedure (trace-type) plus additional specifications. You may define the specifications in any order.
A continuation sign has to follow the comma if you use the console for input. If you use SYSIN to enter the procedure statement the continuation sign has to be in column 72.

Procedure Statement: Notational Conventions

Mandatory Operands

The description of the various procedure statements shows the mandatory operands in highlighted form.

Optional Operands

The operands which are enclosed in brackets are the optional operands.

Default Values

Two types of default values have to be considered.

- Defaults defined by SDAID
- Defaults defined by the cataloged procedure

The description in this chapter shows the defaults defined in the cataloged procedures by an underline.

The SDAID defined defaults are not shown in this chapter. The SDAID defaults are given under the description for each particular trace type in "The Various TRACE Types" on page 6-6.

Figure 8-1 on page 8-4 shows an example of a typical trace procedure statement.

```
// EXEC PROC=SDINST,AREA=partition-id[,OFFSET=|ADDRESS=|PHASE=OFFSET=]
                            =SUPvr[,OFFSET=|ADDRESS=]
                            =ALL|space-id[ADDRESS=]
                    [,INST=*|inst|'inst1 inst2 ..']
                    [,OUTPUT=GREG|'outp1 outp2 ..']
                    [,OPTION=NOJCL|'opt1 opt2 ...']
                    [,BUFFER=nn][,BUFFOUT= ][,TERM=]
                    [,Tape=cuu|Printer=cuu]
```

Figure 8-1. How the Procedure Statements are Described in this Chapter

Note the default values INST=*, OUTPUT=GREG and OPTION=NOJCL which are defined in the cataloged procedure.

The format of the procedure statement follows the conventions for procedure job control statements as described in VSE/Advanced Functions System Control Statements.

Default Value Considerations

How to Activate Defaults Defined by the Procedure

Some of the optional operands have default values defined in the cataloged procedure. The default value for a given keyword operand is activated if you omit the operand and the keyword, altogether.

For example, if you omit 'OUTPUT= ' in the procedure statement 'PROC=SDINST', the default 'OUTPUT=GREG' is activated.

How to Avoid the Procedure Defined Default Values

You may avoid the defaults set via the cataloged procedure if you enter the keyword without an operand.

For example, if you define 'OUTPUT= ' with the procedure SDINST, the default set in the cataloged procedure (OUTPUT=GREG) is omitted and no GREG output is provided.

DEFAULT VALUES WHICH YOU CANNOT AVOID: Some of the cataloged procedures contain fixed definitions which are active in every case.

For example the definition 'OUTPUT=TOD' is always active with the SIO procedure without regard to your 'OUTPUT' definition. These fixed definitions are mentioned in the description of the concerned trace procedures.

THE STATEMENTS OF A CATALOGED PROCEDURE

The result of the execution of each procedure statement is a complete direct input trace initialization. The direct input mode statements are cataloged in the following form:

FIXED DEFINITIONS are those definitions in the cataloged procedure which are always active without regard to your keyword operand definitions.

PLACEHOLDER DEFINITIONS are replaced by the value which you define in the '// EXEC PROC=' statement.

For example the 'UNIT=&UNIT' statement. The placeholder '&UNIT' is replaced by the unit address you specified.

The statement in the cataloged procedure:

```
UNIT=&UNIT
```

Your definition in the procedure statement:

```
UNIT='280 281'
```

The created direct input mode statement:

```
UNIT=(280 281)
```

PLACEHOLDER PLUS DEFAULT VALUE DEFINITIONS

Various default values are defined in the cataloged procedures. A default value is defined for example, for the output definition in the IO trace procedure in the following way:

```
OUTPUT=<CCWD=256>
```

Therefore, if you omit OUTPUT=, the default value is set into the direct input statement; it looks like this:

```
OUTPUT=CCWD=256
```

If you do not want to define any specification, and want to avoid the default also, your definition must look like this:

OUTPUT= with no operands

No operand defined in the procedure statement.

Figure 8-2 on page 8-6 shows an example of a fixed definition a placeholder definition and a placeholder plus default value definition. The shown cataloged procedure is called by the SDIO procedure statement.

```
// EXEC SDAID
TRACE SIO AREA=&area                  -
          UNIT=&unit                  -
          OUTPUT=TOD                  - <-- Fixed Definition
          OPTION=&option
TRACE IO  AREA=&area                  - <-- Placeholder Definition
          UNIT=&unit                  -
          OUTPUT=&output<CCWD=256>    - <-- Placeholder plus Default
          OPTION=&option                    Value Definition
/+
```

Figure 8-2. Example: Cataloged Procedure

Consider the two trace types which are initialized. If you do not specify the OUTPUT operand, the default OUTPUT=CCWD=256 is defined. The SIO trace event record contains always the time of day clock.

How to Create a Procedure

You may easily create and catalog your own procedures for a particular problem determination situation.

For example, you want to define additional default values or you want to create a procedure for a trace type for which no procedure has been cataloged.

When you create a procedure, consider that you have to follow the right command input sequence. For example, the TRACE= definition for some trace types has to be followed by the AREA specification. You can use the figures shown under "Command Input Path" on page 9-7 to establish the correct command input sequence.

PROCEDURES TO INITIALIZE SDAID TRACES

This section describes the trace procedures available with VSE/Advanced Functions to initialize SDAID traces. The additional keyword operands which you find in the trace procedure statements are described under "Additional Keyword Operands in Trace Procedure Statements" on page 8-17.

Choose the appropriate procedure from Figure 8-3.

Summary of Trace Procedures

Procedure	Provides Information on:	Page
SDBRANCH	successfully executed branch instructions	8-8
SDINST	selected or all instruction(s) execution	8-9
SDIO	I/O interrupts and SIO instructions	8-10
SDLOAD	phase load requests, or actual load	8-11
SDPGMC	program check interruptions	8-12
SDREG	contents or alterations of 1 to 8 registers	8-13
SDSTOR	storage alterations	8-14
SDSVC	executed supervisor calls	8-16

Figure 8-3. Trace Procedures Summary

Branch Trace Initialization

```
// EXEC PROC=SDBRANCH,AREA=partition-id[,OFFSET=|ADDRESS=|PHASE=OFFSET=]
                           =SUPvr[,OFFSET=|ADDRESS=]
                           =ALL|space-id[ADDRESS=]
                         [,OUTPUT= ]
                         [,OPTION=NOJCL|opt|'opt1 opt2 ...']
                         [,BUFFER=nn][,BUFFOUT= ][,TERM=]
                         [,Tape=cuu|Printer=cuu]
```

Find the "Additional Keyword Operands in Trace Procedure Statements" on page 8-17.

The procedure SDBRANCH initializes traces for all branch instructions which actually caused a branch.

Find the description of the trace type and an example of the output under "BRANCH Trace" on page 6-8.

Defaults Set in the Procedure

OPTION=NOJCL is active if you omit OPTION=.

Statement Example

Here are the items of the trace set-up shown below:

- Trace type: BRANCH
- Area for which events are collected: storage address 80010 up to address 80100
- Traced tasks: F4 main task and its subtasks
- Output destination: Printer with device address 00E
- Avoid the tracing of JCL instructions (default)

```
// EXEC PROC=SDBRANCH,AREA=F4,ADD='80010:80100',PRINTER=00E
```

Instruction Trace

```
// EXEC PROC=SDINST,AREA=partition-id[,OFFSET=|ADDRESS=|PHASE=OFFSET=]
                            =SUPvr[,OFFSET=|ADDRESS=]
                            =ALL|space-id[ADDRESS=]
                       [,INST=*|inst|'inst1 inst2 ..']
                       [,OUTPUT=GREG|'outp1 outp2 ...']
                       [,OPTION=NOJCL|'opt1 opt2 ...']
                       [,BUFFER=nn][,BUFFOUT= ][,TERM=]
                       [,Tape=cuu|Printer=cuu]
```

Find the "Additional Keyword Operands in Trace Procedure Statements" on page 8-17.

The procedure SDINST initializes traces for all instructions or for selected instructions executed within a specified area.
Find the description of the trace type and an example of the output under "INSTRUCTION Trace" on page 6-11.

Defaults Set in the Procedure:

If you omit INST=, all instructions are traced (*). OPTION=NOJCL and OUTPUT=GREG are defined if these two operands are omitted.

Statement Example

Here are the items of the trace set-up shown below:

- Trace type: INSTRUCTION
- Trace all instructions (default)
- Area for which events are collected: storage address 40328 up to address 40350
- Traced tasks: BG main and subtasks
- Additional Output: default GREG output
- Output destination: 16K bytes buffer
- Output device for buffer: tape with device address 281
- Event to write the buffer to tape: program check in BG partition
- Avoid the tracing of JCL instructions (default)
- See the conventions for the continuation sign under "Introduction" on page 8-3.

```
// EXEC PROC=SDINST,AREA=BG,ADDRESS='40328:40350', -
                           BUFFER=16,BUFFOUT=PGMC,T=281
```

// EXEC PROC=SDIO

```
         [,UNIT=cuu|'cuu1 cuu2 ..']
         [,AREA=BG|Fn]
         [,OUTPUT='CCWD=256'|'outp1 outp2 ...']
         [,OPTION= ]
         [,BUFFER=nn][,BUFFOUT= ][,TERM=]
         [,Tape=cuu|Printer=cuu]
```

Find the "Additional Keyword Operands in Trace Procedure Statements" on page 8-17.

The procedure SDIO initializes the SIO instructions and I/O interruptions trace.

Note, that the TOD clock entry is added to each SIO instruction event record.

Find the description of the trace types and examples of the output under "IO Trace (I/O Interrupt)" on page 6-12 and "SIO Instruction Trace" on page 6-18.

Default Set in the Procedure

OUTPUT='CCWD=256' is the default definition for the I/O interrupt trace.

If you do not define a certain unit address, all devices are traced.

All tasks in the system are traced if you omit the AREA definition (both SDAID defaults).

Statement Example

- Trace types: IO, SIO
- Traced tasks: SDAID default value used (ALL)
- Traced unit: 281
- Additional Output: procedure default CCWD=256
- Output destination: printer with device address 00E

```
// EXEC PROC=SDIO,UNIT=281,PRINTER=00E
```

Fetch/Load Trace

```
// EXEC PROC=SDLOAD,AREA=partition-id|SUPvr|
                          =ALL|space-id
                    [,PHASE=phasename]
                    [,ADDRESS=0:*|addr1:addr2]
                    [,OUTPUT= ][,OPTION= ]
                    [,BUFFER=nn][,BUFFOUT= ][,TERM=]
                    [,Tape=cuu|Printer=cuu]
```

Find the "Additional Keyword Operands in Trace Procedure Statements" on page 8-17.

The procedure SDLOAD initializes traces for all phase load requests and phase load operations.

Find the description of the trace type and an example of the output under "PGMLOAD (Fetch/Load) Trace" on page 6-16.

Defaults Set in the Procedure

ADDRESS=0:* is defined if you omit ADDRESS=.

Statement Example

Here are the items of the trace set-up shown below:

- Trace type: PGMLOAD
- Traced tasks: all tasks of the BG partition
- Traced storage area: whole VSE/Advanced Functions storage (default)
- Phase whose fetch/load operation is to be traced: MYPHASE
- Additional Output: dump of the storage contents with the address 0 to X'3000', relative to the BG partition start address on occurrence of the PGMLOAD trace event.
- Output destination: printer with device address 00E
- Note, that the continuation sign has to follow the comma if you use the console for input. If you use SYSIN to enter the procedure statement the continuation sign has to be in column 72.

```
// EXEC PROC=SDLOAD,PHASE=MYPHASE,AREA=BG,OUTPUT='DUMP BG OFF=0:3000,-
               P=00E
```

Program Check Trace

```
// EXEC PROC=SDPGMC,AREA=partition-id|SUPvr|
                        =ALL|space-id
                       [ PGMC=*|pgmc|'pgmc1 pgmc2 ..']
                       [,ADDRESS=0:*|addr1:addr2]
                       [,OUTPUT= ]
                       [,OPTION= ]
                       [,BUFFER=nn][,BUFFOUT= ][,TERM=]
                       [,Tape=cuu|Printer=cuu]
```

Find the "Additional Keyword Operands in Trace Procedure Statements" on page 8-17.

The procedure SDPGMC initializes traces for program check interruptions.

Find the description of the trace type and an example of the output under "PGMCheck Trace (Program Check)" on page 6-15.

Defaults Set in the Procedure

ADDRESS=0:* is defined if you omit ADDRESS=.

All program check interrupts are traced if you omit PGMC=.

Statement Example

Here are the items of the trace set-up shown below:

- Trace type: PGMCHECK
- Traced tasks: all tasks of the BG partition
- Traced storage area: BG partition area (OFF=0:* defined by SDAID defaults)
- Additional Output: dump of the storage contents with the address 0 to X'5000', relative to the BG partition start address on occurrence of the PGMCHECK trace event.
- Output destination: printer with device address 00E

```
// EXEC PROC=SDPGMC,AREA=BG,OUTPUT='DUMP BG OFFSET=0:5000',PRINTER=00E
```

Register Alteration Trace

```
// EXEC PROC=SDREG,REG=reg|'reg1 reg2 ..'
                  ,AREA=partition-id[,OFFSET=|ADDRESS=|PHASE=[OFFSET=]]
                                 =SUPvr[,OFFSET=|ADDRESS=]
                                 =ALL|space-id[ADDRESS=]
                       [,PATTERN=xxxxxxxx|'xxxxxxxx yyyyyyyy']
                       [,OUTPUT= ]
                       [,OPTION= ]
                       [,BUFFER=nn][,BUFFOUT= ][,TERM=]
                       [,Tape=cuu|Printer=cuu]
```

Find the "Additional Keyword Operands in Trace Procedure Statements" on page 8-17.

The procedure SDREG initializes traces for register alterations. All instructions which change the contents of the specified register(s) are traced. The optional parameter PATTERN restricts monitoring to those instructions which change the register contents pattern. The pattern may be up to four bytes long. If you specify an odd number of digits, zeros are inserted to the left.

Find the description of the trace type and an example of the output under "REGISTER Alteration Trace" on page 6-17.

Defaults Set in the Procedure

If the keyword PATTERN is omitted, all instructions are monitored which alter the specified registers.

Statement Example

Here are the items of the trace set-up shown below:

- Trace type: REGISTER
- Traced tasks: all tasks of the F3 partition
- Traced storage area: F3 partition area (OFF=0:* defined by SDAID default)
- Register whose contents alteration is traced: 15 defined as F
- Alteration to which value is traced: 00000008
- Output destination: printer with device address 00E

```
// EXEC PROC=SDREG,PATTERN=00000008,REG=F,AREA=F3,PRINTER=00E
```

Storage Alteration Trace

```
// EXEC PROC=SDSTOR,AREA=partition-id[,OFFSET=|ADDRESS=|PHASE=OFFSET=]
                         =SUPvr[,OFFSET=|ADDRESS=]
                         =ALL|space-id[ADDRESS=]
                  [,PATTERN=xxxxxxxx]
                  [,OUTPUT= ]
                  [,OPTION= ]
                  [,BUFFER=nn][,BUFFOUT= ][,TERM=]
                  [,Tape=cuu|Printer=cuu]
```

Find the "Additional Keyword Operands in Trace Procedure Statements" on page 8-17.

The procedure SDSTOR initializes traces for storage alterations.

You use this trace type as a tool to find those instructions which modify a certain storage area. In most cases you do not know which phase in your system alters this area. For this, define AREA=ALL to watch all tasks operating in your system. The observed storage area is defined via the ADDRESS= keyword.

The optional keyword 'PATTERN=' restricts monitoring to those instructions which change the storage contents into the defined pattern. The specified storage interval which you define with the ADDRESS= keyword should have the same length as the specified pattern (if any).

Find the description of the trace type and an example of the output under "STORAGE Alteration Trace" on page 6-20.

Statement Example

The example finds the instruction which alters a specified storage location into the pattern (FEFE); it puts the system into a wait state when this event occurred.

Here are the items of the trace set-up shown below:

- Trace type: STORAGE
- Traced tasks: all tasks in the VSE/Advanced Functions system
- Address area whose contents alteration is traced: X'074754' up to X'074755'
- Alteration value to be traced: FEFE
- Additional trace definition: put the system into a wait state on the event of the defined storage alteration.
- Output destination: printer with device address 00E

- Note that the continuation sign has to follow the comma if you use the console for input. If you use SYSIN to enter the procedure statement, the continuation sign has to be in column 72.

```
// EXEC PROC=SDSTOR,PATTERN=FEFE,AREA=ALL,ADDRESS='74754:74755',-
                    OPTION=HALT,P=00E
```

SVC Trace

```
// EXEC PROC=SDSVC,AREA=partition-id[,OFFSET=|ADDRESS=|PHASE=OFFSET=]
                        =SUPvr[,OFFSET=|ADDRESS=]
                        =ALL|space-id[ADDRESS=]
                    [,SVC=*|svc|'svc1 svc2 ..'
                    [,OUTPUT= ]
                    [,OPTION= ]
                    [,BUFFER=nn][,BUFFOUT= ][,TERM=]
                    [,Tape=cuu|Printer=cuu]
```

Find the "Additional Keyword Operands in Trace Procedure Statements" on page 8-17.

The procedure SDSVC initializes traces which provides event records for all or specified SVC instructions. Define the SVC code in hexadecimal form.

Find the description of the trace type and an example of the output under "SVC Trace (Supervisor Call)" on page 6-21.

Defaults Set in the Procedure

If you omit SVC=, all SVC instructions are traced.

Statement Example

Here are the items of the trace set-up shown below:

- Trace type: SVC
- SVC instruction defined by the SVC code: 3F
- Traced tasks: all tasks of the BG partition
- Traced storage area: BG partition area (SDAID default)
- Output destination: 10K bytes buffer
- Output device for buffer: printer with device address 00E
- Event to write the buffer to tape: cancel or EOJ condition in the BG partition

```
// EXEC PROC=SDSVC,AREA=BG,SVC=3F,P=00E,BUFFER=10,BUFFOUT=CANCEL
```

ADDITIONAL KEYWORD OPERANDS IN TRACE PROCEDURE STATEMENTS

You define the trace type with the procedure name. However, the initialization of a trace needs some other additional definitions, for example the specification of the output device.

The additional operands which are specific for the trace initialization via procedures are described in this section. The other additional operands have been described in Chapter 6, SDAID General Description.

The table shows all additional keyword operands in the format accepted in the procedure statement, a short description for each and a page reference to their detail description.

Operand	Function	Page
ADDRESS	Limit tracing to a certain address range	6-28
AREA	Limit tracing to a certain system area	6-24
OFFSET	Limit tracing to a partition or phase area	6-29
PHASE	Limit tracing to a certain phase	6-29
OPTION	Define additional trace options	6-45
OUTPUT	Define additional trace areas	6-32
UNIT	Define the device address	6-46
BUffer	Define the size of the output buffer	8-18
BUFFOUT	Define the event to write the buffer	8-18
TERM	Define the event which terminates the trace	8-19
Printer	Define the printer device address	8-18
Tape	Define the tape device address	8-18

Figure 8-4. Additional Keywords, Summary

Define the Output Device in a Procedure Statement

BUffer=, Printer=, Tape= Keyword Operands

[BUffer=nn] [Printer=cuu|Tape=cuu]

You define the output destination of the event trace records via the keyword operands BUffer=nn, Tape=cuu, or Printer=cuu.

BUffer=nn defines the size of a wrap-around buffer to collect the trace event records.

Note: The definition of a large wrap-around buffer may cause a lack of SDAID storage. For detailed information please refer to "Space Requirements for SDAID Execution" on page 5-10.

BUFFOUT= Keyword Operand

BUFFOUT=CANCEL|PGMC|FULL|EXT

Via the BUFFOUT= keyword operand you define the condition which forces the write buffer operation.

BUFFOUT=CANCEL defines that the contents of the wrap-around buffer is to be written to the output device (Printer or Tape) when a cancel or EOJ condition occurs.

BUFFOUT=PGMC defines that the contents of the wrap-around buffer is to be written to the output device (Printer or Tape) on any program check interruption (except page faults).

Note: If you specify BUFFOUT=CANCEL or BUFFOUT=PGMC the keyword operand 'AREA=BG|Fn' has to be specified, too.

BUFFOUT=FULL defines that the buffer is to be written to the output device whenever it is full.

BUFFOUT=EXT defines that the buffer is to be written to the output device whenever the external interrupt key is pressed.

TERM= Keyword Operand

TERM=CANCEL|PGMC|EXT

TERM=CANCEL defines that tracing is to be terminated as soon as a cancel condition occurs in the traced partition.

TERM=PGMC defines that tracing is to be terminated as soon as a program check occurs in the traced partition.

TERM=EXT defines that tracing is to be terminated as soon as the external interrupt key is pressed.

Note: If TERM=CANCEL or TERM=PGMC is specified, the AREA=BG|Fn has to be specified also.

CHAPTER 9. INITIALIZE A TRACE IN PROMPT INPUT MODE

SDAID

Overview

Chapter 5

SDAID

General
Description

Chapter 6

Initialize an SDAID Trace in Direct Input Mode	Initialize an SDAID Trace via Procedures	> Initialize an SDAID Trace in Prompt Input Mode <
Chapter 7	Chapter 8	Chapter 9

Start/Stop
End

the Trace
via AR

Chapter 10

This chapter describes how you initialize an SDAID trace in prompt input mode. The prompt input mode works only in the attention routine.

Contents

OVERVIEW

Beside the trace initialization in direct input mode or via procedures, you can use the prompt mode, which operates in the attention routine, to set-up SDAID traces.

You invoke the program SDAID with the attention routine (AR) command **SDAID.**

You get into prompt mode if you enter the SDAID commands in the AR without another specification. The prompt mode is also activated if you process direct input mode commands in the attention routine with at least one prompt mode statement, like the question mark (?).

The trace output device is defined via prompts after you enter the **OUTDEV** command.

SDAID prompts you for the trace type when you enter the **TRACE** command in the AR.

You end the initialization process with the **READY** command.

Once you have initialized the SDAID trace, attention routine commands are used to start the trace execution (STRTSD), interrupt it (STOPSD), and end it (ENDSD).

The trace output, an **event record,** is supplied for each occurrence of a traced event, according to your instructions.

You may request the event records to be written to a line printer, onto a magnetic tape, or into a wraparound buffer. The definition of the output device is given via the prompts following the OUTDEV command.

Please find the prompts and the possible replies under "Command Input Path" on page 9-7.

HOW TO INITIALIZE AN SDAID TRACE IN PROMPT MODE

An SDAID trace initialization in prompt mode is based on the following commands:

Command	Description	Page
SDAID	Attention routine command to invoke the SDAID program.	-
OUTDEV	Defines output device for the trace (printer, tape, or buffer).	9-16
TRACE	Defines the event(s) to be traced. At least one TRACE command is required; up to ten may be submitted.	9-18
READY	Ends input of initialization commands OUTDEV and TRACE.	-

Figure 9-1. Input Command Summary

The Various SDAID Commands

SDAID prompts you for the output device of the trace when you enter **OUTDEV**.

One OUTDEV definition can be active in the system at one time. Any newly entered OUTDEV command overwrites the existing one.

Enter **TRACE** to be prompted by SDAID for the type(s) of traces you want. Up to ten 'TRACE' commands may be entered in one session.

You end the trace initialization in the attention routine with the **READY** command. When the READY command has been processed, no further OUTDEV or TRACE command can be entered.

Sample SDAID Trace Initialization

Figure 9-2 shows a typical trace initialization session.

The session starts with the AR command 'SDAID'. With the command 'OUTDEV' the output device is defined and the command 'TRACE' is entered to specify the trace type. The initialization process ends with the READY command.

```
Prompt Set-up via the Attention Routine

-> sdaid □
   4C05I PROCESSING OF 'SDAID'  COMMAND SUCCESSFUL
-> outdev □
   4C08D SPECIFY OUTPUT DEVICE.+
-> buffer □
   4C08D SPECIFY SIZE OF WRAP BUFFER.+
-> 8 □
   4C08D SPECIFY OUTPUT DEVICE FOR BUFFER.+
-> tape □
   4C08D SPECIFY PHYSICAL ADDRESS OF PRINTER/TAPE.+
-> 281
   4C05I PROCESSING OF 'OUTDEV' COMMAND SUCCESSFUL.
-> trace □
   4C08D SPECIFY TRACE TYPE.+
-> inst □
   4C08D SPECIFY OP CODES OR * OR BR.
-> *
   4C08D SPECIFY TRACE AREA.+
-> bg □
   4C08D SPECIFY TYPE OF LIMITS.+
-> □
   4C08D SPECIFY ADDITIONAL TRACE AREA.+
-> □
   4C08D SPECIFY OUTPUT.+
-> □
   4C08D SPECIFY OPTIONS.+
-> nojcl □
   4C08D SPECIFY OPTIONS.+
-> □
   4C05I PROCESSING OF 'TRACE'  COMMAND SUCCESSFUL.
-> ready
   4C05I PROCESSING OF 'READY' COMMAND SUCCESSFUL.
   AR 015 1I40I READY.
```

Figure 9-2. Example: Prompt Mode Trace Initialization

Notational Conventions

- SDAID messages (or help information) are shown in upper-case with a message number.
- Responses or commands for you to enter are shown in lower-case. In most cases a short form of the command is also allowed.
- It is possible to abbreviate the SDAID parameters. For example the BRanch trace type specification can be abbreviated in the following way:

  BR BRa BRan BRanc BRanch

  Note that the minimum definition is indicated through capital letters.
- The symbol □ indicates you are to press the ENTER key (generally after entering any response or command).

How to Use Help and Cancel in Prompt Mode

- Messages for which you can request additional help information are indicated by a plus sign (+) at the end of the message.

- Request additional help by entering a question mark (?).

- You can cancel data entered for the current command by entering two question marks (??).

  Figure 9-3 shows how you can request help information and how the initialization process can be canceled.

```
-> trace □
   4C08D SPECIFY TRACE TYPE.+
-> ? □
   ENTER ONE OF THE FOLLOWING KEYWORDS:
    SVC         PGMCHECK    MONITOR     CANCEL
    INSTR       REGISTER    STORAGE     BRANCH
    PAGING      PGMLOAD     EXTERNAL    BUFFER
    IO          SIO         VTAMIO      VTAMBU
-> ?? □
   4D03I COMMAND CANCELED DUE TO USER REQUEST
```

Figure 9-3. Help and Cancel Initialization Example

COMMAND INPUT PATH

This section shows the prompt messages and the possible replies in the sequence of their processing.

Figure 9-4 shows an example of the trace statement path. You can find the possible input in accordance to the prompt message 'SPECIFY TRACE TYPE' (BR, CA, ..). The example also indicates the prompt message after the reply 'inst' (SPECIFY OP * OR BR).

```
TRACE
 |  ┌── SPECIFY TRACE TYPE
 |  V
 └─>O──BRanch──────────────────────────────────────>
    |
    ├──CAncel──────────────────────────────────────>
    |
    |              ┌── SPECIFY OP CODES OR * OR BR
    |              V
    ├──INSTr───────O────── * ──────┬───────────────>
    |              └──op-codes─────┘
    v              └──BRanch───────┘
```

Figure 9-4. Sample Command Input Path

NOTATIONAL CONVENTIONS

The diagrams use a solid line, or a number of solid lines in parallel, as a specification path. Follow the line of the option that you select for your SDAID execution.

```
────O──────                     Indicates a prompting message.
    │
```

```
──O......O:*......┌──>      Three options, the dotted line indicating
  └─addr1:*──────>│         a default (in this case from 0 to end);
  └─addr1:addr2───┘             Hit END/ENTER for the default.
```

```
─O......... □ .........┌──>  Same as above, but END/ENTER indicates
 └─pattern1───────────>│      no pattern instead of a default.
 └─pattern1 pattern2───┘
```

```
─O...... □ .......┌──>      (,) Indicates that the optional operands
 │  ┌── , ──┐      │             may be entered more than once and in
 │  V       │      │             any sequence.
 └─(└─┬─LTA ┌┘─)>│                Hit END/ENTER when you have finished
      └─SVA─┘                     the input or none of the options is
                                  desired.
```

```
┌───┐
│ n │       On-page connector.
└───┘
```

```
┌───┐
│ n │       Off-page connector (to a following page).
└─V─┘
```

```
──>;        Command input completed.
```

OUTDEV Command Input Path

```
                ┌── SPECIFY OUTPUT DEVICE
                |
                |         ┌── SPECIFY SIZE OF WRAP BUFFER
                |         |
                |         |         ┌── SPECIFY OUTPUT DEVICE FOR BUFFER
                |         |         |
                V         V         V
OUTDEV>──>O──BUffer──O── nn ──O......... □ ..............────>;
                |                   |
                |                   |  SPECIFY PHYSICAL ADDRESS OF PRINTER/TAPE
                |                   |                  |
                |                   |                  V
                |                   |──Printer──>┌───O── cuu ──────>;
                |                   └──Tape ────>|
                |                                |
                |──Printer──────────────────────>|
                |                                |
                └──Tape──────────────────────────┘
```

Figure 9-5. OUTDEV Command: Syntax Diagram

TRACE Command Input Path

```
 TRACE
 |   ┌── SPECIFY TRACE TYPE
 |   V
 └─>O─BRanch──────────────────────────────────────────────────>┐
    |                                                          |     ┌───┐
    |─CAncel──────────────────────────────────────────────────>└──>| 2 |
    |                   ┌── SPECIFY TYPE OF EXTERNAL INTERRUPT       └───┘
    |                   V                                            ┌───┐
    |─EXTernal──────────O.......ALL.......┌───────────────────────>| 6 |
    |                   |  ┌──── , ────┐  |                          └───┘
    |                   |  V           |  |
    |                   └─(──┌──KEY──┌─┘ )─┘
    |                        |-TIMER-|
    |                        └─SIGNAL┘
    |
    |                   ┌-- SPECIFY OP CODES OR * OR BR
    |                   V
    |─INSTr─────────────O───── * ──────┌───────────────────────>┐
    |                   └──op-codes────┘                         |
    |                   └──BRanch──────┘                         |
    |                                                            |
    |                   ┌─── SPECIFY MONITOR CALL CLASS(ES) OR * |
    |                   V                                        |
    |─MONitorcall─O──────── * ────────────┌──────────────>|
    |                   └────monitorclass-list──┘                |
    |                                                            |
    |─PAGing──────────────────────────────────────────────────>|
    |                                                            |
    |                   ┌─── SPECIFY PGM INTERRUPT CODE(S) OR *  |
    |                   V                                        |
    |─PGMCheck──────────O───── * ──────┌───────────────────────>|
    |                   └──pgmc-codes──┘                         |
    |                                                            |
    | SPECIFY PGMLOAD TRACE OPTIONS                              |
    |                   |             SPECIFY NAME OF PHASE      |
    |                   |             TO BE MONITORED, OR PRESS  |
    |                   |                 |      END/ENTER       |
    |                   |                 |                      |
    |                   V                 V                      |     ┌───┐
    |-PGMLoad───────────O...... ALL .......┌──O... □ .(all)...┌──>└──>| 2 |
    |                   |  ┌──── , ────┐   |  └───phasename───┘         └─V─┘
    |                   |  V           |   |                         AREA definition
    |                   └─(───┌─ HDL -┌┘-)─┘
    |                         └─ REQ ─┘
    |
  ┌───┐
  | 1 |
  └─V─┘
```

Figure 9-6 (Part 1 of 6). TRACE Command: Syntax Diagram

```
┌───┐
│ 1 │ (TRACE TYPE, CONT'D)
└─V─┘
  │
  │          ┌── SPECIFY REGISTER NUMBER(S)
  V          │
  │          │                  ┌── SPECIFY REGISTER
  │          V                  V          PATTERN
  │─REGister──O─reg-numbers──────O.....□..(every).....┌─┐
  │                             └─pattern1──────────>│ │
  │                             └─pattern1 pattern2──┘ │
  │                                                    │
  │          ┌── SPECIFY STORAGE PATTERN                │
  │          V                                         │
  │─STorage───O.......... □ ..(every).┌──────────────>│
  │          └───────────pattern──────┘                │
  │                                                    │
  │          ┌── SPECIFY SVC NUMBER(S) OR *             │
  │          V                                         │  ┌───┐
  │─SVC───────O───── * ─────┌──────────────────────────└─>│ 2 │
  │          └─svc-numbers──┘                              └─V─┘
  │                                                    AREA definition
  │              ┌── SPECIFY PARTITION ID
  │              V                          ┌───┐
  │─IO──────┌─────O....ALL..........>┌─────>│ 5 │ I/O definition
  │         │    │───partition-id──>│       └─V─┘
  │─SIO─────┘    └───SUPvr──────────┘
  │
  │
  │              ┌── SPECIFY PARTITION ID
  │              V                          ┌───┐
  │─VTAMIO────────└───partition-id─>──────>│ 5 │ I/O definition
  │                                         └-V-┘
  │                                         ┌───┐
  │─VTAMBU─────────────────────────────────>│ 6 │ Trace output definition
  │                                         └─V─┘
  │                                         ┌───┐
  └─BUffer─────────────────────────────────>│ 8 │ Occurrence
                                            └─V─┘ definition
```

Figure 9-6 (Part 2 of 6). TRACE Command: Syntax Diagram

```
        ┌── SPECIFY TRACE ┌─── SPECIFY TYPE OF LIMITS
        |   AREA         |            ┌── SPECIFY OFFSET RANGE
   ┌──┐ V                V            V
   | 2 |>|- partition-id──>0..OFFset....>0.........0:*...............┐
   └──┘  |                |            └──reladdr1:*──────────────>|
         |                |            └──reladdr1:reladdr2───────>|
         |                |                                         |
         |                |            ┌── SPECIFY ADDRESS RANGE    |
         |                |            V                            |
         |                |──ADDress────>0......0:*................>|
         |                |            |──addr1:*─────────────────>|
         |                |            └──addr1:addr2─────────────>|
         |                |                                         |   ┌──┐
         |                |                                         └─>| 4 |
         |                |                                             └──┘
         |                |            ┌── SPECIFY NAME OF PHASE
         |                |            V                                ┌──┐
         |                └──PHase──────>0──────phasename─────────────>| 3 |
         |                                                              └──┘
         |                ┌── SPECIFY TYPE OF LIMITS
         |                |              ┌── SPECIFY OFFSET RANGE
         |                V              V                                ┌──┐
         |─SUPvr──────────0.....OFFset....0.........0:*..............┌─>| 6 |
         |                |              └──reladdr1:*─────────────>|    └─V─┘
         |                |              └──reladdr1:reladdr2──────>|
         |                |              ┌── SPECIFY ADDRESS RANGE  |
         |                |              V                           |
         |                |──ADDress─────>0......0:*...............>|
         |                |              |──addr1:*────────────────>|
         |                |              └──addr1:addr2────────────>|
         |                |                                          |
         |                |                                          |
         |                └───────PTA────────────────────────────────┘
         |
         |                ┌── SPECIFY ADDRESS RANGE
         |                V                                        ┌──┐
         |- space ID ─┌───0.......0:*.........┌──────────────────>| 6 |
         |            |   └───addr1:*────────>|                    └─V─┘
         |            |   └───addr1:addr2─────┘
         └──── ALL ───┘


  SPECIFY OFFSET RANGE              ┌──┐    ┌-- SPECIFY ADDITIONAL
          |                         | 4 |    |   TRACE AREA
          V                         └──┘    |
  ┌──┐    V                          V       V                        ┌──┐
  | 3 |──────>0.........0:*.........  ┌──────────>0....... □ .(none)┌─>| 6 |
  └──┘        └──reladdr1:*────────>|             |    ┌──── , ────┐   |  └─V─┘
              └──reladdr1:reladdr2──┘             |    V           |   | Output de-
                                                  └─(──┌─LTA─┌──┘─)─┘  finition
                                                       └-SVA-┘
```

Figure 9-6 (Part 3 of 6). TRACE Command: Syntax Diagram

```
               ┌── SPECIFY RANGE OF IO/SIO TRACE
               V
┌───┐                                                        ┌───┐
| 5 |─────────>O..... □.......(all units traced)..........┌────>| 6 |
└───┘          |                                          |   └───┘
  I/O          |                ┌-SPECIFY CHANNEL ADDRESS  |
definition     |                V                          |
               |────CHannel────O──channel address(es)─┌───>|
               |                └───────── , ────────────┘  |
               |                                           |
               |                ┌─── SPECIFY CU ADDRESS    |
               |                V                          |
               |────CU─────────O──control unit(s)───────┌───>|
               |                └───────── , ────────────┘  |
               |                                           |
               |                ┌─── SPECIFY UNIT ADDRESS  |
               |                V                          |
               └────UNit───────O──device address(es)──┌─────┘
                                └───────── , ────────────┘
            ┌── SPECIFY OUTPUT
            V
┌───┐                                                                    ┌───┐
| 6 |────>O............................... □ ...(no additional output)..┌─>| 8 |
└───┘      |                 ┌──────────────── , ─────────────────┐   |  └─V─┘
trace      |                 V                                    |   | Option
output     └───────────────(───┬-GReg─────────────────────────────┌──┘─)─┘ defini-
definition OUTPut              |-CReg────────────────────────────>|          tion
                               |-FReg────────────────────────────>|
                               |-COMReg──────────────────────────>|
                               |-SYSCom──────────────────────────>|
                               |-IOTab───────────────────────────>|
                               |-LTA─────────────────────────────>|
                               |-PTA─────────────────────────────>|
                               |-PIB─────────────────────────────>|
                               |-CCB─────────────────────────────>|
                               |-CCW─────────────────────────────>|
                               |-TOD─────────────────────────────>|
                               |-LOwcore─────────────────────────>|
                               |-Supvr───────────────────────────>|
                               |-BUffer──────────────────────────>|
                               |-PTAB────────────────────────────>|
                               |-TTAB────────────────────────────>|
                               |        ┌──SPECIFY LENGTH OF DATA  |
                               |        |                         |
                               |-CCWD───O───── data length ──────>|
                               |                                  |
                               |        ┌──SPECIFY DUMP AREA      |
                               |        V                         |  | See also
                               └-DUMP───O───── dump definition >──┘  | NOTE 1 and 3
                                                    |        ┌───┐   | at the end
                                                    └─see───>| 7 |   | of the dump
                                                             └─V─┘   | definition
```

Figure 9-6 (Part 4 of 6). TRACE Command: Syntax Diagram

```
        ┌── SPECIFY DUMP AREA (see Note 1)
        |
        |
        |                      ┌── SPECIFY OFFSET RANGE
┌──┐    V                      V
| 7 |-O─partition-id ─────────O......0:*.................┐
└──┘  |(partition              |-relad1:*──────────────>|
      |     -id)               └─relad1:relad2 ─────────>|
      |                                                  |
      |                                                  |
      |                                                  |
      |                        ┌── SPECIFY OFFSET RANGE  |
      |                        V                         |
      |─PHase ───────────────O.......0:*...............>|
      |  ───                  |─relad1:* ──────────────>|
      |    |                   └─relad1:relad2 ─────────>|
      |    └───see Note 2                                |
      |                                                  |
      |                        ┌── SPECIFY ADDRESS RANGE |
      |                        V                         |
      |─ADDress──────────────O─────0:*─────────────────>|
      |                        |─addr1:*────────────────>|
      |                        └─addr1:addr2────────────>|
      |                                                  |
      |                                                  |
      |                        ┌── SPECIFY DUMP RANGE    |
      |                        V                         |
      |─REG──────────────────O───reg:bytes─────────────>|
      |                                                  |
      |                                                  |
      |     ┌── SPECIFY REGISTER    ┌─SPECIFY OFFSET     |
      |     |   AND OFFSET           |  AND DUMP LENGTH  |
      |     V                        V                   |         ┌───┐
      └─PTR─O─reg:offset───────────O─offset:length─> └─────────>| 8 |
                                                                   └-V-┘
```

Notes:

1. Up to ten different areas may be specified with DUMP.
2. Can be specified only if a phase was previously defined in the area definition of the TRACE.
3. You need not specify the word OUTPut in prompt mode. SDAID prompts you for the definition of the additional output.

Figure 9-6 (Part 5 of 6). TRACE Command: Syntax Diagram

```
┌───┐
| 8 |
└───┘
 |  ┌── SPECIFY OPTIONS
 |  V
 └──O....... □ ..(no option).......................┌─────> ;
    |                                              |
    |  ┌─────── , ──────────────────────────────┐  |
    |  V                                        |  |
    |  ┌──────HALT──────────────────────────┐   |  |
    |  |                                    |   |  |
    |  |──────NOJCL─────────────────────────>|  |  |
    |  |                                    |   |  |
    └(─O───TERMinate────────────────────────>┌──┘─)┘
       |                 ┌── SPECIFY OCCURRENCE |
       |                 V   RANGE              |
       └───OCCURRENCE────O─────── □ ──(all)────>|
                         |────────1:*──────────>|
                         └───────val1:val2──────┘
```

Figure 9-6 (Part 6 of 6). TRACE Command: Syntax Diagram

OUTPUT DEVICE DEFINITION IN PROMPT MODE: OUTDEV COMMAND

This section shows promptings and possible replies for the trace output device definition. Detailed information about the output device is given under "Define the Output Device" on page 6-3.

When you enter the OUTDEV command, SDAID prompts you for control information as follows:

```
outdev
4C08D SPECIFY OUTPUT DEVICE.+
```

As a response to this prompting message, enter one of the following:

Printer If printer is specified, the event records are written to a line printer at the time the particular event occurs.

Tape When tape is defined, the trace records are written to tape in the form of 3K bytes blocks.

SDAID prompts you for the address of the output device in the following way:

```
4C08D  SPECIFY PHYSICAL ADDRESS OF PRINTER/TAPE.
```

Enter the device address in the form **cuu**.

BUffer writes the trace output to a wraparound buffer. SDAID prompts you for the size of the buffer as follows:

```
4C08D  SPECIFY SIZE OF WRAP BUFFER.+
```

Enter the desired size of the buffer in number of blocks of 1K bytes.

POSSIBLE BUFFER SIZES:

The possible buffer sizes depend on the output device for the buffer which is defined next.

Buffer to Printer or no output device	3K - 99K
Buffer to Tape	3K - 32K

Now, SDAID prompts you as follows:

```
4C08D  SPECIFY OUTPUT DEVICE FOR BUFFER.+
```

Respond with either **Printer, Tape, or END/ENTER**.

Pressing END/ENTER causes no output device being defined.

SPECIFYING THE TRACE: TRACE COMMAND

Once you enter the command: **TRACE**, SDAID prompts you for the following control information:

Trace-type definition: The type of event to be traced.

=======> "Trace Type Definition" on page 9-19.

Area definition: The range of the trace in storage.

=======> "AREA Definition" on page 9-31.

I/O definition: Limits a trace operation to one or more channels, control units, or devices.

=======> "I/O Definition" on page 9-35.

Output definition: Additional (optional) trace information that is required to analyze the particular problem.

=======> "Additional Output Definition" on page 9-36.

Option definition: An option to:

- Stop system execution when the specified trace event occurs.
- Discontinue tracing when the specified trace event occurs.
- Avoid tracing of Job Control phases
- Discontinue tracing when a defined number of events has been exceeded.

=======> "Option Definition" on page 9-38.

You will find sample event records and a description of most of the trace types under "The Various TRACE Types" on page 6-6.

TRACE TYPE DEFINITION

This section shows promptings and the possible replies to define an SDAID trace.

Detailed information about the various trace types is given under "The Various TRACE Types" on page 6-6.

You start the definition of your trace with the trace command in the attention routine.

```
trace
4C08D SPECIFY TRACE TYPE.+
```

Respond to the prompting message with any of the available trace types. For example if you want to initialize a branch trace, the response would look like this:

```
trace
4C08D SPECIFY TRACE TYPE.+
branch      <------------ Your response
```

The SDAID then prompts you for additional information.

Please find a summary of the SDAID trace types in Figure 9-7 on page 9-20.

Summary of Trace Types

Figure 9-7 gives the following information:

- The trace types shown in the format they can be entered. Note, that the upper case letters indicate the shortest possible abbreviation.
- A short description of the trace type. All trace types are described in more detail under "The Various TRACE Types" on page 6-6.
- A page reference to the format description of the trace type.

Trace Type	Provides a Trace of:	Page
BRanch	Successfully executed branch instructions	9-21
BUffer	Output the trace buffer when it is full	9-21
CAncel	Program (main task) cancel or EOJ	9-21
EXTernal	External interrupts	9-21
INSTruction	Selected or all instruction(s) execution	9-22
IO	I/O interrupts	9-23
MONitorcall	MC instructions	9-23
PAGing	Page faults	9-24
PGMCheck	Program checks	9-24
PGMLoad	Phase load requests, or actual load	9-25
REGister	Contents or alterations of 1 to 8 registers	9-26
SIO	SIO instructions	9-28
STorage	Storage alterations	9-28
SVC	Executed supervisor calls	9-29
VTAMBU	Usage of ACF/VTAM buffers	9-30
VTAMIO	VTAM I/O operations	9-30

Figure 9-7. Trace Type Summary

BRanch Trace

```
trace
4C08D SPECIFY TRACE TYPE.+
branch       <------------- Your response
```

The SDAID prompts you for the definition of the AREA as next. See

=======> "AREA Definition" on page 9-31.

BUffer Trace

```
trace
4C08D SPECIFY TRACE TYPE.+
buffer       <------------- Your response
```

The SDAID prompts you for the OCCurrence definition as next. Please find the format of this definitions under:

=======> "Option Definition" on page 9-38.

CAncel Trace

```
trace
4C08D SPECIFY TRACE TYPE.+
cancel       <------------- Your response
```

The SDAID prompts you for the definition of the AREA as next. See

=======> "AREA Definition" on page 9-31.

EXTernal (External Interrupt) Trace

```
trace
4C08D SPECIFY TRACE TYPE.+
external     <------------- Your response
```

The SDAID prompts you for additional control information until you press END/ENTER, as follows:

```
4C08D SPECIFY TYPE OF EXTERNAL INTERRUPT.+
```

Your response may be one of the following:

ALL or END/ENTER to trace all types of external interrupts.

KEY to trace only key interrupts.

TIMER to trace timer interrupts.

SIGNAL to trace CPU signals.

END/ENTER to continue.

The SDAID now prompts you for the definition of the OUTPUT. See

=======> "Additional Output Definition" on page 9-36.

INSTr (Instruction Execution) Trace

```
trace
4C08D SPECIFY TRACE TYPE.+
instr        <------------- Your response
```

The SDAID prompts you for additional control information, as follows:

```
4C08D SPECIFY OP CODE(S) OR BR OR *.+
```

Your response may be one of the following:

op code(s) (one to eight) entered as either one-byte or two-byte hexadecimal values. If you specify more than one operation code, separate them by one or more blanks or by a comma (with or without blanks).

asterisk (*) defines all op codes.

BRanch defines that all executed branch instructions have to be traced regardless whether the branch has been taken or not.

Sample responses:

```
d2
18 41,58 40, 50 9608
*
```

The SDAID now prompts you for the definition of the AREA. See

=======> "AREA Definition" on page 9-31.

IO (I/O Interrupt) Trace

```
trace
4C08D SPECIFY TRACE TYPE.+
io   <-------------------- Your response
```

SDAID prompts you for the definition of the partition id as shown below:

```
4C08D SPECIFY PARTITION ID.+
```

You may select a certain partition, for example BG or F3, or the supervisor defined as SUPvr.

If you press END/ENTER all tasks in the system are traced.

SDAID then prompts you for the definition of the specific I/O channel(s), control unit(s), or unit(s), as described under

=======> "I/O Definition" on page 9-35.

MONitorcall Trace

```
trace
4C08D SPECIFY TRACE TYPE.+
monitorcall <------------- Your response
```

SDAID prompts you for additional control information as follows:

```
4C08D SPECIFY MONITOR CALL CLASS(ES) OR *.+
```

Your response may be one of the following:

monitor classes defines the MC instructions to be traced by one or up to eight monitor classes.

Monitor classes must be specified as one-digit hexadecimal values. If you specify two or more classes, separate them by one or more blanks, or by a comma with or without blanks.

You may specify any valid monitor class; however, SDAID ignores a specification of class 2.

asterisk (*) defines all classes except class 2.

Sample responses:

```
3 5, 8,c d e f
a
*
```

The SDAID now prompts you for the definition of the AREA. See

=======> "AREA Definition" on page 9-31.

PAGing Trace

```
trace
4C08D SPECIFY TRACE TYPE.+
paging      <------------- Your response
```

The SDAID prompts you for the ARea definition. See

=======> "AREA Definition" on page 9-31.

PGMCheck (Program Check) Trace

```
trace
4C08D SPECIFY TRACE TYPE.+
pgmcheck    <------------- Your response
```

SDAID prompts you for additional control information as follows:

```
4C08D SPECIFY PROGRAM INTERRUPT CODE(S) OR '*'.+
```

Your response may be one of the following:

Program codes (one to 16) must be specified in hexadecimal notation; leading zeros may be omitted, or an asterisk (*) for all program check interruption codes except the page fault interruption.

If you specify more than one program interrupt code, separate them by one or more blanks, or by a comma with one or more blanks.

```
Sample specifications:   1 13,05, 10
                         9
                         *
```

The SDAID now prompts you for the definition of the AREA. See

=======> "AREA Definition" on page 9-31.

PGMLoad (program load) Trace

```
trace
4C08D SPECIFY TRACE TYPE.+
pgmload     <------------- Your response
```

SDAID prompts you for additional control information as follows:

```
4C08D SPECIFY PGMLOAD TRACE OPTIONS.+
```

Your response may be one of the following:

Hit END/ENTER writes an event record for all program load events (phase load, fetch request, or actual phase-load operation) within the specified trace range.

req writes an event record each time loading/fetching a phase is requested (see notes 1 and 3 below).

hdl writes an event record each time a phase load/fetch request is handled; that is, when a requested phase is actually loaded into storage for execution (see notes 1 and 3 below).

all writes an event record each time a phase load/fetch request occurs, and also each time a phase is actually loaded into storage for execution. This is the default (see notes 1 and 2 below).

Notes:

1. When you have entered a response to the above prompting message, SDAID repeats the prompting message until you respond by hitting END/ENTER.

2. If you want all program-load events to be traced, respond by hitting END/ENTER when SDAID displays the above prompting message for the first time.

3. If you want only one phase to be traced, submit the name of this phase after specifying HDL, REQ or ALL.

SDAID prompts you for additional control information as follows:

```
4C08D SPECIFY NAME OF PHASE TO BE MONITORED, OR PRESS END/ENTER
```

Your response may be one of the following:

Hit END/ENTER defines all phases to be traced.

phase name defines the phase to be traced.

Figure 9-8 is an example of a prompting sequence for a program load trace request.

The SDAID now prompts you for the definition of the AREA. See

=======> "AREA Definition" on page 9-31.

```
trace □
4C08D SPECIFY TRACE TYPE +
pgmload □             <--------------------- your response
4C08D SPECIFY PGMLOAD TRACE OPTIONS.+
hdl □                 <--------------------- your response
4C08D SPECIFY PGMLOAD TRACE OPTIONS.+
□                     <--------------------- your response    (A)
4C08D SPECIFY NAME OF PHASE TO BE MONITORED OR PRESS END/ENTER.+
myphase □             <--------------------- your response    (B)


(A) Indicates that no further control information of this type
    is to be entered.
(B) Restricts the trace to the loading of the named phase.
```

Figure 9-8. Prompting for a PGMLoad Request

Register Alteration Trace

```
trace
4C08D SPECIFY TRACE TYPE.+
register    <------------- Your response
```

SDAID then prompts you for additional control information as follows:

```
4C08D SPECIFY REGISTER NUMBER(S).+
```

Your response may be the following:

One or up to eight register numbers traces the alteration of the contents of the specified registers. If you specify two or more register numbers, these numbers must be separated by one or more blanks, or by a comma with or without one or more blanks.

Sample responses:

```
9
8 9,a, f
```

SDAID then prompts you for a further limitation of the trace operation in the following way:

```
4C08D SPECIFY REGISTER PATTERN.+
```

Your response to this prompting message may be one of the following:

Hit END/ENTER to write an event record for any change of the contents of the specified register(s).

pp to write an event record whenever the contents of the named register(s) are changed to the hexadecimal value specified.

pp nn to write an event record whenever the contents of the named register(s) are changed to a value that is equal to or within the limits of the two given hexadecimal values.

The two values must be separated by one or more blanks, or by a comma with or without one or more blanks.

Sample pattern specifications:

```
3e8      (the same as 000003e8)
0 7d0    (the same as 00000000, 000007d0)
```

The SDAID now prompts you for the definition of the AREA. See

=======> "AREA Definition" on page 9-31.

SIO Instruction Trace

```
trace
4C08D SPECIFY TRACE TYPE.+
sio          <------------- Your response
```

SDAID then prompts you for the definition of the partition id as shown below:

```
4C08D SPECIFY PARTITION ID.+
```

You may select a certain partition, defined in the form of BG or F3 for example or the supervisor defined as SUPvr.

If you press END/ENTER all tasks of the system are traced.

SDAID then prompts you for the definition of the specific I/O channel(s), control unit(s), or unit(s), as described under

=======> "I/O Definition" on page 9-35.

STorage Alteration Trace

```
trace
4C08D SPECIFY TRACE TYPE.+
storage      <------------- Your response
```

SDAID then prompts you for additional control information as follows:

```
4C08D SPECIFY STORAGE PATTERN.+
```

Your response to this prompting message may be either of the following:

Hit END/ENTER: requests an event record to be written whenever storage within the trace range is altered.

hexvalue: requests an event record to be written whenever storage within the trace range is set to the specified value (any hexadecimal value of up to four bytes). If you specify an odd number of digits, a zero is inserted to the left of the first specified hexadecimal digit.

Note: This option traces only program-altered storage, not that altered by I/O operations.

The SDAID now prompts you for the definition of the AREA. Define ALL, if you want to get all tasks of your system watched. For more information see

=======> "AREA Definition" on page 9-31.

SVC (Supervisor Call) Trace

```
trace
4C08D SPECIFY TRACE TYPE.+
svc         <------------- Your response
```

SDAID then prompts you for additional control information as follows:

```
4C08D SPECIFY SVC NUMBER(S) OR '*'.+
```

Your response to this prompting message may be either of the following:

one or up to 16 SVCs: Specify the SVC number in hexadecimal notation. If you specify two or more SVC numbers, they must be separated by one or more blanks, or by a comma with or without blanks.

asterisk (*): defines all SVC instructions to be traced.

Sample SVC specifications:

```
02 9,A 26
25
*
```

SDAID then prompts you for the definition of the AREA. See

=======> "AREA Definition" on page 9-31.

VTAMBU Trace

```
trace
4C08D SPECIFY TRACE TYPE.+
vtambu      <------------- Your response
```

The SDAID prompts you for the OUTPut definition. See

=======> "Additional Output Definition" on page 9-36.

VTAMIO (VTAM I/O) Trace

```
trace
4C08D SPECIFY TRACE TYPE.+
vtamio      <------------- Your response
```

SDAID then prompts you for the definition of the partition id as shown below:

```
4C08D SPECIFY PARTITION ID.+
```

Specify the partition where VTAM is running, F3 for example.

SDAID then prompts you next for the definition of the specific I/O channel(s), control unit(s), or unit(s), as described under:

=======> "I/O Definition" on page 9-35.

AREA DEFINITION

This section describes the SDAID promptings and gives some examples of the possible replies to the ARea Definition. More detailed information about the ARea definition and the corresponding storage region definitions is given in the following sections:

- "Define the Tasks to be Traced: AREA Definition" on page 6-24.
- "Define the Storage to be Traced: OFFSET, ADDRESS, PHASE" on page 6-26.

SDAID prompts you for the required area definition by displaying the message:

```
4C08D SPECIFY TRACE AREA.+
```

Enter one of the following responses:

partition-ID
SUPvr
space ID
ALL

Prompts for Response Partition-ID

If you enter a partition-id (for example BG) to the above message, SDAID prompts you for the definition of the storage area to be traced as follows:

```
4C08D SPECIFY TYPE OF LIMITS.+
```

Your response to this prompting message is one of the following:

END/ENTER
OFFset
PHase
ADDress

Hit END/ENTER to trace the requested events within the storage occupied by the defined partition.

OFFset SDAID prompts you for the actual offset values:

```
4C08D  SPECIFY OFFSET RANGE.+
```

Your response to this message, a pair of offsets, is discussed below, under "PHase."

PHase SDAID prompts for the phase name as follows:

```
4C08D  SPECIFY NAME OF PHASE
```

Then SDAID prompts you for further limitation of the trace range:

```
4C08D  SPECIFY OFFSET RANGE.+
```

Your response to this message (for either partition or phase offset) is one of the following:

END/ENTER
reladdr1:reladdr2
reladdr1:*

Hit END/ENTER to trace the events in the entire area allocated to the specified partition supervisor or phase.

reladdr1:reladdr2 are defined in hexadecimal notation.

ADDress SDAID prompts you for the actual address range:

```
4C08D  SPECIFY ADDRESS RANGE.+
```

Your response to this message is either of the following:

END/ENTER
addr1:addr2
addr1:*

Hit END/ENTER to trace the events of the tasks with the defined partition-id without address limitation.

addr1:addr2 to define a certain address area within the storage allocated to VSE/Advanced Functions.

After you defined the address range, SDAID prompts you for the definition of additional trace areas as follows:

```
4C08D SPECIFY ADDITIONAL TRACE AREA.+
```

Your response to this prompting message is one of the following:

Hit END/ENTER indicates that you do not want any additional trace area.

LTA to trace also in the logical transient area.

SVA to trace also in the shared virtual area.

You enter one definition after the other via prompts. SDAID prompts you for the additional trace areas as long as it does not receive an END/ENTER without a definition.

Note: Please try to define the trace area as small and exact as possible. If you define too large an area the performance may be degraded. The definition of LTA and SVA normally leads to very large tracing intervals. For more details about the performance situation please refer to "Performance Degradation if LTA and/or SVA is Defined" on page 5-9.

Prompts for Response SUPvr

SDAID prompts you for additional control information as follows:

```
4C08D SPECIFY LIMITS OF SUPERVISOR AREA.+
```

Your response to this prompting message is one of the following:

Hit END/ENTER to trace the requested events within the entire storage area occupied by the supervisor.

OFFset
ADDress
PTA

If you respond with **OFFset**, SDAID prompts you for the actual offset range:

```
4C08D  SPECIFY OFFSET RANGE.+
```

Your response to this message one of the following:

```
END/ENTER
reladdr1:reladdr2
reladdr1:*
```

Hit END/ENTER to have the trace active in the whole storage defined to the supervisor (same as 0:* defined).

If you respond with **ADDress**, SDAID prompts you for the actual address range:

```
4C08D  SPECIFY ADDRESS RANGE.+
```

Your response to this message is one of the following:

END/ENTER
addr1:addr2
addr1:*

Hit END/ENTER to have the trace active in the whole storage defined to VSE/Advanced Functions (same as 0:* defined).

Prompts for Response Space-ID or ALL

SDAID prompts you for additional control information as follows:

```
4C08D SPECIFY ADDRESS RANGE.+
```

Your response to this prompting message is either of the following:

END/ENTER
address1:address2

Hit END/ENTER to trace the requested events within the defined address space (same as 0:* defined).

I/O DEFINITION

The I/O definition limits the range of an **IO, SIO, or VTAMIO** trace to one or more channels, to one or more control units, or to one or more devices. This section describes the SDAID promptings. Detailed information about the I/O definitions is provided under "Define the Traced I/O Devices" on page 6-46.

SDAID prompts you for the definition as follows:

```
4C08D SPECIFY KEYWORD UNIT OR CU OR CHANNEL.+
```

Your response to this prompting message may be one of the following: (detailed descriptions follow)

END/ENTER
CHannel
CU
UNit

Hit END/ENTER to define all I/O devices.

CHannel The program prompts you for one or up to 16 channel addresses as follows:

```
4C08D SPECIFY CHANNEL ADDRESS(ES).+
```

If you specify more than one address, separate them by one or more blanks, or by a comma with one or more blanks or without a blank.

Sample channel address specifications:

```
1
0 2,  3
```

CU SDAID prompts you for the hexadecimal definition of up to 16 control unit addresses as follows:

```
4C08D SPECIFY CONTROL UNIT ADDRESS(ES).+
```

If you specify more than one address, separate them by one or more blanks, or by a comma with one or more blanks or without a blank.

Sample control-unit address list specifications:

```
1, 2a 3f
1c
2    (same as 02)
```

UNit SDAID prompts you for the hexadecimal specification of up to 8 unit addresses as follows:

```
4C08D SPECIFY UNIT ADDRESS(ES).+
```

If you specify more than one address, separate them by one or more blanks, or by a comma with one or more blanks or without a blank.

If you specify a 1-digit device address, SDAID assumes channel 0 and control unit 0; for a 2-digit device address, SDAID assumes channel 0.

Sample device-address list specifications:

```
003, e 181 281
282
e      (same as 00e)
0e     (same as 00e)
```

ADDITIONAL OUTPUT DEFINITION

This section gives information on the various responses to the SDAID promptings. If you want more detailed information on the output definitions, refer to "Define Additional Trace Output" on page 6-32.

SDAID prompts you to specify additional output in the following way:

```
4C08D SPECIFY OUTPUT.+
```

You may respond with one or more of the following definitions:

BUffer,CCB,CCW,CCWD,COMReg,CREG,DUMP,FReg,GReg,IOTab,
LOwcore,LTA,PIB,PTA, PTAB,Supvr,SYSCom,TOD,TTAB

If you specify more than one definition, separate them by one or more blanks, or by a comma with one or more blanks or without a blank.

END/ENTER does not print any control block or register contents.

CCWD If you define CCWD, SDAID prompts you for the required control information as follows:

```
4C08D SPECIFY LENGTH OF DATA
```

Enter a decimal number between 32 and 65535.

DUMP If you define DUMP for output, SDAID prompts you for the required control information as follows:

```
4C08D SPECIFY DUMP AREA.+
```

You can answer with one of the following:

- partition-id
- PHase
- ADDress
- REG
- PTR

If you define **partition-id**, SDAID prompts you for additional definitions.

```
4C08D SPECIFY OFFSET RANGE.+
```

Respond to this message by one of the following:

- Hit END/ENTER
- reladdr1:reladdr2

If you define **PHase**, SDAID prompts you for additional definitions.

```
4C08D SPECIFY OFFSET RANGE.+
```

Respond to this message by one of the following:

- Hit END/ENTER
- reladdr1:reladdr2

If you define **ADDress**, SDAID prompts you for the address range as follows:

```
4C08D SPECIFY ADDRESS RANGE.+
```

- Hit END/ENTER
- addr1:addr2

If you define **REG**, the following prompting message is issued by SDAID:

```
4C08D SPECIFY DUMP RANGE.+
```

Respond to this message with:

- reg:bytes

If you define **PTR**, the SDAID prompts you for the required control information as follows:

```
4C08D SPECIFY REGISTER AND OFFSET.+
```

Respond to this message with:

- reg:offset

If you defined the register number and the offset (both hexadecimal) the following message is written by SDAID.

```
4C08D SPECIFY OFFSET AND DUMP LENGTH.+
```

Respond to this message with:

- offset:length

The offsets and the dump length are specified in hexadecimal notation.

OPTION DEFINITION

This section gives information on the various responses to the SDAID promptings. If you want more detailed information on the option definitions, refer to "Define Trace Options" on page 6-45.

SDAID prompts you for an option definition as follows:

```
4C08D SPECIFY OPTIONS.+
```

You respond with one of the following:

- **HALT**
- **TERMination** Specification
- **NOJCL** Specification
- **OCCurrence** Definition

If you define OCCurrence, SDAID prompts you as follows:

```
4C08D SPECIFY OCCURRENCE RANGE.+
```

Respond with:

END/ENTER

to indicate that you want all occurrences of the specified event to be traced (same as if you defined 1:*).

value1:value2

to limit tracing. See the examples below.

Sample occurrence definitions:

```
1:1     trace only the first occurrence
        of the event

1:*     trace all occurrences of the
        event (this is the default value)

5:12    trace selected occurrences
        (5 to 12) of the specified event
```

CHAPTER 10. START/STOP AND END THE TRACE

```
                    ┌────────────┐
                    │            │
                    │   SDAID    │
                    │            │
                    │            │
                    │  Overview  │
                    └────────────┘
                      Chapter 5

                    ┌────────────┐
                    │            │
                    │   SDAID    │
                    │            │
                    │  General   │
                    │Description │
                    └────────────┘
                      Chapter 6

 ┌────────────┐     ┌────────────┐     ┌────────────┐
 │ Initialize │     │ Initialize │     │ Initialize │
 │  an SDAID  │     │  an SDAID  │     │  an SDAID  │
 │  Trace in  │     │   Trace    │     │  Trace in  │
 │Direct Input│     │    via     │     │Prompt Input│
 │    Mode    │     │ Procedures │     │    Mode    │
 └────────────┘     └────────────┘     └────────────┘
   Chapter 7          Chapter 8          Chapter 9

                    ┌────────────┐
                    │ Start/Stop │
                    │    End     │
                    │            │
                   >│ the Trace  │<
                   >│   via AR   │<
                    └────────────┘
                      Chapter 10
```

This chapter describes how you can start, stop, or terminate an initialized SDAID trace.

Contents

THE REQUIRED COMMANDS

STRTSD/STOPSD Commands: Starting and Stopping

Once you have entered the READY command which ends the initialization process, you can activate the trace at once or later. To start or restart the trace operation, enter the command

```
STRTSD
```

without any operand.

> **Note:** If the trace was stopped by an event itself (TERMinate specified with the TRACE command), the trace operation can be restarted by issuing the STOPSD command followed by the STRTSD command.

To interrupt the trace operation with the restart capability retained, enter the command

```
STOPSD
```

without any operand.

ENDSD Command: Ending Execution

You can end the SDAID session by issuing the command:

```
ENDSD
```

without any operands. The ENDSD command releases all resources that were used by the program during the session, including the storage space that was occupied by SDAID and closes the trace output device. You may enter this command at any time during a session.

Attention Routine Command Example

The example in Figure 10-1 on page 10-4 shows how an initialized SDAID trace is started, interrupted and ended. After the ENDSD command has been processed all of the initialized trace information is released.

```
 •
strtsd □
4C05I PROCESSING OF 'STRTSD' COMMAND SUCCESSFUL
 •
 •
 •
stopsd □
4C05I PROCESSING OF 'STOPSD'  COMMAND SUCCESSFUL
 •
 •
 •
strtsd □
4C05I PROCESSING OF 'STRTSD' COMMAND SUCCESSFUL
 •
 •
 •
endsd □
4C05I PROCESSING OF 'ENDSD'  COMMAND SUCCESSFUL
```

Figure 10-1. Attention Routine Commands to Start, Stop and End the Trace

HOW TO CONTROL THE TRACE UNDER EXCEPTIONAL CONDITIONS

This section covers information on these situations where the normal way of directing the trace is not possible, since the attention routine is locked for some reason.

Tracing an Unintended Loop

Perform the following steps to use the SDAID traces: branch, instruction, storage or register alteration for information gathering about an unintended loop:

1. Initialize one of the trace types mentioned above in the normal way.

2. Start the trace with the STRTSD command.

3. Display the contents of control register 9 with the control processors alter/display feature.

4. Notice the contents of this control register for later use.

5. Set bits 0 through 3 to zeros with the alter display feature which stops the trace.

6. Recreate the loop condition by submitting the same job mix that existed when the particular loop occurred the first time.

7. When the loop appears again, restart SDAID operation by setting those bits of control register 9 to a value of 1 which you have set to zeros before.

Bit	Effect if set to 1
0	Successful branches are traced.
1	Instruction executions are traced.
2	Storage alterations are traced.
3	General register alterations are traced.

(See "Hardware Alter/Display" on page E-6 for information on how to use the Alter/Display feature.)

Control register A contains the start and control register B the end address of the trace. You may change this address range by varying the addresses stored in control registers A and B.

To resume operation after SDAID has collected sufficient information about the loop, and if you cannot exit from the loop, re-IPL VSE.

How to Terminate the SDAID Program under Exceptional Conditions

It may happen that you can no longer request the attention routine to gain control of your processor. At that point, SDAID operation cannot be stopped as usual by entering the command STOPSD. Instead you can perform the following steps:

If the trace type is INST, BR, REG, or STORAGE, you can use the following method:

1. Change your processor's mode of operation to manual.

2. Alter bits 0 through 3 of control register 9 to zero. (using the alter/display feature)

3. Let SDAID finish execution by changing your processor's mode of operation back to normal.

For more information on the values to be set into the control registers used by SDAID, consult the <u>Principles of Operation</u> manual for your processor.

If OUTDEV is a printer, the following method to stop the trace is possible:

1. Stop the printer device.
2. Wait until the system goes into the wait state.
3. Press the external interrupt key twice to stop the trace output.
4. Stop the trace with the STOPSD command.

HOW TO START/TERMINATE TRACING IN A SYSTEM WAIT CONDITION

In some cases SDAID forces a system wait condition. How you can restart the system by starting or terminating the trace options is the subject of this section.

Wait Due to OPTION=HALT

You may define that the system enters the wait state at occurrence of a specific event. This is accomplished by the option 'OPTION=HALT' defined together with the desired event.

How to Get out of this Wait State

When the system has entered the wait the address part of the wait PSW contains the value x'00EEEE'. The following actions may be taken to get out of the wait state:

1. If you want to continue tracing:

   Press the external interrupt key once. The system will enter the wait state again on the occurrence of the next traced event.

2. If you want to continue tracing but without OPTION=HALT:

   Enter X'FF' in storage location zero,
   Press the external interrupt key.

That removes the OPTION=HALT specification. The system continues tracing but does not enter the wait state on the occurrence of the specified event again.

System Wait Due to an Error Condition at the Output Device

SDAID loads a wait PSW with the value of X'EEEEEE' in the address part in each case the output device enters an exceptional condition.

The reason for the wait state is indicated in bytes 0-1 of the low address storage.

The device address of the SDAID output device which caused the wait state is stored in low-core bytes 2-3.

SDAID Output Device Error Codes

The following error codes may be found in storage location 0-1.

x'62C1'	End of tape condition on output tape
x'62C2'	Device not operational
x'62C3'	Missing device end interrupt
x'62C4'	Control unit busy
x'62C5'	Intervention required -device not ready
x'62C6'	Channel error
x'62C7'	Busout check
x'62C8'	UCS Parity|data converter check
x'62C9'	Sense missing or wrong
x'62D1'	Attention
x'62D2'	Command reject
x'62D3'	Open error
x'62D4'	Invalid I/O function requested
x'62D5'	Stack module terminate
x'62D6'	No print buffer available
x'62D7'	Max.number of write retries exceeded
x'62D8'	Max.number of SIO retries exceeded
x'62D9'	Unusual command sequence

How to Get out of a System Wait

At this point you may start or stop the trace operation. If the error condition has been removed (for example the printer or tape device is ready again), start the trace operation by **pressing the external interrupt key once.**

You may stop the trace data collection by **pressing the external interrupt key twice**.

Note that, if you stop the trace data collection, the VSE/Advanced Functions system is not reset to its previous state. You should issue the STOPSD and/or ENDSD command to stop or terminate the SDAID execution regularly.

Info/Analysis, illustrated in Figure 10-2, is a component of VSE/Advanced Functions Version 2 Release 1. It is a tool for:

- Dump file management
- Problem source identification
- Problem analysis

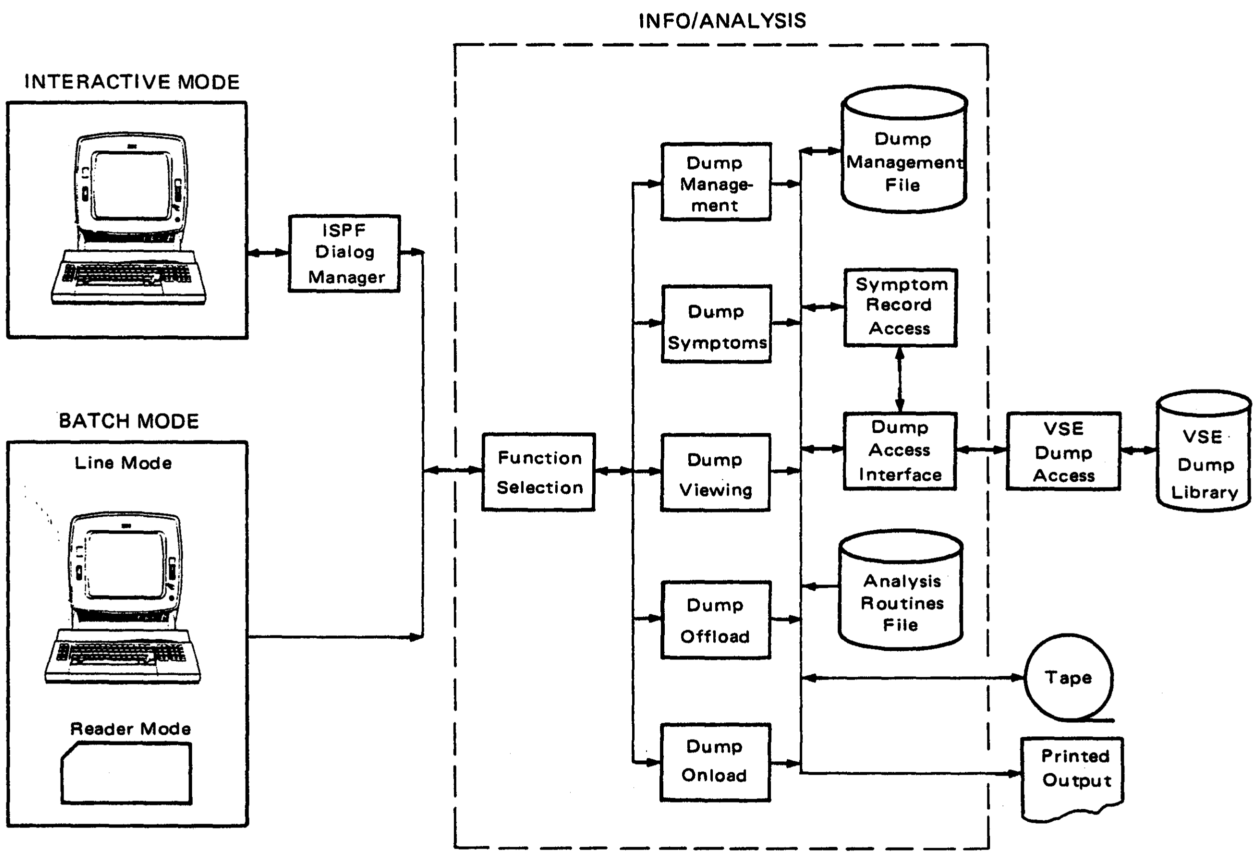


Figure 10-2. Information/Analysis Environment

That's the Contents of this Part

CHAPTER 11. INTRODUCTION

With Info/Analysis, you can simplify the task of using dump data to solve software problems. Info/Analysis assists you in this task through the following functions:

- Dump Management - to list the dumps being managed by Info/Analysis, to add or delete dumps from that list, and to delete dumps from the system.

- Dump Symptoms - to display problem failure information collected by the dumping component and by subsequent analysis routines.

- Dump Viewing - to display dump data in hexadecimal and character format, to locate, format, and display control blocks and other dump data that may be pertinent to the problem, to invoke dump analysis routines, and to display the results of those routines.

- Dump Offload - to copy a dump to tape for later retrieval.

- Dump Onload - to copy a dump to a dump sublibrary.

As shown in Figure 10-2, you may use Info/Analysis in either interactive or batch mode. All major functions are available in both modes; however, Dump Viewing has more flexibility in interactive mode while it is preferable to use Dump Offload and Dump Onload in batch mode.

In this part, the term control block is used to represent both the control blocks and the data areas in a dump.

The table in Figure 11-1 is a summary of the various Info/Analysis functions. It shows page references to the description of the functions for use in interactive and in batch mode.

The page references to the descriptions are given under the following headings:

Inter Function described for use in interactive mode.

Batch Function described for use in batch mode.

Function	Inter	Batch
Dump Management	13-1	19-8
Dump Symptoms	14-1	19-10
Dump Viewing	15-1	19-11
Dump Offload	16-3	19-13
Dump Onload	16-6	19-16

Figure 11-1. Info/Analysis Functions Summary

OPERATING ENVIRONMENT

Info/Analysis is a component of VSE/Advanced Functions that runs in a VSE/Advanced Functions or VSE/Interactive Computing Control Facility (VSE/ICCF) partition. Info/Analysis may be used in two modes, interactive and batch; both are described below.

Info/Analysis processes storage dumps that result from errors within the system, subsystems or user programs running on the system. The dumps are created by system dump and Stand-alone dump programs. Info/Analysis does not directly access the dump data. Rather, it uses system facilities to retrieve and update dump data and the symptom record. The symptom record is a collection of problem-related information stored in the dump and its extensions.

Info/Analysis uses a dump management file to maintain information about dumps. A dump must be identified in this file before it can be processed by Info/Analysis. This file is maintained using the Dump Management function.

Info/Analysis also uses an external routines file. This file contains a list of analysis routines that you may invoke to process dump data. The file also identifies user exit routines and dump access routines called by Info/Analysis.

Interactive Mode

In interactive mode, Info/Analysis runs in a VSE/ICCF interactive partition as a dialog managed by the Interactive System Productivity Facility (ISPF) Program Product.

You communicate with Info/Analysis through a 3270-type display terminal. By responding to a sequence of panels, you may select functions and options, enter commands, and display data. All printed output is routed to the ISPF list data set.

Interactive use of Info/Analysis is described in

=====> Chapter 12, Interacting with Info/Analysis.

The descriptions of Info/Analysis functions are also oriented toward the interactive environment in the following chapters:

=====> Chapter 13, Dump Management

=====> Chapter 14, Dump Symptoms

=====> Chapter 15, Dump Viewing

=====> Chapter 16, Dump Offload and Onload

Batch Mode

In batch mode, Info/Analysis runs in a VSE/Advanced Functions partition or in a VSE/ICCF interactive partition. You may enter control statements in two modes:

- Line mode - from the operator console or ICCF console
- Reader mode - from the system input device or the ICCF virtual reader defined as the input area

From a VSE/Advanced Functions partition, all output of batch operations is routed to the SYSLST device assigned to the partition. In line mode, messages are sent to the console as well as to SYSLST.

From a VSE/ICCF interactive partition, all output of batch operations is routed to the ICCF virtual printer called the print area. In line mode, messages are sent to the console as well as to the print area.

Batch use of Info/Analysis is described in

=====> Chapter 19, Info/Analysis in Batch Mode

The descriptions of Info/Analysis functions in Chapters 14 to 18 are oriented toward the interactive environment; however, the batch user should also read those chapters to become familiar with Info/Analysis capabilities.

You can find examples for most of the batch statements in

=====> Chapter 3, Printing the Stored Dump.

FUNCTIONAL OVERVIEW

When a dump is created, you can use it to solve a problem by taking actions that range from printing the dump symptoms to analyzing the dump in detail. The actions that you take depend on local procedures for dealing with dumps and your own techniques of dump analysis. Info/Analysis is a tool that can be used to enhance these procedures and techniques.

This section presents the stages of a dump's life cycle from problem occurrence to resolution. The ways in which you can use Info/Analysis at each stage are briefly presented.

This description is written for the interactive user. Similar facilities are available to the batch user. If you are a batch user, your Dump Viewing functions are limited to invoking analysis routines and printing either all formatted dump data and/or specific dump areas in hexadecimal mode.

Collecting Problem Data

When a problem occurs during system operation, the detecting component captures the condition of the system in a dump. The component stores the dump in the dump sublibrary designated for the partition that failed. Sometimes, a system operator may detect a problem and use Stand-alone dump or other dumping facilities to create a dump of all part of the system. Stand-alone dumps are stored on tape.

In either case, a dump contains a copy of part or all of system storage, and possibly a symptom record. The symptom record is a collection of failure-related information gathered by the dumping component when the dump is taken or added later by dump analysis routines.

The symptom record may contain:

- A description of the operating environment at the time the problem occurred.

- Symptoms that provide clues to the problem's origins.

- Free-form text and hexadecimal information that may describe the problem.

- Entries that define the format and location of dump data that may be pertinent to the problem. These entries are used when data is displayed in formatted mode.

For further information about the symptom record, see Appendix C, "Symptom Record Overview."

Using Info/Analysis Functions

Once the dump is created, you may invoke Info/Analysis. Interactive use of Info/Analysis is preferable for most tasks so that you may look at the results of your requests on your terminal and use all the Dump Viewing capabilities.

Your first task is to specify the dump you wish to process. To do so, enter the dump name on the first Info/Analysis panel displayed or select **Dump Management** from that panel:

- If you enter the dump name on the first panel displayed, that dump is selected for processing.

- If you select Dump Management, Info/Analysis displays a list of the dumps identified in the dump management file. You may select a dump from this list.

The first time in each session that you take either of these actions, Info/Analysis searches for the names of any dumps in the dump library. If Info/Analysis locates dumps in the dump library that are not yet identified in the dump management file, it adds identifying information for them to the file. When kept up to date, the dump management file lists all of the dumps in your system, those you have offloaded to tape, and those you plan to onload.

In either case, if the dump you specify does not reside in the dump library (for instance, a Stand-alone dump on tape), you must copy the dump into the system before you can work with it. The **Dump Onload** function provides this capability. Due to the time required to onload a dump, Info/Analysis submits a batch job to do the onload. The interactive partition is then free to do other problem analysis while the onload is taking place. When the dump is onloaded, you may then use the interactive mode for dump examination and analysis.

By selecting **Dump Symptoms,** you may look at some of the contents of the symptom record. This record contains failure information such as a description of the environment at the time of the failure and any reason, abend, or error codes that were issued.

To examine the dump and determine other symptoms of the problem, you may use **Dump Viewing.** There is no fixed method for analyzing a dump. Your method should suit the current problem type, installation procedures, and your individual preferences. Dump Viewing can accommodate a variety of techniques:

- Analysis Summary -- You may look at a list of the items in the dump that might be related to the problem. The item can be problem-related text and hexadecimal data, descriptions of control block chains and linkages, and control block contents. You may select an item from this list for display.

- Dump Display Options -- You may define the format of dump display panels. In hexadecimal mode, dump data is presented in the traditional dump format. In formatted mode, field names and offsets may be correlated to the storage displayed for specific dump addresses.

- Dump Display -- You may examine any areas of the dump that interest you.

- Analysis Routines -- If you need more information about the problem, you may choose a dump analysis routine for execution. The routines are provided by the system components. A routine may access the dump, examine control block fields, verify chains of control blocks, and perform other tasks to isolate the problem areas. A routine may then display or print its results or add symptoms and comments to the symptom record. Just as important, a routine may add descriptions of control block chains and the structure of specific control blocks to the symptom record.

When you diagnose the problem and find a satisfactory solution, you may use **Dump Management** to delete the dump from the system and remove associated information from the dump management file. If the dump is on tape, the information is deleted from the dump management file. You are responsible for the positioning of the tape.

If diagnosis must be delayed, you may leave the dump in the dump library, or, if you prefer, you may select the **Dump Offload** function to copy the dump onto tape. Due to the time it takes to offload a dump, Info/Analysis submits a batch job to do the offload. This leaves the interactive partition to do other problem analysis. An offloaded dump may subsequently be onloaded for additional Info/Analysis sessions.

CHAPTER 12. INTERACTING WITH INFO/ANALYSIS

This chapter describes how to use Info/Analysis at a 3270-type display terminal.

After you familiarize yourself with the contents of this chapter, you are ready to conduct an Info/Analysis session.

Info/Analysis is dialog managed by the Interactive System Productivity Facility (ISPF) Program Product. Familiarity with ISPF and how it manages the dialogs that run under it is of value to you. For information about this relationship, see <u>ISPF Dialog Management Service.</u>

Contents

PANELS

You interact with Info/Analysis through a series of displays called panels. A panel is a predefined display image that contains a collection of data (selections, data entry fields, or data) that is presented on your terminal. Each panel requires a response which determines the next panel to be displayed or the function to be performed. The response may be a selection, data, or a command, and may be entered in uppercase or lowercase.

Panel Format

This publication shows the panels as they appear on most 3270 display terminals; that is, 24 lines of 80 characters each. Info/Analysis can reformat the width of scrollable panels to show as much data as possible on your terminal type. This includes, for example, displays that are 132 characters wide. Each of the Info/Analysis panels conforms to the general ISPF format shown in Figure 12-1.

```
line 1 | Panel Name                    Function Name
line 2 | ===>                                             SCROLL ===> PAGE
line 3 |
line 4 | DUMP NAME .... ____________________________________________
       |
       |    -----
       |      |
       |      |
       |      |    Panel Body
       |      |
       |      |      - Selection items
       |      |      - Data entry fields
       |      |      - Scrollable display data
       |      |
       |      |
       |      |
       |      |
       |      |
       |      |
       |      |
       |      |
line n |    -----
```

Figure 12-1. General Panel Format

The header line (line 1) includes the panel identifier or name and the name of the function to which the panel relates.

The entry line (line 2) contains an input field labeled with an arrow (===>). Use this field to enter selection numbers and commands. If the panel is scrollable, line 2 also contains a SCROLL field that defines the number of lines or columns to be scrolled when you enter a scroll command. The scroll command determines the direction of the scroll movement. The scroll amount shown in Figure 12-1 on page 12-3 is one page. See "Scrolling" on page 12-10 for a discussion of scroll amounts and how to change them.

The message line (line 3) contains any message issued by Info/Analysis. If more than one message is issued in one interaction, line 3 contains a message that instructs you to enter the MSG command to display a panel containing the messages. The MSG command is discussed in Chapter 17, "Interactive Commands" on page 17-1.

The DUMP NAME field (line 4) identifies the current dump; that is, the dump with which you are working. You can specify this dump name on either of two panels:

- The function selection panel, the first panel displayed when you invoke Info/Analysis.
- The dump management panel, displayed when you select Dump Management.

You may change the current dump only on these two panels.

The format of the remaining lines in the panel depends on the function of that panel. These lines may contain selection items, data entry fields, scrollable display data, or a combination of these.

Panel Types

While using Info/Analysis, you see three basic types of panels:

- Selection panels contain a list of numbered or unnumbered items. For numbered lists, you can select an item by typing its number on the entry line and pressing the ENTER key. If your terminal has a light pen, you may use it for selection by positioning the pen on the desired item's selection number and pressing the pen to the screen.

  For unnumbered lists, you select an item by positioning the cursor at the beginning of the line containing that item, typing a valid 'line command', and pressing the ENTER key. There is no light pen selection capability for unnumbered lists.

- Data entry panels contain a collection of labeled fields in which you can enter data. Data entry fields on any panel are indicated by an arrow (===>) following the label of the field. In some cases, information is pre-entered in the fields based on prior entries or Info/Analysis defaults. You may respond by filling in or changing the data.

When information in a field cannot be changed, the field label is separated from the data by a series of periods (....). The DUMP NAME field exemplifies this distinction. On the function selection panel, DUMP NAME is a data entry field. An arrow follows the field label because a dump name may be entered. On the dump symptoms panel, DUMP NAME is not a data entry field. Periods follow the field label because the displayed dump name may not be changed now.

- Scrollable display panels contain data that does not fit on one screen. You may not change any of the data displayed in the scrollable area of the panel.

Panels are presented in a logical order. When you invoke Info/Analysis, you see a panel from which you select a function. This panel may be followed by other selection panels if they are needed to further define the function. Data entry panels may then collect any required information. Finally, the results are shown on scrollable display panels. This logical series of panels is called the panel sequence.

INVOKING INFO/ANALYSIS IN INTERACTIVE MODE

Installation Steps

The following steps have to be performed before you can start Info/Analysis in interactive mode:

1. Install and Customize ISPF (refer to the ISPF Installation and Customization manual).

2. Tailor the start-up for ICCF and CICS. Items like the following have to be considered:

   - Recommended interactive partition size: 1.5 MB (896 KB if Info/Analysis is loaded into the SVA); use a unique partition class.
   - LIBDEF's for used libraries
   - Dynamic space allocation

3. Modify the ISPF start-up procedure (an example is shown in Figure 12-2 on page 12-6)

```
&&OPTIONS 00100010
/INPUT
&/LOAD ISPSTART
&/FILE NAME=ISPLOG,DISP=DELETE,SPACE=5
&/FILE NAME=ISPLIST,DISP=DELETE,SPACE=10
&/FILE NAME=ISPCTL1,DISP=DELETE,SPACE=2
&/FILE NAME=ISPCTL2,DISP=DELETE,SPACE=2
&/FILE NAME=ISPLST1,DISP=DELETE,SPACE=2
&/FILE NAME=ISPLST2,DISP=DELETE,SPACE=2
&/FILE NAME=ISPEDA,DISP=DELETE,SPACE=2
&/FILE NAME=ISPEDB,DISP=DELETE,SPACE=2
&/OPTION DUMP GETVIS=P-30 TIME=32767,65535
ISPF NOABEXIT PGM(INFOANAI) NEWAPPL(BLN) &&PARAM2 &&PARAM3 &&PARAM4
ISPDEF * SEARCH=(IJSYSRS.SYSLIB,PRD2.COMM)
ISPDEF ISPTLIB SEARCH=(PRD2.CETO,IJSYSRS.SYSLIB,PRD2.COMM)
ISPDEF ISPPROF SEARCH=PRD2.CETO,CATALOG=PRD2.CETO
ISPDEF ISPTABL CATALOG=PRD2.CETO
/END
/PEND
/RUN
```

PRD2.COMM .... the library where ISPF is installed.
PRD2.CETO .... the library containing the user profile.

Figure 12-2. Example: ISPF Start-up Procedure

For complete information, refer to the following manuals:

- Interactive System Productivity Facility Installation and Customization
- Interactive System Productivity Facility Dialog Management Services
- VSE/Interactive Computing and Control Facility Terminal User's Guide
- VSE/Advanced Functions System Management Guide

Start Info/Analysis

You initiate an Info/Analysis session by entering the name of the ISPF start-up procedure in ICCF command mode.

Alternatively, you may select Info/Analysis from the ISPF master menu if your installation has included Info/Analysis as an option on that menu.

When you issue the procedure name or select Info/Analysis from the menu, Info/Analysis looks at your user profile to establish default values for your session. See User Profiles.

When Info/Analysis is invoked, interactive recording is defaulted to "no". If you wish to start this recording, or if you subsequently wish to stop it, use the LOG command described in Chapter 17, "Interactive Commands" on page 17-1.

User Profiles

To eliminate repetitious data entry, Info/Analysis retains information for you in your user profile. The following items are saved from one session to another:

- Current dump name
- Dump display options
- PF key definitions
- Scroll amount

As a new user, you must enter the current dump name to work with a dump. From then on, Info/Analysis initializes the DUMP NAME field with the name you entered most recently. The name is saved at the end of each sessions in your user profile. You may verify the name as you perform tasks with Info/Analysis and change the current dump if you wish. There is no current dump when you begin your first Info/Analysis session nor is the dump name saved if you delete the current dump before you complete your session.

You may select the dump display options function of Dump Viewing so that you can set (as a new user) or change your preferences for the format of dump displays. Info/Analysis saves the most recent options in your user profile. Until you set the options, Info/Analysis assumes that you prefer the defaults. See under "Specifying Dump Display Options" on page 15-4 for a discussion of the defaults.

Default PF key definitions are established by ISPF and saved in your user profile. You may change the defaults using ISPF facilities described in ISPF Dialog Management Services.

A default scroll amount of PAGE is used unless you set a new default. You can set a new default as described under "Scrolling" on page 12-10.

Function Selection Panel

The function selection panel shown in Figure 12-3 is displayed when you invoke Info/Analysis. It is the focal point for all Info/Analysis activities.

```
BLNFS001          INFO/ANALYSIS FUNCTION SELECTION
===> _

DUMP NAME ===> ____________________________________________

Type the dump name in the field above, or select a function
by typing its number, and press ENTER:


1  DUMP MANAGEMENT     Select, add, or delete a dump

2  DUMP SYMPTOMS       Display dump symptoms

3  DUMP VIEWING        Examine a dump

4  DUMP OFFLOAD        Copy a dump to tape

5  DUMP ONLOAD         Load a dump from tape

T  TUTORIAL            Learn how to use Info/Analysis

X  EXIT                End the Info/Analysis session
```

Figure 12-3. Function Selection Panel

Prior to selecting a function from this panel, you may enter the name of the dump you wish to work with in the DUMP NAME field. If a dump name is stored in your user profile, the DUMP NAME field on the function selection panel is initialized with the stored name. You may change this name now if you wish to work with a different dump, or when the dump management panel is displayed. For details about specifying the current dump, see Chapter 13, "Dump Management" on page 13-1.

To indicate the function you wish to perform, you may enter the associated selection number on the entry line of the panel. Your selection causes a panel appropriate to that function to be displayed.

The functions you may select are:

- Dump Management - to list the dumps being managed by Info/Analysis, to add or delete dumps from that list, and to delete dumps from the dump library (See Chapter 13, "Dump Management" on page 13-1 for a description).

- Dump Symptoms - to display problem failure information collected by the dumping component and by subsequent analysis routines (See Chapter 14, "Dump Symptoms" on page 14-1 for a description).

- Dump Viewing - to display dump data in hexadecimal format, to locate, format, and display control blocks and other dump data that may be pertinent to the problem, to mask out (overlay) data in specific areas of the dump, to invoke dump analysis routines, and to display the results of those routines (See Chapter 15, "Dump Viewing" on page 15-1 for a description).

- Dump Offload - to copy or move a dump to tape for later retrieval (See "Offloading a Dump" on page 16-3 for a description).

- Dump Onload - to copy an offloaded dump, a dump from another system, or a Stand-alone dump from tape to the dump library (See "Onloading a Dump" on page 16-6 for a description).

- Tutorial - to display panels that describe how to use Info/Analysis (See "Online Help Information" on page 12-15 for a description).

- Exit - to immediately end the Info/Analysis session (See "Ending the Info/Analysis Session" on page 12-18 for a description).

When you have completed a function, Info/Analysis redisplays the function selection panel so that you can choose another function or conclude your session.

In this publication, the point in the panel sequence at which the function selection panel is displayed is called the selection level. Once you have selected a function from this panel, you are at the function level (Dump Management, Dump Symptoms, etc.).

CHAINING RESPONSES

Once you are familiar with the panel sequence and the option numbers on the selection panels, you may streamline your use of Info/Analysis by using response chaining. These techniques enable you to bypass selection panels and directly display the functional panel that is of interest to you.

The panel sequence and the option numbers are:

1. Dump Management
2. Dump Symptoms
3. Dump Viewing
   1. Dump Display Options
   2. Dump Display
   3. Analysis Summary
   4. Analysis Routines
4. Dump Offload
5. Dump Onload

Response Chaining

Response chaining enables you to go directly from the current function to any other function in the selection sequence without returning to the function selection panel. Your string of responses is processed starting from the current panel.

This type of chain may include commands as well as selection numbers. Response chaining requires a special delimiter; the default is a semicolon (;).

For example, suppose the function selection panel is displayed. You wish to print the symptoms of the current dump and go on to dump display. Enter the following chain:

```
2;print data;end;3;2
```

Info/Analysis responds by displaying the dump display panel. For another example, suppose you were on the dump symptoms panel. You wish to display the analysis summary panel for the current dump. Enter the following chain:

```
RETURN;3;3
```

Info/Analysis responds by displaying the analysis summary panel. You may change the response chaining delimiter using the ISPF parms option (see ISPF Dialog Management Services, for more information).

SCROLLING

Sometimes, an action may result in more data than fits on one screen. Scrolling commands are available to move the screen "window" across the information.

You may use two ISPF commands for scrolling (UP and DOWN). You may type the command on the entry line or press the PF key that represents that command. The two commands are described in Chapter 17, "Interactive Commands" on page 17-1.

Whenever scrolling is allowed, a scroll amount is displayed in the SCROLL field on line 2 of the panel. This amount determines the number of lines scrolled with each use of a scroll command. The command (or PF key) determines the direction of the scrolling action.

The first time that a scrollable panel is displayed, the SCROLL amount is PAGE. This means that any scroll request causes the data to be moved by a full screen if another full screen of data is available. For example, if you scroll down by one page, the line following the last line displayed when you request the scroll action is the first line of data in the new display.

To change the scroll amount, move the cursor to the SCROLL field of any scrollable panel and overtype the displayed amount. This change remains in effect until you change the SCROLL amount again. Valid scroll amounts are:

- A number from 1 to 9999 - specifies the number of lines (up or down) to be scrolled.

- PAGE - specifies scrolling by one page, the amount of information currently visible on the screen.

- HALF - specifies scrolling by a half page, half the amount of information currently visible on the screen.

- MAX - specifies scrolling to the top, bottom, left margin, or right margin of the displayed item, depending on which scrolling command is used.

- CSR - specifies scrolling based on the current position of the cursor. The line indicated by the cursor is moved to the top or bottom of the screen, depending on which scrolling command is used. If the cursor is not in the scrollable area of the panel, or if it is already positioned at the top or bottom, a PAGE scroll occurs.

To reduce keystrokes when changing the scroll amount, you need to overtype only the first character as follows:

- To change the scroll amount to PAGE, HALF, MAX, or CSR, overtype the first character with 'p', 'h', 'm', or 'c', respectively.

- To change the scroll amount to a number of lines, overtype the first character(s) with the desired number. Any alphabetic characters following the number are ignored. For example, "3AGE" is interpreted as "3".

If you do not wish to change the scroll amount for the session but wish to scroll a different amount on one particular scroll action, you may type any valid scroll amount on the entry line of the panel and use it with a scrolling command or PF key.

For example, you can enter:

```
===> UP 3
```

and press the ENTER key, or you may simply enter:

```
===> 3
```

and press the UP PF key. Either form results in a temporary, one-time override of the scroll amount.

TERMINAL KEYS

You may use the program access (PA) and program function (PF) keys to request common operations. PF keys are useful but are not required for Info/Analysis.

Program Access Keys

PA keys have system-defined meanings. They may not be redefined.

RESHOW (PA1) If you press PA1 at any point in an Info/Analysis session, it acts as a RESHOW key. PA1 may be useful if the user has pressed the ERASE INPUT or has unwanted information but has not yet pressed ENTER or a PF key.

CANCEL (PA2) If you press PA2 at any point in an Info/Analysis session, the session is terminated. This key should not be used while in Info/Analysis.

Program Function Keys

If you prefer and if your terminal has PF keys, you may set a PF key to contain an Info/Analysis command. Pressing the PF key has the same effect as typing the command on the command line and pressing the ENTER key. For example, instead of typing "help" on the command line and pressing the ENTER key, you may press the PF key defined with the "HELP" command. The use of PF keys is preferable because it reduces keystrokes and possible errors. The commands are described in Chapter 17, "Interactive Commands" on page 17-1.

You may modify the PF key definitions using ISPF facilities described in ISPF Dialog Management Services. The settings are stored in your user profile.

ENTER Key

To enter the information you have typed on the current panel, press the ENTER key. Any data you have typed is processed. From the entered data, Info/Analysis determines what operation to perform next.

If you press ENTER without typing any data, ENTER has no effect with the following exceptions:

- On the dump offload and onload panels, a null ENTER causes an offload or onload job to be submitted to background if the required panel fields are filled.

- On a tutorial panel, a null ENTER causes the display of either the next panel in sequential mode or the prior panel in selection mode. See "Online Help Information" on page 12-15 for details.

CLEAR Key

If you press the CLEAR key, a blank screen is displayed. To redisplay the previous panel, press PA1. Any data typed but not entered before CLEAR is pressed will be lost.

ORDER OF PROCESSING

COMMAND	FUNCTION PERFORMED
HELP	The tutorial panel is displayed.
MSG	The additional messages panel is displayed.
PRINT no operand	A copy of the screen is printed.
END RETURN	Info/Analysis returns to the previous logical panel or to the function selection panel.
LOG	The logging status is changed.
PRINT operand	The data is formatted and printed.
UP, DOWN	The data is scrolled in the specified direction. If in dump viewing, any line commands are processed first.
FIND	The data is searched for the specified string.
MASK	The specified address range is masked and the current data is re-formatted and displayed.

Figure 12-4. Input Processing

ONLINE HELP INFORMATION

Info/Analysis provides a set of panels called the tutorial which describes Info/Analysis, its panels, and its messages. There are three ways to enter the tutorial:

- Select the tutorial option (T) from the function selection panel.
- Enter the HELP command on any panel.
- Select a message on an additional messages panel.

Once you have entered the tutorial, you may use selections, the ENTER key, and commands (or PF keys) to view additional tutorial panels.

Selections on tutorial panels are made in the same way that numbered items are selected on other Info/Analysis panels, except that selection by light pen is not possible.

For the first time, before using other parts of the tutorial, select the "Tutorial Use" topic from the table of contents shown as the first tutorial panel. This section of the tutorial summarizes the techniques for using the tutorial.

Types of Online Help

Info/Analysis provides online help in three forms:

- General tutorial - a set of panels that introduces Info/Analysis including descriptions of the panels and the commands; it is displayed when you select the tutorial option (T) on the function selection panel. The first panel in this set is a table of contents. The structure of the general tutorial is shown in Figure 12-5 on page 12-16.

```
BLNTU001 - Table of Contents Selection Panel
    |
    |
    |——  BLNTUG00 - General Information
    |
    |——  BLNTUC00 - Command Descriptions
    |
    |——  BLNTUS00 - Function Selection
    |
    |——  BLNTU100 - Dump Management
    |
    |——  BLNTU200 - Dump Symptoms
    |
    |——  BLNTU700 - Dump Viewing
    |
    |——  BLNTU400 - Dump Offload
    |
    |——  BLNTU500 - Dump Onload
    |
    |——  BLNTUT00 - Tutorial Use
    |
    |——  BLNTUI00 - Index of Topics
```

Figure 12-5. Tutorial Hierarchy

- Message help - each panel describes the message for which help is requested. A message help panel is displayed when you enter the HELP command on a panel when a message is displayed on the message line or when you select a message from the additional messages panel. Message help provides an explanation of the message, the resultant system action, and the suggested response to it.

- Panel help - a panel that describes the panel you were viewing when you requested help. Panel help is displayed if you enter the HELP command on a panel when no message is displayed on the message line or if you press the ENTER key when you are viewing a message help panel.

  If you enter the HELP command on a tutorial panel, a panel that describes the tutorial is displayed. This panel is the tutorial help panel.

  If you enter the HELP command on a message help panel, a panel that describes message help is displayed. This panel is the message help tutorial panel.

Tutorial Commands

ISPF provides special commands to help you move through the tutorial. In addition, certain Info/Analysis commands are also appropriate within the tutorial. All of the commands that are valid within the tutorial are described in this section. Details about the Info/Analysis commands are provided in Chapter 17, "Interactive Commands" on page 17-1. Once you enter the tutorial, three of the PF keys assigned to the scrolling commands are reassigned to three ISPF commands that help you move through the tutorial. The commands are:

- BACK or B - to back up to the previously viewed panel.
- SKIP or S - to move to the next topic in the tutorial hierarchy.
- UP or U - to display a higher level list of topics.

Scrolling commands are not applicable within the tutorial.

Other commands that you may enter during the tutorial are:

- TOP or T - displays the table of contents of the tutorial. The table of contents presents the complete hierarchy of subjects within the tutorial. You may select topics from this list.
- INDEX or I - displays the tutorial index which provides random access to topics within the tutorial hierarchy. You may select topics from the index.
- END - ends the tutorial as described below in "Ending the Tutorial."
- HELP - displays a tutorial panel for the tutorial. This panel is called the tutorial help panel.
- RETURN - ends the tutorial and returns to the panel from which the tutorial was invoked.

BACK, SKIP, UP, TOP, and INDEX are applicable to the main tutorial only. They do not apply to message help.

On the message help tutorial panel and the tutorial help panel, pressing the ENTER key is equivalent to entering the END command.

Ending the Tutorial

There are three ways to leave the tutorial:

- Enter the END command from any panel as follows:
  - If entered from the tutorial help panel, END returns you to the help panel from which you requested help. Enter END again to leave the tutorial.
  - If entered from the message help tutorial panel, END returns you to the message help panel from which you requested help. Enter END again to leave the tutorial.
  - If entered from any other panel, you are returned to the panel from which you requested help or to the function selection panel, depending on how the tutorial was invoked.
- Enter the RETURN command on any tutorial panel. RETURN causes a return to the panel from which help was requested, or to the function selection panel.
- Enter the response chaining function on any tutorial panel. The Info/Analysis panel displayed is determined by the chain.

ENDING THE INFO/ANALYSIS SESSION

You can end an Info/Analysis session from any panel.

Two termination options are available from the function selection panel:

- Enter the END command. The END command is described in Chapter 17, "Interactive Commands" on page 17-1.
- Invoke the ISPF exit option by entering 'x' in the entry line.

If you are viewing a panel other than the function selection panel and you wish to end your session, enter 'Return;x' on the entry line. This response chaining function has the same result as entering 'x' from the function selection panel.

If the list files and the log files have not been used during the session, any of the termination methods described above cancels Info/Analysis and returns you directly to VSE/ICCF without displaying the ISPF termination panel.

If you have used the list or the log file and have set up defaults for list and log processing using the ISPF parms option, those defaults are used. Any of the termination options described above cancels Info/Analysis and returns you directly to VSE/ICCF without displaying the ISPF termination panel.

If you have used the list or the log files and have not specified defaults for list and log processing, the ISPF termination panel is displayed. Select the applicable options and subsequently exit to VSE/ICCF.

If Info/Analysis is processing and you need to cancel your session, you may press the PA2 key once to stop processing and return immediately to VSE/ICCF.

CHAPTER 13. DUMP MANAGEMENT

To process a dump, Info/Analysis requires that the dump reside in the system and that it be identified in the dump management file.

This chapter discusses the Info/Analysis Dump Management function with which you:

- Identify dumps to Info/Analysis

- Select dumps for processing

- Delete dumps from the dump library

The dump management file contains information about dumps being managed by Info/Analysis. Dump Management uses and maintains the contents of this file. The file should list all of the dumps in your system, those that have been offloaded to tape, and those you plan to load into the system.

You select Dump Management by entering '1' on the function selection panel. This causes the display of the dump management panel. All of the Dump Management tasks -- listing, selecting, adding, and deleting -- are requested from this panel.

Contents

LISTING MANAGED DUMPS

The first time in each Info/Analysis session that you select Dump Management or specify a dump name on the function selection panel, Info/Analysis responds by invoking an exit routine, BLNUSADM. This routine searches the dump library for dumps that are not yet identified in the dump management file. The routine adds the identification of each of these dumps, if any, to the file.

The Dump Management Panel

After this task is completed and at any other time that you select Dump Management during a session, Info/Analysis displays the dump management panel, shown in Figure 13-1, which lists all of the dumps identified in the dump management file. The dumps are listed in reverse chronological order by date and time stamp. That is, the most recent dump is at the top of the list.

```
BLNDM001                    DUMP MANAGEMENT
===> _                                          SCROLL ===> PAGE

DUMP NAME ===>

To select a dump, type 's' before its name or type its name above.
To add a dump, type its name above.
To delete a dump, type 'd' before its name.
Press ENTER:

         DUMP NAME              ONLINE   DATE/TIME TAKEN       VOLID
  SYSDUMP.F3.00000010              Y     84/04/12 08:44:51
  SYSDUMP.F3.DMPC001                        TO BE ONLOADED
  SYSDUMP.F6.00000007              Y     84/04/12 08:11:06
  SYSDUMP.F4.00000006              Y     84/03/28 22:05:16    T02818
  SYSDUMP.F4.00000003                    84/03/17 01:31:21    T42901
  SYSDUMP.F5.00000002                    84/03/08 15:38:42    T03496
```

Figure 13-1. Dump Management Panel

The panel lists the following information for each dump:

- DUMP NAME - the name of the dump.

- ONLINE - an indication of whether or not a copy of the dump resides in the dump library. 'Y' indicates that there is a copy. A blank indicates that there is no copy.

- DATE/TIME TAKEN - the date and time that the dump was created or an indication that the dump is to be onloaded. If the dump has no symptom record and thus no date and time stamp, the date and time the dump was identified to Info/Analysis is used.

- VOLID - the serial number of the tape, if any, on which a copy of the dump resides.

The second of the dumps listed in Figure 13-1 on page 13-3, SYSDUMP.F3.DMPC01, is listed as "TO BE ONLOADED". This indicates that the dump is not in any of the dump sublibraries and must be onloaded before it may be processed. If you select this dump, you may only:

- Select Dump Onload to load the dump into the dump library

- Remain in Dump Management to delete the dump from the list.

If you have previously specified the current dump during this session or if a dump name has been carried over from a prior session, the DUMP NAME field is primed with that name when the dump management panel is displayed.

ADDING A DUMP

To add a dump to the dump management file, enter the name in the DUMP NAME field of the function selection or dump management panel. You may need to add a dump if:

- You wish to process a dump and it was created after Info/Analysis last searched the dump library for new dumps.

- The dump is not in the dump library and you plan to onload it.

Once you enter a name, the dump is added to the dump management file and is selected for processing. Its name appears in the DUMP NAME field and in the list on the dump management panel.

Dump Name Conventions

If it is the first time that the dump is being introduced to the dump library, you must create a name. The name must follow system conventions and consist of a library name, a sublibrary name, and a dump identifier separated by periods (.) as follows:

- Library - a 1- to 7-character name in which the first character is alphabetic and the remaining characters are alphameric.

- Sublibrary - a 1- to 8-character name in which the first character is alphabetic and the remaining characters are alphameric.

- Dump identifier - a 1- to 8-character name in which the first character is alphabetic and the remaining characters are alphameric.

  Note: The alphabetic characters L, M, N, S, and X are reserved for system use only. Do not use one of this characters in the first dump identifier position.

The standard system dump library is SYSDUMP. The standard sublibraries are BG, F1 through F9, FA, and FB. When naming a dump, use the standard system library and sublibrary or any libraries and sublibraries defined at your installation. To define libraries and sublibraries, use the system librarian facilities. For a discussion of these facilities, see VSE/Advanced Functions System Control Statements.

The dump identifier assigned by the system usually consists of one alphabetic character followed by numeric characters. Thus a typical dump created by VSE/Advanced Functions could be named SYSDUMP.BG.00001111. The VSE/Advanced Functions system appends a member type of DUMP to this name. However, you need not use the type when referring to a dump in Info/Analysis.

SELECTING A DUMP

To work with a dump using Info/Analysis, you must select it. Any dump in the list may be selected by typing 's' in front of the dump name and pressing the ENTER key. If you type 's' in front of more than one dump name, the one closest to the end of the list is selected as current.

An alternative method of selecting a dump is to enter its name in the DUMP NAME field. If a dump name already appears in the field, you may overtype it. However, this method is intended primarily for adding a dump that is not in the list already.

The Current Dump

Once you have selected a dump for processing, it is referred to as the current dump. Its name appears in the DUMP NAME field of all panels. You may choose another dump during the session by returning to the dump management panel or the function selection panel and selecting or entering another dump name. The current dump is saved between sessions in your user profile.

To select and process a dump that is not yet identified to Info/Analysis, you must add it to the dump management file. See "Adding a Dump" on page 13-4.

DELETING A DUMP

The Dump Management delete function erases a dump if it resides in the dump sublibrary and erases associated information from the dump management file. After you delete a dump, Info/Analysis does not maintain any information about it.

You request dump deletions from the dump management panel. Position the cursor at the start of the line containing the name of the dump you wish to delete and type 'd'. You may request that any number of dumps be deleted. When you have typed 'd' for all the dumps you wish to delete, press the ENTER key. A panel is displayed on which you must confirm or cancel the deletion request for the first of the dumps you specified.

The Dump Delete Verification Panel

As indicated in the example in Figure 13-2, the panel instructs you to enter 'y' or 'n'.

```
BLNDMD01                DUMP DELETE VERIFICATION
===>


Verify that this dump is to be deleted:


         DUMP NAME                ONLINE   DATE/TIME TAKEN     VOLID

  SYSDUMP.F6.00000007               Y     84/04/12 08:44:51








      Type 'y' to delete or 'n' to keep, and press ENTER:

      DELETE ===> _
```

Figure 13-2. Dump Delete Verification Panel

If you confirm the request, Info/Analysis removes the dump identification entry from the dump management list. If the dump resides in a dump sublibrary, Info/Analysis also erases the dump. If a dump does not reside in the dump library, you should follow your installation's procedures for disposing of the dump tape.

When Info/Analysis completes the dump deletion, a verification panel for the next dump specified, if any, is displayed. A confirmation message for the prior deletion appears on the message line. If any error occurs during a deletion, the verification panel for the next dump is presented with an error message on the message line. You may continue the process of deleting the other dumps. When you have confirmed or canceled the deletion request for the last dump specified, the dump management panel is displayed containing the updated list.

It is possible to select the current dump and delete any number of dumps in one interaction. To do so, enter the desired "s" and "d"s and press the ENTER key. After the delete confirmation process is complete, the dump management panel is displayed with the updated list of dumps and the selected dump name in the DUMP NAME field.

CHAPTER 14. DUMP SYMPTOMS

The first step in dump analysis is to examine any symptoms that are recorded for the problem. This chapter discusses the Dump Symptoms function with which you may display or print the problem symptoms collected by the dumping component at the time of a failure or by subsequent analysis routines. The list of symptoms may indicate a new problem or a duplicate of a previously encountered problem. If sufficient symptoms are provided, they may pinpoint the cause of the failure.

The successful use of the Dump Symptoms function is dependent on the presence of a symptom record in the dump you are processing. The symptom record is created by the dumping component when the dump is taken (see Appendix C, "Symptom Record Overview" on page C-1).

Request Dump Symptoms by entering '2' on the function selection panel.

Contents

DISPLAYING DUMP SYMPTOMS

When you select Dump Symptoms, Info/Analysis displays the dump symptoms panel, as shown in Figure 14-1. You may not modify, add, or delete data in the body of this panel. It is likely that the symptoms are continued beyond what you see on the panel; you may scroll to view the rest of the symptoms.

```
BLNDS001                    DUMP SYMPTOMS
===> _                                          SCROLL ===> PAGE

DUMP NAME .... SYSDUMP.F6.00000010


ENVIRONMENT:
    CPU MODEL ....... 0148
    CPU SERIAL ...... 060707
    TIME ............ 08:44:51:42
    DATE ............ 84/04/12
    SYSTEM ID ....... 574500000
    RELEASE ......... v2
    FEATURE ......... 02
    DUMPTYPE ........ SCPREQ
    PROBLEM NUMBER ..

REQUIRED SYMPTOMS:
    AB/S0234
    PIDS/5745SCBTM
    RIDS/DFHTUB
    REGS/0C14E
    REGS/0A050
```

Figure 14-1. Dump Symptoms Panel

TYPES OF DUMP SYMPTOMS

The symptoms are organized into the following sections:

- Environment data
- Required symptoms
- Optional symptoms in structured data base (SDB) format
- Optional symptoms in non-SDB format

The symptoms are initially provided by the dumping component. If you subsequently execute analysis routines for the dump, these routines may add symptoms. A plus sign (+) appears before a symptom if it was added by an analysis routine.

The displayed symptoms may pinpoint the cause of the failure. If not, you may compare these symptoms to the symptoms of other locally reported problems. Your installation should set up a procedure whereby a file of problem symptoms is kept and the symptoms can be compared to one another.

If a satisfactory match is found, the problem is considered a local duplicate. If no match is found and the problem is related to an IBM product, a search of known IBM problems would be a logical next step. You may contact IBM service personnel to request this search. If a duplicate set of symptoms is found in either search, a solution may be immediately available or already under investigation. If no match or too many matches occur after a search, additional analysis is necessary. You may perform this analysis using Dump Viewing described in Chapter 15, "Dump Viewing" on page 15-1.

Environment

The environment section of the symptom record describes the environment at the time the dump was created. This section is provided by the dumping component. The CPU, operating system, type of dump, and date and time that the dump was taken are identified. Additional items such as release level may be included.

Required Symptoms

Required symptoms are those considered essential to your problem analysis efforts. They are provided by the dumping component or by subsequent analysis routines.

All of the required symptoms are likely to occur if the error occurs again. Required symptoms are in the format of the structured data base (SDB). The SDB format standardizes the expression of problem symptoms so that more effective keyword searches for duplicate problems may be conducted.

Each symptom is expressed with a prefix and the specific data connected by a slash. For a list of the prefixes and the data they represent, see VSE/System Package Diagnosis Guide. The required symptoms are also described under "Print Dump Symptoms" on page 3-17.

OPTIONAL SYMPTOMS (SDB)

SDB stands for Structured Data Base.

Optional symptoms (SDB) may be provided by the dumping component or by subsequent analysis routines. These are additional symptoms that apply to the problem and may be present if the problem reoccurs. These symptoms are in SDB format. For example, the component level may be included in this section.

Optional Symptoms (Non-SDB)

Optional symptoms (non-SDB) are optionally provided by the dumping component or by subsequent analysis routines. They are free-form symptoms that may be used in problem analysis but do not fit into the SDB format.

CHAPTER 15. DUMP VIEWING

Dump Viewing provides a basic set of display facilities. Within that structure, you may specify options that are appropriate for the type of error, local procedures, and your analysis techniques.

To examine a dump and determine other symptoms that may apply to the problem, select the Dump Viewing function by entering '3' on the function selection panel. Preferences for methods of dump analysis vary from one individual to the next.

Contents

SELECTING A DUMP VIEWING FUNCTION

When you select Dump Viewing, Info/Analysis displays the dump viewing selection panel shown in Figure 15-1.

```
BLNDVS01              DUMP VIEWING SELECTION
===> _

DUMP NAME .... SYSDUMP.F3.00000010

To select a function, type its number and press ENTER:

1  DUMP DISPLAY OPTIONS    Examine or change dump display options

2  DUMP DISPLAY            Examine dump data

3  ANALYSIS SUMMARY        Examine text, control block locations,
                           control block linkages, etc.

4  ANALYSIS ROUTINES       Select an analysis routine for execution
```

Figure 15-1. Dump Viewing Selection Panel

This panel is the gateway to the Dump Viewing functions. On the entry line of the panel, you may enter the selection number of the function you wish to perform. The functions from which you may choose are:

- Dump display options - examine or modify the settings that define the format of data presented during dump display.

- Dump display - examine areas of the dump in hexadecimal or formatted mode.

- Analysis summary - examine data pertinent to dump analysis including:

  - Sequences of control blocks

  - A list of named control blocks

  - Textual comments added to the dump by the dumping component or analysis routines

- Collections of hexadecimal information that may be pertinent to the problem

- Analysis routines - display and choose from a list of the dump analysis routines that you may call during an Info/Analysis session.

The functions available to you depend on the symptom record of the dump you are processing. For example, the dump display and the analysis summary functions rely on the use of section 6 of the symptom record. See Appendix C, "Symptom Record Overview" on page C-1, for an introduction to section 6. If the dumping component and subsequent analysis routines do not include control block locating and formatting information in the symptom record, you cannot display formatted data. Also, analysis summary data may not be available.

The analysis routines function is usable only if analysis routines are provided by a system component which are identified to Info/Analysis in the external routines file.

When you have completed a Dump Viewing function, Info/Analysis redisplays the dump viewing selection panel. You may then choose another function or leave Dump Viewing.

SPECIFYING DUMP DISPLAY OPTIONS

The format of the dump, as displayed on your terminal screen, depends on the options that you set on the dump display options panel. When you enter '1' on the dump viewing selection panel, Info/Analysis displays the dump display options panel, shown in Figure 15-2 on page 15-5. Each of the options is described in subsequent sections.

```
BLNDDP01            DUMP DISPLAY OPTIONS
===> _


Change the options as needed and press ENTER:

PREFERRED DISPLAY ===> F        F = Formatted   H = Hexadecimal


FORMATTED DISPLAY OPTIONS:
           OFFSETS ==> N        Display offsets with dump data
                                  Y = Yes     N = No
           LABELS ===> Y        Display field labels with dump data
                                  Y = Yes     N = No


HEXADECIMAL DISPLAY OPTIONS:
           OFFSETS ==> N        Display offsets with dump data
                                  Y = Yes     (display offsets)
                                  N = No      (display addresses)
```

Figure 15-2. Dump Display Options Panel

As shown in Figure 15-2, Info/Analysis provides defaults for the options. These defaults appear in the option fields the first time that you display the panel. If you wish, you may change the options during your Info/Analysis session. The most recent settings are saved across sessions in your user profile and are displayed on the panel.

To change an option, overtype the current setting. When you have typed all your changes, press the ENTER key. Keep in mind that if you enter an invalid character for any of the options, none of the options is changed. The settings you entered are displayed with the cursor placed under the first error and an error message is issued. You must correct all errors before any of the new settings take effect; if you leave the panel without correcting the error, none of the options are changed.

Once you have examined the options and changed any you wish, enter the END command and press the ENTER key to return to the dump viewing selection panel.

Preferred Display

In the PREFERRED DISPLAY field of the options panel, specify the mode for the presentation of dump data.

Enter 'f' if you prefer that data be displayed in formatted mode according to the specifications in section 6 of the dump's symptom record. If you specify formatted display and there is no formatting descriptor in section 6 for the data you are processing, the data is displayed in hexadecimal mode.

Enter 'h' if you prefer that data be displayed in traditional hexadecimal mode.

Formatted Display Options

You may choose to display formatted data with or without offsets and labels. Sample panels showing the results of the possible combinations are presented under "Formatted Display" on page 15-16.

A 'y' in the OFFSETS field means that formatted data is displayed with offsets from the BASE address next to the contents of the fields. One field of data is displayed per line. If 'n' is specified, data is displayed without offsets.

A 'y' in the LABELS field means that formatted data is displayed with labels next to the fields. Setting LABELS to 'n' suppresses the labels.

The appearance of a formatted display depends on the combination of LABELS and OFFSETS you specify. With OFFSETS set to 'n' and LABELS set to 'y', a display contains multiple labeled fields per line. With both OFFSETS and LABELS set to 'y', a display contains one labeled field per line.

Setting both fields to 'n' results in a display that contains one field of data per line without offsets or labels. This combination is not recommended for usual dump display tasks because the section 6 formatting descriptor used to present formatted data may specify fields out of sequence.

Hexadecimal Display Options

You may choose to display hexadecimal data with a prefix on each line that is either the offset of that line from the start of the displayed data (the address in the BASE field) or the full storage address for the line of data. Sample panels showing the results of these values are presented under "Hexadecimal Display" on page 15-19.

Enter 'y' in the OFFSETS field if you prefer the offset of the line of data from the BASE address. If 'n' is specified, the full storage address is displayed.

DISPLAYING DUMP DATA

Using the dump display function, you can examine the current dump on your terminal screen. When you select '2' from the dump viewing selection panel, Info/Analysis displays a dump display panel.

The fixed portion of the dump display panel is shown in Figure 15-3. The format of the actual data presented on a dump display panel depends on the dump display options in effect.

```
BLNDD001                         DUMP DISPLAY
===> _                                               SCROLL ===> PAGE

DUMP NAME .... SYSDUMP.xx.nnnnnnnn
DISPLAY ==>              BASE ==>              QUAL ==>           MODE ==>
```

Figure 15-3. Fixed Portion of the Dump Display Panel

The body of the first dump display panel is empty and the data entry fields on line 5 of the panel are blank because you have not yet specified an area of the dump. To display data, you must enter information in these data entry fields. See "Dump Display Data Entry Fields" on page 15-8.

Once you have displayed an area of the dump, you may use a variety of methods to display data at other addresses in the dump. These methods include:

- Cursor selection of an address to display data beginning at that address. This technique is described under "BASE Field" on page 15-10.

- Using the Find command to locate the occurrence of a particular hexadecimal or character string. A successful Find operation results in a display with the located data at the top of the scrollable area. The BASE address is then set to the address of the located data in a hexadecimal display only. A description of the Find command is given in Chapter 17, "Interactive Commands" on page 17-1.

- Scrolling from the address in the BASE field to the end of the displayable data. The length of the data is determined by the value in the DISPLAY field. A description of Scrolling is described under "Scrolling" on page 12-10.

- Changing the values in the data entry fields on line 5 of the panel to define a different address and format for the display. The fields are described below under "Dump Display Data Entry Fields."

Dump Display Data Entry Fields

The data entry fields on line 5 of the dump display panel control the display as follows:

- DISPLAY - defines the format.

- BASE - defines the beginning address of the current data.

- QUAL - defines the level of qualification, if any, for the address in the BASE field.

- MODE - defines the mode of access.

Together, the values in these fields define what data is presented. By overtyping the value in one or more of these fields and pressing the ENTER key, you may display other data (or change its format) in the body of the panel.

Keep in mind that certain aspects of dump display are dependent on the availability of entries in the symptom record. The values you enter in the data entry fields work in conjunction with the symptom record entries and the dump display options in effect. If you enter an invalid entry in any of these fields, or if Info/Analysis cannot use your entries because information is not available in the symptom record, the new display shows the values used for the display in the data entry fields and you receive an explanatory message.

DISPLAY Field

DISPLAY is a 1- to 8-character field that identifies the format to be used to display data. If you do not enter an address in the BASE field, DISPLAY also determines the amount of data presented.

If you wish to look at formatted data, enter the control block name in the DISPLAY field. If you also know the control block's address, enter that address in the BASE field. If you do not know the address, enter the name in the DISPLAY field and blank out the BASE, QUAL, and MODE fields. A description of how to use the DISPLAY field with the BASE field to display specific data is given under "Using the Data Entry Fields" on page 15-13.

The default DISPLAY value is DUMP. DUMP format presents data in hexadecimal mode. You may scroll backward and forward through the entire dump. Info/Analysis presents the data at the address in the BASE field in DUMP format in the following situations:

- If you do not specify a value in the DISPLAY field.
- If the symptom record does not contain a formatting descriptor by the name you specify
- If you are in hexadecimal mode.

If you specify a name in the DISPLAY field and there is a locator (which contains an address and a length) but no formatting descriptor by that name in the symptom record, the length of the displayable data is determined by the length indicated in the locator. You may scroll for that length only, regardless of whether the data displayed is determined by a BASE address or by the address in the locator. Figure 15-4 illustrates the case in which a length is taken from a locator.

```
BLNDD001                          DUMP DISPLAY
===> _                                            SCROLL ===> PAGE

DUMP NAME .... SYSDUMP.F3.00000010
DISPLAY ==> SCVT       BASE ==> 00122710    QUAL ==> 0001    MODE ==> V

00122710   E2C3E5E3 001225C0 001285A0 001226E0 00 *SCVT......e.....*
00122720   0011DA6C 0011DA64 0011DA68 00094DF8 00 *...}..........(8*
00122730   00095110 00095270 00095518 000953D0 00 *................*
00122740   000C001C 00080000 000F30                *...........    *
```

Figure 15-4. Hexadecimal Dump Display Panel with DISPLAY and BASE Values

In Figure 15-4, the name SCVT is entered in the DISPLAY field. Info/Analysis searches the symptom record for a locator named SCVT. When found, Info/Analysis extracts the length of the SCVT and uses it to control the amount of data displayed. In this case, the entry specified a length of 59 (X'3B') bytes. Info/Analysis displays 59 bytes of dump data beginning at the location in the BASE field, 122710. The data fits on one screen.

If the data does not fit on one screen, you can scroll to see the remainder of the data.

If Info/Analysis could not find a locator for the control block named in the DISPLAY field, DISPLAY would default to DUMP and a full screen of data starting with the address in the BASE field would be displayed. You could then scroll from the beginning to the end of the dump.

It is possible that a dump may contain control blocks that have the same name and different component identifiers. If you enter such a name in the DISPLAY field without entering a BASE address, Info/Analysis displays the data pointed to by the first locator by that name that it encounters in the symptom record.

There are two ways to display the other control blocks with the same name:

- Select the control block you wish to see from an analysis summary panel. You may select the control block from the LOCATORS display or from a linkage display that includes the control block. Both displays list the component identifier for each control block.

- If you know the address of the control block, enter that address in the BASE field. If you wish to see that data in formatted mode, you must also have specified formatted mode on the dump display options panel and entered the control block name in the DISPLAY field.

BASE Field

The BASE field is a 1- to 12-character field that contains the beginning address of the data to be displayed. Usually, this address is a virtual address. However, the definition of the BASE field may vary according to the QUAL and MODE field values.

Keep in mind that due to scrolling, this address may or may not be the beginning address of the content of the current screen. For instance, if you are looking at a control block and the data does not fit on one screen, you may scroll backward and forward through the data without changing the BASE address.

To look at a different area of the dump, you may change the BASE address in the following ways:

- Overtype the address in the BASE field and press the ENTER key. The typed value becomes the BASE address.

- Position the cursor under an address within the displayed data or under an address prefix (on the left side of the displayed data) and press the ENTER key. The address you select with the cursor becomes the BASE address. The selected address may be any string of two to eight hexadecimal digits. Any string preceded by '+' or '-' is treated as an offset to be added to the BASE address.

- Position the cursor under an offset prefix (on the left side of the displayed data) and press the ENTER key. The sum of that offset and the current BASE address become the new BASE.

The latter two options using cursor positioning, are valid only if you make no change to the BASE address or if you fill the BASE field with blanks.

If you overtype the BASE field and position the cursor in the body of the display, and press the ENTER key, your input is invalid. Be sure to use one technique or the other in one entry.

The BASE field may also be used to display data pointed to by the contents of a register. To do so, enter GRn in the BASE field where "n" is a hexadecimal register number in the range 0 - F. If a locator named GREGS is in the symptom record and is valid for general purpose registers, the content of the register you specified (n) is used as the BASE location.

QUAL Field

The QUAL field is a 1- to 8-character alphameric field that provides qualification for the address in the BASE field. Along with the contents of the BASE and MODE fields, the QUAL value is used to identify the dump data that you wish to display. If you are requesting data at a storage address, you do not need to enter a QUAL value.

QUAL accepts the following values according to the modes of access indicated:

- Relative record number of section 6 extension record for section 6 (E) mode of access

- Specific record number for record (N) mode of access

- For real (R), virtual (V), header (H), and symptom (S) mode of access, QUAL is ignored.

The qualification of the request specifies a particular set of dump data for which multiple iterations may exist. For example, the same virtual address may exist in several address spaces. The qualification in this case is the address space identification.

For example, if you wish to display a specific dump record, you can use the QUAL field to specify a record number, set the MODE field to "n" to indicate that you are retrieving a specific record, and set the BASE field to the offset of the data you wish to see from the beginning of the record.

A further use of the QUAL field is to look at the dump header or symptom record. QUAL can specify that a particular section of the record is of interest to you.

If you do not specify a QUAL value, the system dump access routine assumes one for you. This value, 0001, is displayed in the QUAL field when data is displayed.

MODE Field

Each request for dump data is qualified by a mode of access. Along with the content of the BASE and QUAL fields, the MODE value is used to identify the dump data you wish to display. Both the BASE and QUAL field contents vary depending on the specified mode.

In most cases, the request is for data at a virtual address. In some cases, you may prefer to specify a real address, the dump header, or other dump records. To meet these needs, the MODE field accepts the following values:

- E - section 6 extension record request
- H - header record request
- N - specific record request
- R - real address request
- S - symptom record request
- V - virtual address request

If you do not specify a MODE value, the system dump access assumes one for you. This value, V for virtual, is shown in the MODE field when data is displayed.

The values to use in the BASE and QUAL fields for particular modes of access are described in the following section.

Using the Data Entry Fields

The DISPLAY field and the address defined by the BASE, QUAL, and MODE fields determine what data is displayed. This section discusses how you can use these fields to display the data of interest to you.

There are six possible modes of access for dump data. The following list shows the values to be used in the BASE, QUAL, and MODE fields for each access mode:

- Extension record mode - MODE=E, QUAL=relative record number of section 6 extension record, BASE=offset into the extension record
- Header record mode - MODE=H, QUAL=unused, BASE=offset into the header record
- Dump record mode - MODE=N, QUAL=specific record number in dump records, BASE=offset into the record
- Real address mode - MODE=R, QUAL=unused, BASE=real storage address
- Symptom record mode - MODE=S, QUAL=unused, BASE=offset into the symptom record
- Virtual address mode - MODE=V, QUAL=unused, BASE=virtual storage address

If you do not specify QUAL and MODE, the BASE address applies to virtual storage, the usual case. Whenever you are looking at virtual storage, you will not use the QUAL and MODE fields. Therefore, the DISPLAY and BASE fields are of the most value to you. Figure 15-5 on page 15-14 shows the results of the possible combinations of DISPLAY and BASE values.

Displaying the data you wish to see depends on what information the dumping component or subsequent analysis routines have provided in section 6 of the symptom record. As shown in Figure 15-5 on page 15-14, the presence of locators and formatting descriptors dictates the results you achieve with the DISPLAY and BASE values. See Appendix C, "Symptom Record Overview," for a description of the symptom record.

```
*---------------------*
| DISPLAY -> name     |
| BASE    -> blank    |
*-------+-------------*
        |
        |                  *----------------*
        V                  | No change to   |
   locator      no         | data displayed;|
   available? -------->    | Error message  |
        |                  | is issued      |
   yes  |                  *----------------*
        |
        |                  *----------------------*
        V                  | Display data in      |
     in         no         | hexadecimal          |
   formatted ---------->   | beginning at         |
     mode?                 | address in locator   |
         |                 | for length in locator|
   yes   |                 *----------------------*
         |
         |
         |                 *----------------------*
         V                 | Display data in      |
   format       no         | hexadecimal          |
   descriptor --------->   | beginning at         |
   available?              | address in locator   |
         |                 | for length in locator|
   yes   |                 *----------------------*
         |
         V
  *-------------------------*
  | Display formatted data  |
  | beginning at            |
  | address in locator      |
  | according to format     |
  | and length in descriptor|
  *-------------------------*

 User enters DISPLAY name only
```

```
     *----------------*
     | DISPLAY -> blank |
     | BASE    -> blank |
     *----------------*
            |
            V
     *----------------*
     | No change to   |
     | data displayed;|
     | No message is  |
     | issued         |
     *----------------*

     User blanks out DISPLAY and BASE fields
------------------------------------------*

     *-------------------*
     | DISPLAY -> blank  |
     | BASE    -> address|
     *-------------------*
              |
              V
     *-------------------*
     |   Display data    |
     |   beginning at BASE |
     |   in DUMP format  |
     *-------------------*

     User enters BASE address only
------------------------------------------*

     *----------------*
     | DISPLAY -> DUMP  |
     | BASE    -> blank |
     *----------------*
              |
              V
     *----------------*
     |  Display data    |
     |  beginning at    |
     |  default address |
     |  in DUMP format  |
     *----------------*

     User enters DISPLAY of DUMP only
```

Figure 15-5 (Part 1 of 2). DISPLAY and BASE Field Processing

```
 -----------------------------------------------------------------------+------------------------------------------*
                                                                        |
  *----------------------*                                              |
  |   DISPLAY -> name    |                                              |
  |   BASE     -> address |                                              |
  *----------------------*                                              |
            |                                                           |
            |                                                           |
            V                                    *--------------------* |   *----------------------*
           in        no                       no |  Display data in   | |   |   DISPLAY -> DUMP    |
      formatted -------------> locator   ----->|  hexadecimal       | |   |   BASE     -> address |
        mode?         |        available?        |  beginning at BASE | |   *----------------------*
     yes   |          |            |             |  in DUMP format    | |              |
           |          |        yes |             *--------------------* |              V
           V          |            V                                    |   *----------------------*
      format          |no  *-----------------------*                   |   |   Display data       |
      descriptor---*       | Display data in       |                   |   |   beginning at BASE  |
      available?           | hexadecimal           |                   |   |   in DUMP format     |
           |               | beginning at BASE     |                   |   *----------------------*
      yes  |               | for length in locator |                   |
           |               *-----------------------*                   |   User enters DISPLAY of DUMP and
           V                                                            |   Base address
    *----------------------------*                                      |
    |  Display data              |                                      |
    |  beginning at BASE         |                                      |
    |  according to format       |                                      |
    |  and length in descriptor  |                                      |
    *----------------------------*                                      |
                                                                        |
   User enters DISPLAY name and BASE address                            |
                                                                        |
                                                                        |
```

Figure 15-5 (Part 2 of 2). DISPLAY and BASE Field Processing

Masking Data

You may mask (overlay) areas of the dump using the MASK command. The areas may contain sensitive data related to your organization or dumped output of non-IBM programs.one prior

The MASK command is valid only on dump display panels. Once data is masked, it cannot be restored to its original value. The MASK command is described in Chapter 17, "Interactive Commands" on page 17-1.

Formatted Display

In formatted mode, the display you see correlates field names or other identifiers to the dumped storage at a specified address. The format is defined in a formatting descriptor in the symptom record. Formatting can be done only for those control blocks for which there are formatting descriptors.

If you are in formatted mode and there is no formatting descriptor by the name you enter in the DISPLAY field, the data is presented in DUMP format (hexadecimal mode).

It is possible that a formatting descriptor might map only a subset of the control block fields; in formatted mode, only those fields are presented.

Figure 15-6 shows a formatted display with field labels for the SCVT control block. As shown in this example, the field contents of a formatted display may vary in length and may be character, decimal, binary, or hexadecimal digits indicated as follows:

- C - character display
- D - decimal display
- B - binary display
- No character - hexadecimal display

```
BLNDD001                          DUMP DISPLAY
===> _                                          SCROLL ===> PAGE

DUMP NAME .... SYSDUMP.F3.00000010
DISPLAY ==> SCVT      BASE ==> 00122710   QUAL ==> 0001   MODE ==> V

00122710   E2C3E5E3 001225C0 001285A0 001226E0 00 *SCVT......e.....*
00122720   0011DA6C 0011DA64 0011DA68 00094DF8 00 *...}..........(8*
00122730   00095110 00095270 00095518 000953D0 00 *................*
00122740   000C001C 00080000 000F30                *...........     *
```

Figure 15-6. Formatted Dump Display Panel with Labels

To see the display in Figure 15-6, you must have:

- Entered "f" in the PREFERRED DISPLAY field, "n" in the OFFSETS field, and "y" in the LABELS field of the dump display options panel.
- Entered SCVT in the DISPLAY field of the previous dump display.
- Blanked out or entered the address in the BASE field of the previous dump display.

Alternatively, you could have selected the SCVT during analysis summary from the LOCATORS display or from a linkage display.

Figure 15-7 on page 15-18 shows a formatted display with field offsets and labels for the SCVT control block. The offsets option causes each field to occupy a single line of the display and to be preceded by a plus sign (+) or minus sign (-) together with the offset of the field from the beginning of the displayed item (the address in the BASE field).

```
BLNDD001                   DUMP DISPLAY
===> _                                        SCROLL ===> PAGE

DUMP NAME .... SYSDUMP.F3.00000010
DISPLAY ==> SCVT     BASE ==> 00122710    QUAL ==> 0001   MODE ==> V

   +   0   SCVCHAR   C'SCVT'
   +   4   SCVSCVP   001225C0
   +   8   SCVLINE   001285A0
   +   C   SCVHDRP   001226E0
   +  10   SCVDMHP   0011DA6C
   +  14   SCVDMTOP  0011DA6C
   +  18   SCVDMSEL  0011DA68
   +  1C   SCVBLDP   00094DF8
   +  20   SCVFWDP   00095110
   +  24   SCVBWDP   00095270
   +  28   SCVTOPP   00095518
   +  2C   SCVBOTP   000953D0
   +  30   SCVSXOF   000C
   +  32   SCVFLDOF  001C
   +  34   SCVLNUM   D'8'
   +  36   SCVSTNUM  0000
   +  38   SCVLBLT   D'15'
   +  3A   SCVFLGS   B'00110000'
```

Figure 15-7. Formatted Dump Display Panel with Labels and Offsets

To produce the display in Figure 15-7, you must have:

- Entered "f" in the PREFERRED DISPLAY field and "y" in both the LABELS and OFFSETS fields of the dump display options panel.
- Entered SCVT in the DISPLAY field of the previous dump display.
- Blanked out or entered the address in the BASE field of the prior dump display.

Alternatively, you could have selected the SCVT during analysis summary from the LOCATORS display or from a linkage display.

If labels are not requested and offsets are, the display would contain one entry per line without field labels. If neither labels nor offsets are requested, the formatted data would be displayed without field identifiers, one entry per line.

Hexadecimal Display

Hexadecimal mode display is similar in format to that of traditional printed dumps. The only differences are due to screen size and, if you have specified a preference for offsets, the inclusion of offsets rather than addresses at the start of each line of data. Examples of hexadecimal dump display panels with and without offsets appear in Figure 15-8 and Figure 15-9, respectively. The panels show hexadecimal decoding and EBCDIC translation of dumped storage for a specified dump address along with the storage key. See "Dump Display Data Entry Fields" on page 15-8 for a description of the use of the DISPLAY, BASE, QUAL, and MODE fields.

```
BLNDD001                    DUMP DISPLAY
===> _                                          SCROLL ===> PAGE

DUMP NAME .... SYSDUMP.F3.00000010
DISPLAY ==> DUMP      BASE ==> 00122710   QUAL ==> 0001   MODE ==> V

   +   0   E2C3E5E3 001225C0 001285A0 001226E0 00 *SCVT......e.....*
   +  10   0011DA6C 0011DA64 0011DA68 00094DF8 00 *...}..........(8*
   +  20   00095110 00095270 00095518 000953D0 00 *................*
   +  30   000C001C 00080000 000F3000 00000000 00 *................*
   +  40   D9D9C3E4 00000000 00000000 00000000 00 *RRCU............*
   +  50   00000000 00000000 00000000 00000000 00 *................*
   +  60   00000000 00000000 00000000 00000000 00 *................*
   +  70   C6C5C3E4 0011DA0C 0011DA08 00000001 00 *FECU............*
   +  80   0000000E 0000000B 0000000E 00000021 00 *................*
   +  90   0000000F 0000000B 0000000F 00000024 00 *................*
   +  A0   00000010 0000000B 00000011 0000000B 00 *................*
   +  B0   00000010 0000000C 00000012 0000000C 00 *................*
   +  C0   00000013 0000000C 00000014 00000014 00 *................*
   +  D0   00000010 0000000E 00000011 0000000F 00 *................*
   +  E0   00000012 0000000E 00000011 00000011 00 *................*
   +  F0   00000011 0000002A 00000013 00000011 00 *................*
   + 100   00000013 0000002D 00000010 00000010 00 *................*
   + 110   00000011 00000010 00000013 00000010 00 *................*
```

Figure 15-8. Hexadecimal Dump Display Panel with Offsets

As illustrated in Figure 15-8, hexadecimal dump display with offsets causes displays in which each line of the panel body consists of the offset of the data from the base address, the dump data at that offset, the storage key, and an EBCDIC translation of that data.

```
BLNDD001                        DUMP DISPLAY
===> _                                              SCROLL ===> PAGE

DUMP NAME .... SYSDUMP.F3.00000010
DISPLAY ==> DUMP      BASE ==> 00122710   QUAL ==> 0001   MODE ==> V

00122710   E2C3E5E3 001225C0 001285A0 001226E0 00 *SCVT......e.....*
00122720   0011DA6C 0011DA64 0011DA68 00094DF8 00 *...}..........(8*
00122730   00095110 00095270 00095518 000953D0 00 *................*
00122740   000C001C 00080000 000F3000 00000000 00 *................*
00122750   D9D9C3E4 00000000 00000000 00000000 00 *RRCU............*
00122760   00000000 00000000 00000000 00000000 00 *................*
00122770   00000000 00000000 00000000 00000000 00 *................*
00122780   C6C5C3E4 0011DA0C 0011DA08 00000001 00 *FECU............*
00122790   0000000E 0000000B 0000000E 00000021 00 *................*
001227A0   0000000F 0000000B 0000000F 00000024 00 *................*
001227B0   00000010 0000000B 00000011 0000000B 00 *................*
001227C0   00000010 0000000C 00000012 0000000C 00 *................*
001227D0   00000013 0000000C 00000014 00000014 00 *................*
001227E0   00000010 0000000E 00000011 0000000F 00 *................*
001227F0   00000012 0000000E 00000011 00000011 00 *................*
00122800   00000011 0000002A 00000013 00000011 00 *................*
00122810   00000013 0000002D 00000010 00000010 00 *................*
00122820   00000011 00000010 00000013 00000010 00 *................*
```

Figure 15-9. Hexadecimal Dump Display Panel without Offsets

As illustrated in Figure 15-9, when hexadecimal data is displayed without offsets, each line of the panel body consists of an address for the line, the dump data for that address, the storage key, and an EBCDIC translation of that data.

Text Displays with Control Block Data

The dumping component or subsequent analysis routines may create text entries that are associated with specific control blocks. These entries are part of section 6 of the symptom record. Unlike the text entries that are displayed through the analysis summary function, these entries are displayed as part of the dump display when you specify the control block name in the DISPLAY field.

The component or routine that creates the entry specifies whether it should appear above or below the control block data. Figure 15-10 on page 15-21 is an example of a text entry for the RRCU control block. Whenever RRCU is entered in the DISPLAY field, the text appears in the display.

```
BLNDD001                          DUMP DISPLAY
===> _                                                    SCROLL ===> PAGE

DUMP NAME .... SYSDUMP.F3.00000010
DISPLAY ==> RRCU       BASE ==> 00122750   QUAL ==> 0001   MODE ==> V

SUBCHAIN NL1 IS VALID FROM THE RRCU.
FLAG1 BIT 4 OF RRCU IS SET BUT IS NOT CONSISTENT WITH RRFB STATUS.
RRFB STATUS IS ZERO AND ACTIVE CHAIN IS VALID.

00122750   D9D9C3E4 88000005 0091482C 00086FF8 00 *RRCU.....j....?8*
00122760   00470041 00129004 0012AD1C 00129448 00 *..............m.*
00122770   0101E9C3 D4E5E540 4040              00 *..ZCMVV         *
```

Figure 15-10. Dump Display Panel for Control Block with Associated Text

The dump display options for the above panel are hexadecimal mode with addresses. The same principles apply to other screen formatting. All functions of the dump display panel are valid for this type of panel. Cursor positioning for changing the BASE field can be done using both the dump data and valid hexadecimal fields, if any, in the text data.

EXAMINING ANALYSIS SUMMARY DATA

When you select '3' from the dump viewing selection panel, Info/Analysis displays the analysis summary panel which lists analysis-related data that is identified in the dump symptom record. Figure 15-11 on page 15-22 is a sample analysis summary panel.

```
BLNAS001                    ANALYSIS SUMMARY
===> _                                          SCROLL ===> PAGE

DUMP NAME .... SYSDUMP.F3.00000010

To select an item for display, type 's' before its name and
press ENTER:

   BLOCK        COMP          TYPE

  BDDTEXT1    VSE/AF       TEXT DATA
  BDDTEXT2    VSE/AF       TEXT DATA
  BDDTEXT3    VSE/AF       TEXT DATA
  BDDBASE     VSE/AF       LINKAGE
  REGS        VSE/AF       HEXADECIMAL DATA
  BDDRSVT     VSE/AF       LINKAGE
  BDDCSVT     VSE/AF       LINKAGE
  BDDSCVT     VSE/AF       LINKAGE
  BDDLIBR     VSE/AF       LINKAGE
  IVZLIBR     ANLRTNA     +TEXT DATA
  LOCATORS                 CONTROL BLOCK LIST
  ANLRTNA                 +ANALYSIS ROUTINE OUTPUT
```

Figure 15-11. Analysis Summary Panel - Main Display

This panel lists analysis-related entries that you may select for display. The BLOCK column is the entry name. The COMP column is the name of the component that placed the information in the dump. The TYPE column defines the category of the entries, described in detail in the subsequent sections. Possible types are:

- Text data - text entries added to the symptom record by the dumping component or subsequent analysis routines.
- Hexadecimal data - collections of hexadecimal data added to the symptom record by the dumping component or subsequent analysis routines.
- Linkages - entries added to the symptom record by the dumping component or subsequent analysis routines. Each linkage descriptor describes a sequence of connected control blocks.
- Control block list - an entry called LOCATORS that defines a summary of all the locators included in section 6 of the symptom

record by the dumping component and subsequent analysis routines.

- Analysis Routine Output - an entry containing the display output of the last analysis routine run for this dump during this session.

A list may be continued on additional panels. To view the rest of a list, use the scroll PF keys or commands. Once you locate an entry of interest, you may select it for display by positioning the cursor at the beginning of the line containing the desired entry and entering 's'.

When you make a selection, Info/Analysis displays a panel containing the item as described in the following sections. Once you have examined the item, you may select another analysis summary entry for display. To do so, you must enter the END command to return to the analysis summary panel and make another selection. You may also, however, select more than one entry from any analysis summary selection panel in one interaction. If you do so, Info/Analysis displays your first selection. When you enter the END command, the next selection is displayed and so on.

Once you have viewed the entries, enter the END command to return to the dump viewing selection panel or the RETURN command to go back to the function selection panel.

If a plus sign (+) appears next to an item in the list, the item was created by an analysis routine after the dump was taken. The IVZLIBR entry in the panel in Figure 15-11 on page 15-22 exemplifies this.

If the data created by an analysis routine is a permanent part of the dump, it appears near or at the end of the list, immediately preceding the LOCATORS entry, if there is one. If the data created by an analysis routine is display output, the entry appears last in the list. Its name is the name of the analysis routine that created it. No more than one routine at a time can use the display function. The output is saved until:

- A call for Dump Viewing is made for a different dump.
- Another analysis routine is run.
- The session is ended.

Consequently, it is advisable to select analysis summary and look at the display output as soon as possible after executing an analysis routine.

Text Display

Text data may be created by the dumping component or subsequent analysis routines and added to the symptom record. Figure 15-12 is an example of a panel displaying a text entry.

```
BLNAS002                    ANALYSIS SUMMARY
===> _                                          SCROLL ===> PAGE

DUMP NAME .... SYSDUMP.FB.00000010
DISPLAY .... BDDTEXT1

 ERROR OCCURRED DURING A SCREEN UPDATE.
 ALL SCREEN LINE ENTRIES WERE ALLOCATED SUCCESSFULLY.
 ROUTINE BDDSCRBC DETECTED DATA ERROR WHILE INSERTING DATA
     INTO LINE BUFFER.
 SCRLINE CONTROL BLOCK AT 001298C0.
 ATTEMPT TO RECOVER HAS FAILED FOR THAT LINE.
```

Figure 15-12. Analysis Summary Panel - Text Display

The entry being displayed is named in the DISPLAY field. The text is usually sentences or phrases. It may be the result of any validation checks done by the failing component or it may suggest that a particular condition exists that requires further analysis. The text is displayed exactly as entered by the dumping component or analysis routine and is formatted to fit the terminal screen. Info/Analysis does not process the data in any way.

Hexadecimal Display

Hexadecimal data may be extracted from the dump and formatted into symptom record entries by the dumping component or subsequent analysis routines. Figure 15-13 on page 15-25 is an example of a panel displaying a hexadecimal entry.

```
BLNAS002                     ANALYSIS SUMMARY
===> _                                          SCROLL ===> PAGE

DUMP NAME .... SYSDUMP.F3.00000010
DISPLAY .... GREGS


  GP REGS  0-3   9110900F  48708036  8002984C  00000000
           4-7   0033FCB0  003FD5CC  00002000  00000000
           8-B   003FB910  003FCF48  0004F860  003FD518
           C-F   00029738  00054808  00054898  00029758
```

Figure 15-13. Analysis Summary Panel - Hexadecimal Display

The entry being displayed is identified in the DISPLAY field. The data is formatted to fit the terminal screen and displayed as entered by the dumping component or analysis routine. Info/Analysis does not process the data in any way.

A flag in the entry identifies the data as register data, PSW data, or other hexadecimal data. General purpose register and control data are formatted as shown in Figure 15-13. Floating point register data is displayed with two registers per line. PSW data is displayed with the PSW identifier. Should more data be found than expected, all data is presented in hex format.

Other hexadecimal data is shown in standard dump format. This format consists of an offset from the start of the hexadecimal data and the number of display bytes that fit the line. The far right contains EBCDIC translation of that data.

Linkage Display

A linkage descriptor is a symptom record entry that defines a sequence of connected control blocks. The control blocks are related to each other and their presentation in context is usually necessary for dump analysis. The sequence consists of an anchor control block, all the control blocks pointed to by the anchor, control blocks pointed to by those control blocks, etc.

Figure 15-14 is an example of a panel displaying a linkage descriptor. To create this display, Info/Analysis also uses locators from the dump symptom record.

```
BLNAS003                    ANALYSIS SUMMARY
===> _                                           SCROLL ===> PAGE

DUMP NAME .... SYSDUMP.F3.00000010
DISPLAY .... BDDSCV

   BLOCK        COMP    MODE  QUAL  BASE(ADDR)       KEYFIELD
  SYVT          VSE/AF   V    0007   00120948   SYVREMN   = 00039569
   SCVT         VSE/AF   V    0007   00122320   SCVLBLT   = D'0'
    SCLIBHDR    VSE/AF   V    0007   001212F0   DATA NOT FOUND
   SCVT         VSE/AF   V    0007   00122470   SCVLBLT   = D'9'
    SCLIBHDR    VSE/AF   V    0007   00122590   SCLNUM    = 6
     SCRLINE    VSE/AF   V    0007   00129C40   SCRLNERC  = 00
     SCRLINE    VSE/AF   V    0007   00129BD0   SCRLNERC  = 00
     SCRLINE    VSE/AF   V    0007   00129B60   SCRLNERC  = 00
     SCRLINE    VSE/AF   V    0007   00129AF0   SCRLNERC  = 00
     SCRLINE    VSE/AF   V    0007   00129A80   SCRLNERC  = 00
     SCRLINE    VSE/AF   V    0007   00129A10   SCRLNERC  = 00
   SCVT         VSE/AF   V    0007   001227E0   SCVLBLT   = D'12'
    SCLIBHDR    VSE/AF   V    0007   00122770   SCLNUM    = 8
     SCRLINE    CICS     V    0007   001299A0   SCRLNERC  = 00
     SCRLINE    CICS     V    0007   00129930   SCRLNERC  = 00
     SCRLINE    CICS     V    0007   001298C0   SCRLNERC  = 05
     SCRLINE    CICS     V    0007   00129850   SCRLNERC  = 00
```

Figure 15-14. Analysis Summary Panel - Linkage Descriptor Display

The panel you see when you select a linkage entry from the analysis summary panel describes direct control block relationships. These relationships are reflected in the way the data on the screen is indented; that is, any control blocks pointed to by a given control block are indented under that control block. However, the indentation is done only to the fourth level. That is, all control blocks fifth or later in the chain are indented to the same degree in the display. In the example, the linkage relationship is:

SYVT --> SCVT --> SCLIBHDR --> SCRLINE

As a result, the display shows:

- The anchor control block, the SYVT.
- Each SCVT chained from the SYVT.
- The SCLIBHDR for each SCVT.
- Each SCRLINE chained from each SCLIBHDR.

For each of the listed control blocks, the linkage display presents:

- BLOCK - the name of a control block.
- COMP - the name of the component that added the entry to the dump.
- MODE, QUAL, BASE - together these fields define the location of the control block. See "Dump Display Data Entry Fields" on page 15-8 for a description of these fields.
- KEYFIELD - a 1- to 8-character string used as a diagnostic indicator or a field in the control block that contains information that may be pertinent to your dump analysis activities. This field may be a flag or other indicator that has been determined to be of particular significance to that control block.

  For example, in Figure 15-14 on page 15-26, the field SCRLNERC in the SCRLINE control block may indicate a code that describes the status of that block. If the code is zero, there may be nothing unusual about the control block. A nonzero SCRLNERC may indicate some significant condition or that the control block may be a possible starting point for further analysis.

  If the KEYFIELD column indicates DATA NOT FOUND, the storage area for this control block is not available.

Locators Display

A locator is an entry in the dump symptom record that describes the location of control blocks in the dump. Locators specify simple control block addresses, chains of similar control blocks, or arrays of similar control blocks. See Appendix C, "Symptom Record Overview" on page C-1, for a description of a locator.

If you select the LOCATORS entry from the analysis summary panel, Info/Analysis lists all the control blocks in the dump for which a locator exists. An example of this panel appears in Figure 15-15 on page 15-28.

```
BLNAS004                    ANALYSIS SUMMARY
===> _                                          SCROLL ===> PAGE

DUMP NAME .... SYSDUMP.F3.00000010
DISPLAY .... LOCATORS

   BLOCK        COMP     MODE   QUAL BASE(ADDR)     KEYFIELD
  TCB         VSE/AF      V     0007   00C20948   TCBCMPC = 000000
  TCB         VSE/AF      V     0007   00C32A48   TCBCMPC = 00080A
  RB          VSE/AF      V     0007   009353C8
  RB          VSE/AF      V     0007   00935410
  RB          VSE/AF      V     0007   00935490
  RB          VSE/AF      V     0007   00935510
  SCVT        VSE/AF      V     0007   00122320   SCVLBLT = 00
  SCVT        VSE/AF      V     0007   00122470   SCVLBLT = 09
  SCVT        VSE/AF      V     0007   001225C0   SCVLBLT = 0F
  SCVT        VSE/AF      V     0007   00122710   SCVLBLT = 0E
  SCLIBHDR    VSE/AF      V     0007   001222F0
  SCLIBHDR    VSE/AF      V     0007   00122440
  SCLIBHDR    VSE/AF      V     0007   00122590
  SCLIBHDR    VSE/AF      V     0007   001226E0
  SCRLINE     CICS        V     0007   0012A890
  SCRLINE     CICS        V     0007   0012A900
  SCRLINE     CICS        V     0007   0012A970
  SCRLINE     CICS        V     0007   0012A9E0
```

Figure 15-15. Analysis Summary Panel - Locators Display

The control blocks in the analysis summary locators display are listed in the order in which the locators describing them were found in the symptom record. For each control block, the following information is included:

- BLOCK - the name of a control block.

- COMP - the name of the component that added the entry to the dump.

- MODE, QUAL, BASE - together these fields define the location of the control block. See "Dump Display Data Entry Fields" on page 15-8 for a description of these fields.

- KEYFIELD - a 1- to 8-character string used as a diagnostic indicator or a field in the control block that contains information that may be pertinent to your dump analysis activities. This field may be a flag or other indicator that has been determined to be of particular significance to that control block.

  For example, in Figure 15-15, the field SCVLBLT in the SCVT control block may indicate a code that describes the status of that block. If the code is zero, there may be nothing unusual about the control block. A nonzero SCVLBLT may indicate some significant condition or that the control block may be a possible starting point for further analysis.

If the KEYFIELD column indicates DATA NOT FOUND, the storage area for this control block is not available.

INVOKING AN ANALYSIS ROUTINE

Analysis routines, supplied by system components may be executed in the Info/Analysis environment. These analysis routines access dump data and the symptom record; they provide additional information that assists you in dump analysis.

When you enter '4' from the dump viewing selection panel, Info/Analysis displays the analysis routines panel. An example of this panel is shown in Figure 15-16.

```
BLNAR001                    ANALYSIS ROUTINES
===> _                                          SCROLL ===> PAGE

DUMP NAME .... SYSDUMP.F3.00000010

To select a routine for execution, type 's' before its name
and press ENTER:

  ROUTINE NAME          DESCRIPTION
  IJBXDBUG         Analyze Stand-Alone Dump
```

Figure 15-16. Analysis Routines Panel

This panel lists the names and short descriptions of analysis routines known to Info/Analysis. This list is obtained from the external routines file that is created and maintained by your installation. The list of available analysis routines may be continued on additional panels. To view the rest of the list, use the scroll commands.

You may execute one of the routines in the displayed list by positioning the cursor at the beginning of the line containing the desired entry, typing 's', and pressing the ENTER key.

When you have made a selection, the called routine begins processing. When processing is completed, you may press the END PF key to display the dump viewing selection panel. On this panel, you may choose another dump viewing function or return to the function selection panel.

Analysis routines may have the following kinds of output:

- Print output - some or all of the results of the routine are routed to the printer. This output cannot be displayed.
- Symptom record update - one or more symptoms may be added to the dump's symptom record and can subsequently be displayed via the Dump Symptoms function.
- Section 6 output - some or all of the results of the routine may be added to section 6 and included in subsequent analysis summary displays.
- Display output - some or all of the results of the routine may be temporarily saved and displayed on your terminal. This output may be seen by selecting the item from the analysis summary panel for the dump. The item has the same name as the routine that created it.

For a description of the analysis routine IJBXDBUG, please refer to "The Stand-Alone Dump Analysis Routine IJBXDBUG" on page 4-8.

CHAPTER 16. DUMP OFFLOAD AND ONLOAD

Together, the Dump Offload and Dump Onload functions enable you to control the contents of the system dump library. The length of time required to offload or onload a dump is proportional to the size of the dump and the system load. Due to this I/O dependency and a dependency on the system operator for tape handling, onload and offload jobs are submitted to background and a mount message is issued. If a dump onload or offload request were executed in interactive mode, Info/Analysis would begin the job and wait until its completion before accepting the next command. This could lock out the use of the terminal for long periods of time while onloading or offloading large dumps. For this reason, when running in interactive mode, Info/Analysis automatically submits the request to batch. While the job is executing, you may use Info/Analysis in interactive mode for additional dump examination and analysis. When the onload or offload is complete you may select the dump for viewing and analysis.

Dump Offload copies a dump file to tape for later retrieval. You select Dump Offload by entering '4' on the function selection panel.

Dump Onload copies a dump file to a dump sublibrary. You select Dump Onload by entering '5' on the function selection panel.

Contents

OFFLOADING A DUMP

When you enter '4' (DUMP OFFLOAD) on the function selection panel, Info/Analysis responds by displaying the dump offload panel shown in Figure 16-1.

The Dump Offload Panel

```
BLNDF001                    DUMP OFFLOAD
===> _

DUMP NAME .... SYSDUMP.BG.VALIDLBD

Enter data for dump offload and press ENTER to submit to batch:

  VOLID    ===> ______        Type the output volume identifier

  ERASE    ===> Y             Erase system copy of dump

  BYPASS   ===> N             Prevent copying dump offline if valid
                              copy exists
  SYSNUM   ===> ________      Type the tape symbolic unit name

  TAPADR   ===> ___           Type the tape channel and unit address

  TAPMDE   ===> __            Type the tape mode setting


   CHANGE DEFAULT VALUES IF REQUIRED


===> * $$ JOB JNM=INFOSBT,CLASS=A,DISP=D
===> * $$ LST CLASS=A,DISP=D
===>
```

Figure 16-1. Dump Offload Panel

DUMP NAME Field

The name appearing in the DUMP NAME field of the panel is the name of the current dump, the dump that is to be offloaded. You may offload only the current dump.

VOLID Field

If the dump has not been previously onloaded or offloaded, you must enter a volume id in the VOLID field. If the dump was previously onloaded or offloaded, this field is primed with the volume id that is stored in the dump management file. To override the displayed volume id, overtype it with the new value.

ERASE Field and BYPASS Field

There are three modes of operation for a dump offload:

1. You may offload the dump to tape and erase the system copy after the offload is complete (ERASE=Y).

2. You may offload the dump without erasing the system copy (ERASE=N).

3. You may erase the copy of the dump in the dump sublibrary without first offloading it, provided that an offloaded copy exists (BYPASS=Y). (To erase a dump without retaining a copy on tape, use the Dump Management delete function, described under "Deleting a Dump" on page 13-6.)

These options are controlled by entries in the ERASE and BYPASS fields on the dump offload panel. When the panel is displayed, these fields are primed with the defaults Y and N respectively, which cause the dump to be offloaded and the system copy to be erased. To override these defaults, overtype the displayed values with the new values.

By specifying 'n' in the ERASE field, you have the opportunity to offload multiple copies of the same dump. For example, you may wish to offload a dump to tape, then run analysis routines or mask data, and then offload the dump again. If you offload more than one copy of the dump, the volume id used for the most recent offload is the one placed in the dump management file. No reference to previous offloads is retained.

If you specify 'y' in the BYPASS field, the dump is erased from the system, freeing library space, without writing a copy to tape. Information about the dump remains in the dump management file. As stated on the panel, you may specify 'y' only when a valid offloaded copy of the dump exists; that is, the following conditions are met:

- The current dump has been previously offloaded.

- The current dump has not been modified by an analysis routine or by masking since it was last offloaded.

You may not set the ERASE value to 'n' and the BYPASS value to 'y' together. Together these values indicate that the dump should not be offloaded to tape nor should the system copy of the dump be erased; that is, no operation is requested.

Optional Tape Fields

Three fields are provided on the panel to give the capability of assigning a specific tape unit. These fields are optional and if not entered, the tape unit will be assigned automatically by the system.

SYSNUM FIELD: The symbolic unit name represents the logical unit number (SYSnnn) which can be assigned to a physical tape address via the job control 'ASSGN' statement. If you do not define a symbolic unit name allocation is done automatically by the system. The first available unit will be allocated and a message will be issued for the tape to be mounted on this unit.

TAPADR FIELD: The TAPADR field enables you to define the physical unit address of the tape unit.

TAPMDE FIELD: You may define the tape mode setting in the TAPMDE field.

Job Control Lines

Three lines are provided on the panel for VSE POWER and job control statements.

1. Line one is used for a POWER job statement. If there is no existing POWER job statement in the analyst's profile the system will create the following default POWER job statement:

    - * $$ JOB JNM=INFOSBT,CLASS=A,DISP=D

2. Line two is used for a POWER list statement. If there is no existing POWER job statement in the analyst's profile the system will create the following default POWER list statement:

    - * $$ LST CLASS=A,DISP=D

3. Line three is used for library definition. There is no default provided for this statement and, unlike the previous two statements, it is optional. If this statement is present the system will check columns one through ten for the character string:

    - // LIBDEF

If the JOB or LST statements are not present or the library definition is in error, a message will be presented to the analyst and processing will wait until the statement is corrected.

Once you have provided the required information and set the desired options, press the ENTER key to submit the offload job to batch.

When the job executes in the batch partition the console operator will receive a message to mount the tape on the first free tape drive unless the tape unit options have been entered. The volume id requested will be the same volume id requested by the interactive analyst during the interactive portion of the job. If no tape drives are available, the job will terminate unsuccessfully and an error message will be printed.

When the batch job to offload the dump has been submitted, you can enter the END command to display the function selection panel where you can select another function or end your session.

If you wish to leave the dump offload panel without offloading a dump, enter the END or RETURN command. You return to the function selection panel where you can select another function or end your session.

ONLOADING A DUMP

To process a dump using Info/Analysis, the dump must reside in the system. When you enter '5' (DUMP ONLOAD) on the function selection panel, Info/Analysis responds by displaying the dump onload panel shown in Figure 16-2 on page 16-7.

The Dump Onload Panel

On the dump onload panel, you specify the information needed by Info/Analysis to load a dump into a dump sublibrary.

```
BLNDN001                    DUMP ONLOAD
===> _

DUMP NAME .... SYSDUMP.BG.VALIDLBD

Enter data for dump onload and press ENTER to submit to batch:

   VOLID ===> T42901        Type the volume identifier

   FILE  ===> 0001          Type the file sequence number

   SYSNUM  ===> ________    Type the tape symbolic unit name

   TAPADR  ===> ___         Type the tape channel and unit address

   TAPMDE  ===> __          Type the tape mode setting






   CHANGE DEFAULT VALUES IF REQUIRED


===> * $$ JOB JNM=INFOSBT,CLASS=A,DISP=D
===> * $$ LST CLASS=A,DISP=D
===>
```

Figure 16-2. Dump Onload Panel

DUMP NAME Field

The name appearing in the DUMP NAME field of the panel is the name of the current dump, the dump that is to be onloaded. If a dump name is not saved in your user profile, you must enter this name on the function selection panel or the dump management panels before selecting dump onload.

If you wish to onload a dump that has not yet been introduced to the system, the dump requires a name. In this case, the name you specify in the DUMP NAME field of the function selection panel or the dump management panel must conform to the system conventions for dump names which are described under "Dump Name Conventions" on page 13-4.

VOLID Field

If the dump has not been previously offloaded or onloaded, you must enter the identifier of the volume on which the dump resides in the VOLID field. If the dump has been previously onloaded or offloaded, this field is primed with the volume id that is stored in the dump management file. Figure 16-2 on page 16-7 shows this condition. To override the displayed volume id, overtype it with a new value.

FILE Field

To onload one or more dumps from a multiple dump tape, use the FILE field to position the tape correctly by specifying the sequence number of the dump file you wish to onload. The tape must meet the following conditions:

- The dumps it contains must have been operator requested.
- The tape must be unlabeled.

If you wish to onload dumps from a multiple file tape that does not meet these conditions, you must position the tape via JCL.

The FILE field is primed with "0001." If the input tape does not contain multiple dumps or if you are onloading the first file of a multiple file tape, you need not change this field. To override the default, overtype it with a new value.

Optional Tape Fields

Three fields are provided on the panel to give the capability of assigning a specific tape unit. These fields are optional and if not entered, the tape unit will be assigned automatically by the system.

SYSNUM FIELD: The symbolic unit name represents the logical unit number (SYSnnn) which can be assigned to a physical tape address via the job control 'ASSGN' statement. If you do not define a symbolic unit name allocation is done automatically by the system. The first available unit will be allocated and a message will be issued for the tape to be mounted on this unit.

TAPADR FIELD: The TAPADR field enables you to define the physical unit address of the tape unit.

TAPMDE FIELD: You may define the tape mode setting in the TAPMDE field.

Job Control Fields

Three lines are provided on the panel for VSE POWER and job control statements.

1. Line one is used for a POWER job statement. If there is no existing POWER job statement in the analyst's profile the system will create the following default POWER job statement:

   - * $$ JOB JNM=INFOSBT,CLASS=A,DISP=D

2. Line two is used for a POWER list statement. If there is no existing POWER job statement in the analyst's profile the system will create following default POWER list statement:

   - * $$ LST CLASS=A,DISP=D

3. Line three is used for library definition. There is no default provided for this statement and, unlike the previous two statements, it is optional. If this statement is present, the system will check columns one through ten for the character string:

   - // LIBDEF

If the JOB or LST statements are not present or the library definition is in error, a message will be presented to the analyst and processing will wait until the statement is corrected.

Once you have supplied the necessary information, press the ENTER key to submit the batch job.

When the job executes in the batch partition the console operator will receive a message to mount the tape on the first free tape drive unless the tape unit options have been entered. The volume id requested will be the same volume id requested by the interactive analyst during the interactive portion of the job. If no tape drives are available the job will terminate unsuccessfully and an error message will be printed.

If you attempt to onload a dump and a dump by that name already resides in the system, Info/Analysis informs you of that condition. You must supply a different name for the dump you are onloading or erase the dump that resides in the system before you can onload the dump. To erase the dump, use the Dump Management delete function, which is described under "Deleting a Dump" on page 13-6.

If you wish to leave the dump onload panel without onloading a dump, enter the END or RETURN command. You return to the function selection panel where you may select another function or end your session.

CHAPTER 17. INTERACTIVE COMMANDS

Most interactions in Info/Analysis with ISPF take place through panels and responses to those panels. However, a set of commands is provided for some purposes.

This section presents the commands in alphabetical order. For each command, the syntax and usage are described, along with a list of the panels for which the command is valid. A summary of this information is contained in "Summary" which follows the command descriptions in this chapter.

Contents

COMMAND ENTRY

You enter a command by typing it after the arrow (===>) on the entry line and pressing the ENTER key, or by pressing a PF key if the command is represented by one. In the syntax descriptions in this chapter, the following conventions are used:

- Operands shown in braces { } represent alternatives; you must choose one.
- Operands shown in brackets [ ] are optional; you may choose one or none.
- An underscored operand represents the default assumed if you do not specify an operand.
- Operands shown in lowercase are variable; you substitute a value for them.
- Operands shown in uppercase should be entered as shown, but they are acceptable in uppercase or lowercase.
- Commands and operands shown in mixed case should be entered using at least the letters shown in uppercase. For example, INCRement may be entered as INCR, INCRE, INCREM, etc. You may enter FInd as FI, FIN, or FIND.

For tutorial processing, you may also enter ISPF commands. For a description of these commands, see hdref refid=iaohe..

END - RETURN TO LAST PANEL OR END SESSION

Use the END command to conclude the current operation and return to the previous logical panel. The previous logical panel is the one that precedes the current panel in the panel hierarchy. The previous panel is not necessarily the last panel displayed; you may have skipped its display by using response chaining. If you enter END on the function selection panel, the session is canceled.

```
END
```

When entered on a tutorial panel, END terminates the tutorial and causes the redisplay of the panel from which the HELP command was issued.

FIND - LOCATE DATA

Use the FInd command to locate specific data in a display. The data may be either hexadecimal data or a character string in a dump display, or a character string in a dump management, dump symptoms, analysis summary, or analysis routines display. A successful FInd operation results in a display with the located data at the top line of the scrollable area. On a dump display panel, a successful FInd operation also results in setting the BASE address to the address of the located data for a hexadecimal display only.

You must supply either a hexadecimal or character string with the Find command. The addr-range and INCRement operands are optional and can be entered only from dump display panels in hexadecimal mode.

```
FInd        { X'hexdata'} [from-addr            ] [INCRement incr]
            {chardata   } [from-addr to-addr    ]
                          [from-addr END        ]
                          [from-addr FOR length]
```

---

hexdata
a string of one to eight hexadecimal bytes that you wish to find, enclosed in quotes and preceded by the character 'x'. Info/Analysis searches for this value within the specified or default (8192 bytes) address range. The hexdata operand is valid only on hexadecimal dump display panels.

chardata
a string of one to eight characters that you wish to find. The string must not include any blanks. For dump display, Info/Analysis searches for this value within the specified or default (8192 bytes) address range. If you are searching for characters in a dump management, dump symptoms, formatted dump display, analysis summary, or analysis routines panel, the data you are searching for cannot span lines. The entire data entry is scanned for the chardata value.

addr-range
defines the area to be searched for the specified data. You may enter addr-range in the following forms:

from-addr
from-addr marks the beginning of the area to be searched. The search starts at this address and continues for 8192 bytes.

from-addr to-addr
from-addr marks the beginning of the area to be searched and to-addr marks the end. The to-addr must be equal to or greater than the from-addr.

from-addr END
from-addr marks the beginning of the area to be searched and END is a character string that indicates that the search should continue to the maximum storage address for the dump.

from-addr FOR length
from-addr marks the beginning of the area to be searched. The search continues for the number of bytes indicated by length, which is a number in hexadecimal notation.

All addresses are 1 to 4 bytes long representing valid hexadecimal addresses in the dump. Leading zeros are not required.

The addr-range operand is valid only on a hexadecimal dump display panel; on all other panels, it is invalid. If you do not specify an address range for the FInd command on a hexadecimal dump display panel, the search begins with the line after the current row (line two of the display) and continues for a length of 8192 bytes or until the end of the data, whichever comes first. If you specify an address range, the search begins from the address specified. On all other panels, the search begins with the line after the current row (line two of the display) to the end of the displayable data. This includes the current display and any additional data that may be displayed using scrolling functions.

INCRement incr
causes a search for the data in the specified or default addr-range at the byte increment specified. The increment (incr) is a hexadecimal number of 1000 or less. If you specify both addr-range and INCRement, addr-range must precede INCRement.

If you enter the FInd command without entering an increment, the default is one byte; thus Info/Analysis looks for the desired data at each byte of dump data within the specified or default address range. An incremental search is useful when you know that the data begins on a particular boundary.

EXAMPLES

1. Find the address of the hexadecimal value D202B106 in a hexadecimal dump display panel. Start the search at the line following the current row and search the next 8192 bytes of data for a match. Display data beginning with that address.

```
fi x'd202b106'
```

2. Find the address of the character string 'AVBHMAS' in a hexadecimal dump display panel by scanning the area from address 17000 to address 17800 and display data beginning with that address.

   ```
   fi avbhmas 17000 17800
   ```

3. Find the address of the character string 'QELMR14A' in a hexadecimal dump display by searching from address 1F8C94 to 1F9C94. Test the data at every increment of four bytes from 1F8C94 for the presence of the character string. Ignore any occurrence of the data that is not on a word boundary.

   ```
   fi qelmr14a 1f8c94 for 1000 incr 4
   ```

4. Find the dump SYSDUMP.F3.079543 by searching for 479543 in the dump management panel.

   ```
   fi 079543
   ```

HELP - DISPLAY TUTORIAL INFORMATION

Use the HELP command to display instructional information about Info/Analysis, its panels, and its messages.

```
HELP
```

Entering HELP has the following results:

- If you enter HELP when a panel is displayed with no message on the message line, a tutorial panel for that panel is displayed. Enter the END command to return to the panel for which you requested help.

- If you enter HELP when a panel is displayed with a message on the message line, a tutorial panel describing the message and appropriate responses to it is displayed. To return to the panel from which help was requested, enter the END command. To get help for the panel from which help was requested, press the ENTER key.

- If you enter HELP when a tutorial panel is displayed, the tutorial help panel which describes how to use the tutorial is displayed.

- If you enter HELP from a message help panel, the message help tutorial panel is displayed. This panel describes the message help panel.

For information about the tutorial, see "Online Help Information" on page 12-15.

LOG - RECORD ALL ACTIVITY

Use the LOG command to start or stop recording the interactions of an Info/Analysis session. When logging is active, all selections, data entry field input, entry line input, panel names, and output messages are placed in the ISPF log file.

When you begin a session, logging is by default OFF. To start it, you must use the LOG command with the ON operand or no operand. If you subsequently wish to stop logging during the same session, you must use the LOG command with the OFF operand. You may start or stop the logging function on any panel except a tutorial panel.

```
LOG           [ ON ]
              [ OFF]
```

ON

starts logging. ON is the default for the Log command.

OFF

stops logging. No further entries are made in the log file for the current session until logging is specifically set on. Any entries made in the ISPF log file to the point that LOG OFF is issued are kept.

EXAMPLES

1. Stop recording information about your Info/Analysis session.

   LOG OFF

2. Restart recording information about your Info/Analysis session.

   LOG

MASK - OVERLAY DATA

Use the MASK command to mask out (overlay) specific areas in the dump. Those areas may contain sensitive data related to your organization or dumped output of non-IBM programs.

The MASK command is valid only on dump display panels. Once data is masked, it cannot be returned to its original value. If you want to keep both a dump with masked and unmasked data, you must first offload the dump, onload it with a new name, and mask the renamed dump.

```
MASK          {from-addr to-addr   }
              {from-addr FOR length}
```

addr-range
defines the area of dump data that is to be masked. The addresses specified must be within 64K (65,535) bytes of each other. You may specify addr-range in the following forms:

from-addr to-addr
from-address marks the beginning of the area to be masked and to-addr marks the end. The to-addr must be equal to or greater than the from-addr. The address range parameters must be within 64K (65,535) bytes of each other.

from-address FOR length
from-addr marks the beginning of the area to be masked and length defines the length in hexadecimal notation. Masking extends for the number of bytes indicated by the length. The length may not exceed 64K (65,535) bytes.

All addresses are 1 to 4 bytes long representing valid hexadecimal addresses in the dump. Leading zeros are not required.

After checking the syntax of the command, Info/Analysis displays the verification panel shown in Figure 17-1 on page 17-9. This panel requests that you verify the range before the masking process is given control. To confirm the masking, enter a 'y' in the response field and press the ENTER key. To cancel the masking, you must enter an 'n' and press the ENTER key.

If you confirm the masking, Info/Analysis sets all data in that area to equal signs (=). The panel from which you requested masking is redisplayed with a message confirming that the masking has been done. If your reply is negative, masking is canceled and the panel from which you requested masking is displayed with a message stating that the MASK command was cancelled.

```
BLNMA001          MASK VERIFICATION
===> _

DUMP NAME .... SYSDUMP.F3.00000010



Verify that data from 1234C3 to 1236E9 is to be masked:


001234C0  E2E8E27E 7E7E7E7E 7E7E7E7E 7E7E7E7E   *SYS=============*
001234D0  7E7E7E7E 7E7E7E7E 7E7E7E7E 7E7E7E7E   *================*

THROUGH

001236D0  7E7E7E7E 7E7E7E7E 7E7E7E7E 7E7E7E7E   *================*
001236E0  7E7E7E7E 7E7E7E7E 7E7E0000 00000000   *==========......*




      Type 'y' to mask or 'n' to cancel and press ENTER:

      MASK ===>
```

Figure 17-1. Mask Verification Panel

Figure 17-1 is the mask verification panel shown in response to the following masking request:

    MASK 1234c3 for 226

The panel shows the data to be masked along with some data that precedes and follows it. If the address range specified spans data that is not in the dump, the verification panel displays as much of the data as found along with the message "REMAINDER OF DUMP DATA NOT AVAILABLE."

EXAMPLES

1. Mask 100 bytes (256 decimal bytes) of data beginning at address 1F7643.

    mask 1F7643 for 100

2. Mask the data starting at 286931 and ending at address 29F364.

```
mask 286931 29f364
```

MSG - DISPLAY ADDITIONAL MESSAGES PANEL

Use the MSG command to display a panel that contains multiple messages that have resulted from one interaction. When Info/Analysis issues more than one message, it issues another message that instructs you to enter the MSG command.

```
            MSG
```

When the additional messages panel is displayed, you may type 's' in front of any message and press the ENTER key to display the tutorial panel for the message. If you enter more than one 's' in one interaction, the tutorial for the first message is displayed. Enter the END command to see the next message tutorial.

When you enter END on the panel for the last message you selected, you return to the additional messages panel. To return to the panel displayed before the MSG command was entered, enter the END command.

PRINT - OBTAIN PRINTED COPY

Use the PRint command to obtain printed output of the current function or screen contents. In batch mode the output can be routed to a printer or to a local data set using appropriate ASSGN statements in the JCL. In interactive mode the output is routed to the ISPF LIST data set.

```
PRint          ┌                      ┐
               |from-addr             |
               |from-addr to-addr     |
               |from-addr END         |
               |from-addr FOR length  |
               |FORMAT                |
               |DATA                  |
               └                      ┘
```

When you enter PRint without operands, the current screen contents are printed. Each screen must be displayed before it is printed.

This is the ISPF print function that prints a "snapshot" of the current screen image. Without operands, PRint may be issued on any panel.

The PRint operands are optional and mutually exclusive. The addr-range and FORMAT operands are valid only on dump display panels. The DATA operand is valid on the dump management, dump symptoms, dump display, analysis summary, and analysis routines panels.

addr-range
defines an area of dump data to be printed. The output is in traditional hexadecimal format with EBCDIC translation. The specified addresses must be located between the start and end of the dump. You may specify addr-range in the following forms:

from-addr
from-addr marks the beginning of the area to be printed. The print starts at this address and continues for 8192 bytes or until the end of the data, whichever comes first.

from-addr to-addr
from-addr marks the beginning of the area to be printed and to-addr marks the end. The to-addr must be equal to or greater than the from-addr.

from-addr END
from-addr marks the beginning of the area to be printed and END is a character string indicating that the end of the data to be printed is the maximum storage address for the dump.

from-addr FOR length
from-addr marks the beginning and length represents the length of the area to be printed in hexadecimal notation.

All addresses are 1 to 4 bytes long representing valid hexadecimal addresses in the dump. Leading zeros are not required.

FORMAT
causes dump data to be printed with correlated field names and other identifiers. The data printed is determined by information in section six of the symptom record. See Chapter 15, "Dump Viewing" on page 15-1 for a description of formatted dump display and analysis summary data.

DATA
causes the output of the current function to be printed. This output is defined as the current screen plus any additional data that may be displayed using scroll commands.

If issued from a hexadecimal dump display, 4096 bytes of data are printed starting at the BASE address. Note that if you have scrolled more than 4096 bytes from the BASE address, the data you are currently viewing will not be included in the printed data. To print the data you are currently viewing, use Print without operand to print the current screen or use Print with an address range.

DATA enables you to print several related screens with one interaction. For example, when a dump symptoms display spans more than one screen, entering "pr data" prints all the dump symptoms at once. The DATA operand is valid on the dump management, dump symptoms, dump display, analysis summary, and analysis routines panels.

EXAMPLES

1. Print the dump symptom record (when at the Dump Symptoms level).

   ```
   pr data
   ```

2. Print dump data with corresponding field names (when viewing dump data at the dump display level of Dump Viewing).

   ```
   pr format
   ```

3. Print the dump data beginning at address 847765 (when viewing dump data at the dump display level of Dump Viewing).

   ```
   pr 847765 end
   ```

4. Print the dump data between addresses 6479FD and 98675D (when viewing dump data at the dump display level of Dump Viewing).

   ```
   pr 6479fd 98675d
   ```

5. Print 2946 hexadecimal bytes of dump data beginning at address 633014 (when viewing dump data at the dump display level of Dump Viewing).

   ```
   pr 633014 for 2946
   ```

6. Print 8192 bytes (the default length) of dump data beginning at address 34687 (when viewing dump data at the dump display level of Dump Viewing).

   ```
   pr 34687
   ```

RETURN - GO TO FUNCTION SELECTION LEVEL

Use the RETURN command to end the current function and display the function selection panel.

```
RETURN
```

RETURN simulates repeated END commands without displaying intermediate panels.

When entered on a tutorial panel, RETURN ends the tutorial and causes the redisplay of the panel from which you issued the HELP command.

UP, DOWN - SCROLL IN THE SPECIFIED DIRECTION

Use the UP and DOWN commands to display another section of data by moving the screen window in the specified direction. UP and DOWN scroll commands are valid only when the data exceeds a single screen length.

```
                ┌          ┐
UP              | {amount} |
DOWN            | {MAX    }|
                | {HALF   }|
                | {PAGE   }|
                | {CSR    }|
                └          ┘
```

The scrolling commands are defined as follows:

- UP - The screen window moves toward the top of the data by the number of lines specified in the optional parameter or the SCROLL field.
- DOWN - The screen window moves toward the bottom of the data by the number of lines specified in the optional parameter or the SCROLL field.

The operands represent valid scroll amounts. If you do not wish to use the default value shown in the SCROLL field, you may enter:

- A number from 1 to 9999 - specifies the number of lines (up or down) to be scrolled.
- PAGE - specifies scrolling by one page, the amount of information currently visible on the screen.
- HALF - specifies scrolling by a half page, half the amount of information currently visible on the screen.
- MAX - specifies scrolling to the top or bottom of the displayed item, depending on which scrolling command is used.
- CSR - specifies scrolling based on the current position of the cursor. The line or column indicated by the cursor is moved to the top or bottom of the screen, depending on which scrolling command is used. If the cursor is not in the body of the panel, or if it is already positioned at the top, bottom, a PAGE scroll occurs.

Note: When scrolling dump data, a 'DATA NOT AVAILABLE' message may occur because you reached an address area which is not available in the dump. Increase the scroll amount to skip this area.

COMMAND SUMMARY

This is a summary of the interactive commands for Info/Analysis. These commands are presented in alphabetical order. The "Valid Functions" column represents the functions during which the commands may be entered as follows:

M - Dump Management
S - Dump Symptoms
V - Dump Viewing
OF - Dump Offload
ON - Dump Onload
SEL - Selection level
T - Tutorial

COMMAND	DESCRIPTION	M	S	V	OF	ON	SEL	T
DOWN	Scroll down	X	X	X				
END	End function or session	X	X	X	X	X	X	X
Find {'hexdata' } [from-addr] {chardata } [from-addr to-addr] [from-addr END] [from-addr FOR length] [INCRement incr]	Locate data in display	X	X	X				
HELP	Display help	X	X	X	X	X	X	X
Log [ON] [OFF]	Record session interactions	X	X	X	X	X	X	
MASK [from-addr to-addr] [from-addr FOR length]	Overlay data			X				
MSG	Display messages	X	X	X	X	X	X	
PRint [from-addr to-addr]	Print data			X				
[from-addr END]				X				
[from-addr]				X				
[from-addr FOR length]				X				
[FORMAT]				X				
[DATA]		X	X					
RETURN	Return to selection level or panel from which tutorial was called	X	X	X	X	X	X	X
UP	Scroll up	X	X	X				

CHAPTER 18. EXAMPLE INTERACTIVE SESSIONS

The two examples of Info/Analysis interactive sessions are intended to acquaint you with the panel sequence and to illustrate the tasks that you may perform with Info/Analysis.

The step-by-step presentation of the user and system interactions during an Info/Analysis session is meant to assist you in understanding what happens when you use Info/Analysis in interactive mode.

The examples do not include the whole range of Info/Analysis functions. They only provide the data to introduce you to Info/Analysis processing and methods you can apply. In some cases, you may prefer an approach that is different from the one chosen here.

The format enables you to associate each step with the results of data, selections, and commands you enter. The dump data presented on the panels is for illustration purposes only. Naturally, the data displayed when you perform the same steps is dependent on the contents of your dump.

The format of each example is as follows:

First, the task is defined, then the user and system interactions are described. Explanations of the interactions are shown on the left; the sequence of panels on the right. The response is provided in the explanation on the left and also on the appropriate lines of the panel. User entries are shown in bold type in the explanations and on the panels.

PRINTING DUMP SYMPTOMS

Your data processing system has encountered an error. You wish to print a copy of the dump symptoms to compare them with a file of local problems.

1. To invoke Info/Analysis, enter the name of the Info/Analysis start-up command procedure.

2. To list the dumps identified to Info/Analysis, select Dump Management by entering: 1 and pressing ENTER

```
BLNFS001                          INFO/ANALYSIS FUNCTION SELECTION
===> 1

DUMP NAME ===>

Type the dump name in the field above, or select a function
by typing its number, and press ENTER:


1  DUMP MANAGEMENT      Select, add, or delete a dump

2  DUMP SYMPTOMS        Display dump symptoms

3  DUMP VIEWING         Examine a dump

4  DUMP OFFLOAD         Copy a dump to tape

5  DUMP ONLOAD          Load a dump from tape

T  TUTORIAL             Learn how to use Info/Analysis

X  EXIT                 End the Info/Analysis session
```

3. To select the dump you wish to process, enter on the line containing the name of the dump: s and pressing ENTER

```
BLNDM001                         DUMP MANAGEMENT
===> _                                              SCROLL ===> PAGE

DUMP NAME ===>

To select a dump, type 's' before its name or type its name above.
To add a dump, type its name above.
To delete a dump, type 'd' before its name.
Press ENTER:

         DUMP NAME              ONLINE   DATE/TIME TAKEN     VOLID
  SYSDUMP.F8.09004016                     TO BE ONLOADED
s SYSDUMP.F4.P6093403             Y     84/04/12 08:44:51
  SYSDUMP.F6.S0033119             Y     84/04/12 08:11:06
  SYSDUMP.F4.S0015837             Y     84/03/28 22:05:16   T02818
  SYSDUMP.F4.S0007812                   84/03/17 01:31:21   T42901
  SYSDUMP.F5.00003132                   84/03/08 15:38:42   T03496
```

4. The name of the dump you selected is copied to the DUMP NAME field. It is now the current dump. Enter the end command to leave Dump Management.

```
BLNDM001                       DUMP MANAGEMENT
===> end                                               SCROLL ===> PAGE

DUMP NAME ===> SYSDUMP.F4.P6093403

To select a dump, type 's' before its name or type its name above.
To add a dump, type its name above.
To delete a dump, type 'd' before its name.
Press ENTER:

          DUMP NAME                ONLINE   DATE/TIME TAKEN     VOLID
  SYSDUMP.F8.09004016                        TO BE ONLOADED
  SYSDUMP.F4.P6093403                Y     84/04/12 08:44:51
  SYSDUMP.F6.S0033119                Y     84/04/12 08:11:06
  SYSDUMP.F4.S0015837                Y     84/03/28 22:05:16    T02818
  SYSDUMP.F4.S0007812                      84/03/17 01:31:21    T42901
  SYSDUMP.F5.00003132                      84/03/08 15:38:42    T03496
```

5. Select Dump Symptoms by entering: 2 and pressing ENTER

```
BLNFS001                       INFO/ANALYSIS FUNCTION SELECTION
===> 2

DUMP NAME ===> SYSDUMP.F4.P6093403

Type the dump name in the field above, or select a function
by typing its number, and press ENTER:


1  DUMP MANAGEMENT      Select, add, or delete a dump

2  DUMP SYMPTOMS        Display dump symptoms

3  DUMP VIEWING         Examine a dump

4  DUMP OFFLOAD         Copy a dump to tape

5  DUMP ONLOAD          Load a dump from tape

T  TUTORIAL             Learn how to use Info/Analysis

X  EXIT                 End the Info/Analysis session
```

6. The dump symptoms panel is displayed. In this case, the symptoms do not fit on the screen. Enter the down command to look at the rest of the symptoms.

```
BLNDS001                  DUMP SYMPTOMS
===> down                                     SCROLL ===> PAGE

DUMP NAME .... SYSDUMP.F4.P6093403


ENVIRONMENT:
   CPU MODEL ....... 4331
   CPU SERIAL ...... 060707
   TIME ............ 08:44:51:42
   DATE ............ 84/04/12
   SYSTEM ID ....... 574500000
   RELEASE ......... 4
   FEATURE ......... 02
   DUMPTYPE ........ SCPREQ
   PROBLEM NUMBER ..

REQUIRED SYMPTOMS:
   AB/S0234
   PIDS/5745SCBTM
   RIDS/DFHTUB
   REGS/0C14E
   REGS/0A050
```

7. The remaining symptoms are displayed for your review. To print all the symptoms, enter: print data

```
BLNDS001                  DUMP SYMPTOMS
===> print data                               SCROLL ===> PAGE

DUMP NAME .... SYSDUMP.F4.P6093403


OPTIONAL SYMPTOMS (SDB):
   ADRS/0001A694
   LVLS/011
   +FLDS/ADBMRS

OPTIONAL SYMPTOMS (NON-SDB):
   DSN=ABCDE.R0155.FLON
   +STEP=TERMINATION
```

8. A message is displayed informing you that the dump symptoms are printed. To leave Info/Analysis, enter: return;x

```
BLNDS001                    DUMP SYMPTOMS
===> return;x                                       SCROLL ===> PAGE
BLN9012I PRINT FUNCTION COMPLETED
DUMP NAME .... SYSDUMP.F4.P6093403


OPTIONAL SYMPTOMS (SDB):
   ADRS/0001A694
   LVLS/011
   +FLDS/ADBMRS

OPTIONAL SYMPTOMS (NON-SDB):
   DSN=ABCDE.R0155.FLON
   +STEP=TERMINATION
```

ANALYZING A DUMP

You have just completed an offload of dump SYSDUMP.F8.S0093411 to volume 143783. The dump offload panel is displayed with a confirmation message.

1. To proceed directly to Dump Management, enter: return;1

```
BLNDF001                         DUMP OFFLOAD
===> return;1
BLN9040I JOB INFOSBT SUBMITTED TO BATCH
DUMP NAME .... SYSDUMP.F8.S0093411

Enter data for dump offload and press ENTER to submit to batch:

  VOLID   ===> 143783          Type the output volume identifier

  ERASE   ===> Y               Erase system copy of dump

  BYPASS  ===> N               Prevent copying dump offline if valid copy exists

  SYSNUM  ===>                 Type the tape symbolic unit name

  TAPADR  ===>                 Type the tape channel and unit address

  TAPMDE  ===>                 Type the tape mode setting


CHANGE DEFAULT VALUES IF REQUIRED


===> * $$ JOB JNM=INFOSBT,CLASS=A,DISP=D
===> * $$ LST CLASS=A,DISP=D
===>
```

2. Create and enter the dump name sysdump.f8.p6106143 by typing it in the DUMP NAME field and pressing the ENTER key.

```
BLNDM001                         DUMP MANAGEMENT
===>                                                 SCROLL ===> PAGE

DUMP NAME ===> sysdump.f8.p6106143

To select a dump, type 's' before its name or type its name above.
To add a dump, type its name above.
To delete a dump, type 'd' before its name.
Press ENTER:

           DUMP NAME               ONLINE   DATE/TIME TAKEN      VOLID
  SYSDUMP.F8.09004016                        TO BE ONLOADED
  SYSDUMP.F8.S0093411                       84/04/12 08:44:51    143783
  SYSDUMP.F6.S0033119                Y      84/04/12 08:11:06
  SYSDUMP.F4.S0015837                Y      84/03/28 22:05:16    T02818
  SYSDUMP.F4.S0007812                       84/03/17 01:31:21    T42901
  SYSDUMP.F5.00003132                       84/03/08 15:38:42    T03496
```

3. The dump name with associated information is added to the dump management file and appears in the list. To process the dump, you must onload it. To go to Dump Onload, enter: end;5

```
BLNDM001                        DUMP MANAGEMENT
===> end;5                                              SCROLL ===> PAGE
BLN9018I DUMP SYSDUMP.F8.P6106143 ADDED
DUMP NAME ===> SYSDUMP.F8.P6106143

To select a dump, type 's' before its name or type its name above.
To add a dump, type its name above.
To delete a dump, type 'd' before its name.
Press ENTER:

          DUMP NAME              ONLINE   DATE/TIME TAKEN      VOLID
  SYSDUMP.F8.P6016143                      TO BE ONLOADED
  SYSDUMP.F8.09004016                      TO BE ONLOADED
  SYSDUMP.F8.S0093411                     84/04/12 08:44:51    143783
  SYSDUMP.F6.S0033119              Y      84/04/12 08:11:06
  SYSDUMP.F4.S0015837              Y      84/03/28 22:05:16    T02818
  SYSDUMP.F4.S0007812                     84/03/17 01:31:21    T42901
  SYSDUMP.F5.00003132                     84/03/08 15:38:42    T03496
```

4. To perform the onload, specify the volume id of the tape on which the dump resides. Enter in the VOLID field: t42901. Because this is not a multiple-dump or standard labeled tape, you need not alter the FILE field.

Press the ENTER key to submit the onload job.

```
BLNDN001                        DUMP ONLOAD
===> _

DUMP NAME .... SYSDUMP.F8.P6106143

Enter data for dump onload and press ENTER to submit to batch:

   VOLID ===> t42901           Type the volume identifier

   FILE  ===> 0001             Type the file sequence number

   SYSNUM  ===>                Type the tape symbolic unit name

   TAPADR  ===>                Type the tape channel and unit address

   TAPMDE  ===>                Type the tape mode setting




CHANGE DEFAULT VALUES IF REQUIRED


===> * $$ JOB JNM=INFOSBT,CLASS=A,DISP=D
===> * $$ LST CLASS=A,DISP=D
===>
```

5. The dump onload panel is redisplayed with a confirmation message. To examine the dump symptoms, enter: end;2 after the onload job has completed execution.	BLNDN001 DUMP ONLOAD ===> end;2 BLN9040I JOB INFOSBT SUBMITTED TO BATCH DUMP NAME .... SYSDUMP.F8.P6106143 Enter data for dump onload and press ENTER to submit to batch: VOLID ===> T42901 Type the volume identifier FILE ===> 0001 Type the file sequence number SYSNUM ===> Type the tape symbolic unit name TAPADR ===> Type the tape channel and unit address TAPMDE ===> Type the tape mode setting CHANGE DEFAULT VALUES IF REQUIRED ===> * $$ JOB JNM=INFOSBT,CLASS=A,DISP=D ===> * $$ LST CLASS=A,DISP=D ===>
THE NEXT STEPS MUST WAIT UNTIL THE ONLOAD BATCH JOB HAS COMPLETED.	

6. You scroll through the symptoms using PF keys. Because the symptoms don't pinpoint the failure, you go to the analysis routines panel. Enter: end;3;4

```
BLNDS001                  DUMP SYMPTOMS
===> end;3;4                                    SCROLL ===> PAGE

DUMP NAME .... SYSDUMP.F8.P6106143


ENVIRONMENT:
   CPU MODEL ....... 4331
   CPU SERIAL ...... 089432
   TIME ............ 10:52:40:03
   DATE ............ 84/04/13
   SYSTEM ID ....... 645321000
   RELEASE ......... 4
   FEATURE ......... 02
   DUMPTYPE ........ SADUMP
   PROBLEM NUMBER ..

REQUIRED SYMPTOMS:
   AB/S0234
   PIDS/5745SCBTM
   RIDS/DFHTUB
   REGS/0C14E
   REGS/0A050
```

7. From the list, you select a routine for execution. Enter: s

```
BLNAR001                  ANALYSIS ROUTINES
===> _                                          SCROLL ===> PAGE

DUMP NAME .... SYSDUMP.F8.P6106143

To select a routine for execution, type 's' before its name
and press ENTER:

  ROUTINE NAME          DESCRIPTION
s IJBXDBUG          Analyze Stand-alone dump
```

8. The analysis routine is executed. Enter the END command to leave the analysis routines function and return to the dump viewing selection level.

```
BLNAR001                    ANALYSIS ROUTINES
===> end                                        SCROLL ===> PAGE

DUMP NAME .... SYSDUMP.F8.P6106143

To select a routine for execution, type 's' before its name
and press ENTER:

  ROUTINE NAME          DESCRIPTION
  IJBXDBUG         Analyze Stand-alone dump
```

9. The dump viewing selection panel is displayed. To look at the results of the analysis routine you executed, select the analysis summary function. Enter: 3

```
BLNDVS01             DUMP VIEWING SELECTION
===> 3

DUMP NAME .... SYSDUMP.FB.P6106143

To select a function, type its number and press ENTER:

1  DUMP DISPLAY OPTIONS       Examine or change dump display options

2  DUMP DISPLAY               Examine dump data

3  ANALYSIS SUMMARY           Examine text, control block locations,
                              control block linkages, etc.

4  ANALYSIS ROUTINES          Select an analysis routine for execution
```

10. On the main display of analysis summary, you select the new entry created by the analysis routine. Enter on the appropriate line: s

```
BLNAS001                           ANALYSIS SUMMARY
===> _                                                  SCROLL ===> PAGE

DUMP NAME .... SYSDUMP.F8.P6101143

To select an item for display, type 's' before its name and
press ENTER:

   BLOCK         COMP            TYPE
  BDDTEXT1     VSE/AF      TEXT DATA
  BDDTEXT2     VSE/AF      TEXT DATA
  BDDTEXT3     VSE/AF      TEXT DATA
  BDDBASE      VSE/AF      LINKAGE
  REGS         VSE/AF      HEXADECIMAL DATA
  BDDRSVT      VSE/AF      LINKAGE
  BDDCSVT      VSE/AF      LINKAGE
  BDDSCVT      VSE/AF      LINKAGE
  BDDLIBR      VSE/AF      LINKAGE
s IVBXBDD      IVBXPWRF   +TEXT DATA
  LOCATORS                 CONTROL BLOCK LIST
```

11. An analysis summary panel for text data is displayed. The text presents some comments about the error. Because of the reference to SCRLINE in the text, you decide to display that control block. To return to the analysis summary main display, enter: end.

```
BLNAS002                           ANALYSIS SUMMARY
===> end                                                SCROLL ===> PAGE

DUMP NAME .... SYSDUMP.F8.P6106143
DISPLAY .... IVBXBDD

  ERROR OCCURRED DURING A SCREEN UPDATE.
  ALL SCREEN LINE ENTRIES WERE ALLOCATED SUCCESSFULLY.
  ROUTINE BDDSCRBC DETECTED DATA ERROR WHILE INSERTING DATA
      INTO LINE BUFFER.
  SCRLINE CONTROL BLOCK AT 001298C0.
  ATTEMPT TO RECOVER HAS FAILED FOR THAT LINE.
```

12. From the main analysis summary panel, you select the BDDSCVT linkage entry. You know the SCRLINE is part of that linkage. Enter on the appropriate line: s

```
BLNAS001                        ANALYSIS SUMMARY
===> _                                                  SCROLL ===> PAGE

DUMP NAME .... SYSDUMP.F8.P6106143

To select an item for display, type 's' before its name and
press ENTER:

   BLOCK         COMP            TYPE
  BDDTEXT1     VSE/AF         TEXT DATA
  BDDTEXT2     VSE/AF         TEXT DATA
  BDDTEXT3     VSE/AF         TEXT DATA
  BDDBASE      VSE/AF         LINKAGE
  REGS         VSE/AF         HEXADECIMAL DATA
  BDDRSVT      VSE/AF         LINKAGE
  BDDCSVT      VSE/AF         LINKAGE
s BDDSCVT      VSE/AF         LINKAGE
  BDDLIBR      VSE/AF         LINKAGE
  IVBXBDD      IVBXPWRT      +TEXT DATA
  LOCATORS                    CONTROL BLOCK LIST
```

13. The resultant panel shows the sequence of connected control blocks anchored by the SCVT. You scroll through the display by pressing PF keys. The keyfield contents '05' for the SCRLINE arouse your interest. To view this item in detail, enter on the appropriate line: s

```
BLNAS003                        ANALYSIS SUMMARY
===> _                                                  SCROLL ===> PAGE

DUMP NAME .... SYSDUMP.F8.P6106143
DISPLAY .... BDDSCVT

    BLOCK        COMP      MODE  QUAL   BASE(ADDR)       KEYFIELD
   SCVT        VSE/AF      V    0001   00122320   SCVLBLT   = D'0'
    SCLIBHDR   VSE/AF      V    0001   001212F0   DATA NOT FOUND
   SCVT        VSE/AF      V    0001   00122470   SCVLBLT   = D'9'
    SCLIBHDR   VSE/AF      V    0001   00122590   SCLNUM    = 6
     SCRLINE   VSE/AF      V    0001   00129C40   SCRLNERC  = 00
     SCRLINE   VSE/AF      V    0001   00129BD0   SCRLNERC  = 00
     SCRLINE   VSE/AF      V    0001   00129B60   SCRLNERC  = 00
     SCRLINE   VSE/AF      V    0001   00129AF0   SCRLNERC  = 00
     SCRLINE   VSE/AF      V    0001   00129A80   SCRLNERC  = 00
     SCRLINE   VSE/AF      V    0001   00129A10   SCRLNERC  = 00
   SCVT        VSE/AF      V    0001   001227E0   SCVLBLT   = D'12'
    SCLIBHDR   VSE/AF      V    0001   00122770   SCLNUM    = 8
     SCRLINE   CICS         V    0001   001299A0   SCRLNERC  = 00
     SCRLINE   CICS         V    0001   00129930   SCRLNERC  = 00
s    SCRLINE   CICS         V    0007   001298C0   SCRLNERC  = 05
     SCRLINE   CICS         V    0001   00129850   SCRLNERC  = 00
```

14. A dump display panel showing the SCRLINE control block you selected is displayed. Because the preference for formatted display is recorded in your user profile, the panel shown is in formatted mode. At this point, you may use cursor positioning, the data entry fields, and the Find command to look at dump data. When you are finished, enter a chain of END commands to return to the dump viewing selection panel: end;end;end

```
BLNDD001                          DUMP DISPLAY
===> end;end;end                                        SCROLL ===> PAGE

DUMP NAME .... SYSDUMP.F8.P6106143
DISPLAY ==> SCRLINE   BASE ==> 00298C0    QUAL ==> 0001   MODE ==> V

SCRIDC     C'SCRL'    SCRFWDP   00129850   SCRBWDP   00129930
SCREXTC    0001       SCREXTT   0001       SCRDMEFP  00129600
SCRDMEBP   00129520   SCRDMEUD  D'15'      SCRDMEBT  B'00100011'
SCRDMERC   05         SCRDMEML  0070       SCRDMECL  004E
SCRDMETX   C' 4  SYS003  TIME  00000000 0012A3C0      '
```

15. Deciding that you want to look again at the last item you displayed, you select the dump display function. Enter: 2

```
BLNDVS01             DUMP VIEWING SELECTION
===> 2

DUMP NAME .... SYSDUMP.FB.P6106143

To select a function, type its number and press ENTER:

1  DUMP DISPLAY OPTIONS     Examine or change dump display options

2  DUMP DISPLAY             Examine dump data

3  ANALYSIS SUMMARY         Examine text, control block locations,
                            control block linkages, etc.

4  ANALYSIS ROUTINES        Select an analysis routine for execution
```

<table>
<tr>
<td>16. Info/Analysis uses the DISPLAY, BASE, QUAL, and MODE values it has saved from the last dump display to present the panel. After looking at the data, you end the Info/Analysis session by entering: return;x</td>
<td>

```
BLNDD001                          DUMP DISPLAY
===> return;x                                            SCROLL ===> PAGE

DUMP NAME .... SYSDUMP.F8.P6106143
DISPLAY ==> SCRLINE     BASE ==> 1298C0     QUAL ==> 0001   MODE ==> V

SCRIDC     C'SCRL'      SCRFWDP   00129850   SCRBWDP   00129930
SCREXTC    0001         SCREXTT   0001       SCRDMEFP  00129600
SCRDMEBP   00129520     SCRDMEUD  D'15'      SCRDMEBT  B'00100011'
SCRDMERC   05           SCRDMEML  0070       SCRDMECL  004E
SCRDMETX   C' 4  SYS003  TIME  00000000 0012A3C0       '
```

</td>
</tr>
</table>

CHAPTER 19. INFO/ANALYSIS IN BATCH MODE

You may use Info/Analysis in the batch environment. All of the major functions available to the interactive user are available to the batch user. However, the dump viewing function is more flexible in the interactive environment. To familiarize yourself with the functions of Info/Analysis, refer to the Chapters 14 to 17 describing Dump Management, Dump Symptoms, Dump Viewing, and Dump Offload and Onload. Please note, that for most of the batch functions, examples can be found under Chapter 3, "Printing the Stored Dump" on page 3-1.

You may use Info/Analysis in batch mode in two ways:

- Line mode - from the operator console or ICCF console
- Reader mode - from the system input device or ICCF virtual reader

In either case, you invoke Info/Analysis by submitting a series of job control statements (JCL) followed by control statements that request Info/Analysis functions. All output is routed to the SYSLST device or the ICCF virtual printer called the print area. The output includes the input control statements, the results of processing, and any messages issued by Info/Analysis. If you are working at a console, Info/Analysis also routes messages there.

This chapter describes:

- How to invoke Info/Analysis in batch mode.
- Syntax rules for control statements.
- Each Info/Analysis function and the control statements needed to request it.
- How to end Info/Analysis in batch mode.

To illustrate this information, the chapter includes example sequences of control statements.

Contents

INVOKING INFO/ANALYSIS IN BATCH MODE

You invoke Info/Analysis by submitting the necessary JCL followed by control statements that request functions. You may submit the job either in line mode by entering statements on the console, or in reader mode by submitting a job to the system input device.

With JCL, you must specify any nonstandard system device assignments and preallocate and assign any files that you require other than the system libraries. A sample of the JCL for invocation is:

```
// JOB    ANALYSIS
// ASSGN  SYSLST,00E
// ASSGN  SYS016,252    " DUMP MANAGEMENT FILE "
// ASSGN  SYS017,252    " EXTERNAL ROUTINE FILE "
// EXEC   INFOANA,SIZE=400K

(Info/Analysis control statements)

/*
/&
```

Figure 19-1. Sample Job: Invoke Info/Analysis

Once the JCL has been processed, you are at the selection level in Info/Analysis. The program reads for your control statements. An end of input (/*) statement marks the end of these statements. To end your job, enter an end of job (/&) statement. VSE/Advanced Functions System Control Statements contains further information about the JCL statements.

CONTROL STATEMENT SYNTAX

You operate Info/Analysis in batch mode by entering control statements. These control statements specify the major functions you wish to perform (Dump Management, Dump Viewing, etc.) and the information necessary to perform each function. This section describes the syntax rules for entering the control statements and the conventions used in this publication to describe the statements.

Entering Control Statements

The rules for entering the control statements that request Info/Analysis functions are:

- Each card or input line may contain only one control statement.
- A control statement may begin in any column.
- Control statements and their operands may be entered in uppercase or lowercase.
- Control statements must be entered in their complete form; no abbreviations are allowed.
- Each word in a control statement must be a contiguous string of characters.
- Some blanks, at least one, must appear between the words in a statement.
- A blank followed by an asterisk (' *') signifies the start of a comment. If the first non-blank character in any control statement is an asterisk, the entire statement is treated as a comment.

If you enter an invalid control statement in reader mode, the remaining control statements in the job are flushed and the session is canceled. The output indicates the erroneous statement. You should correct the statement and resubmit the job. In line mode, invalid input causes Info/Analysis to flush the control statement. You may then reenter the statement correctly.

Syntax Conventions

The following conventions are used:

- Operands shown in braces { } represent alternatives; you must choose one.
- Operands shown in brackets [ ] are optional; you may choose one or none.
- An underscored operand represents the default assumed if no operand is chosen.
- Operands shown in lowercase are variables; you substitute a value for each.
- Control statements and operands may be entered only in their complete form, as shown. Abbreviations are not allowed.

COMMON CONTROL STATEMENTS

Each function has a set of control statements appropriate to it. In this chapter, the control statement descriptions are presented by function. The same control statement may have different uses within different functions. Therefore statements such as PRINT are described more than once. A summary of the control statements is given under "Control Statement Summary" on page 19-20.

The following control statements are common to all Info/Analysis functions and may also be entered at the selection level; that is, when Info/Analysis is initialized and no function is currently selected.

- **SELECT** - invoke a function or end the Info/Analysis session
- **HELP** - send a short tutorial to the output device
- **RETURN** - end the current function; return to the selection level
- **DUMP NAME** - specifies the name of the dump to be processed.

SELECT - Specify a Function or End Info/Analysis

To perform any function, you must first select it. Use the SELECT statement to specify the function you wish to perform or to end your Info/Analysis session.

```
      SELECT            {DUMP MANAGEMENT}
                        {DUMP SYMPTOMS  }
                        {DUMP VIEWING   }
                        {DUMP OFFLOAD   }
                        {DUMP ONLOAD    }
                        {END            }
```

The functions you may specify on the SELECT statement are the same presented on the function selection panel to an interactive user. You enter a SELECT statement as follows:

```
SELECT DUMP ONLOAD
```

Once you have specified a SELECT statement, you are operating at the function level. All subsequent statements apply to the selected function until you enter a RETURN statement.

Use SELECT END to conclude your Info/Analysis session. If you end your control statement sequence with an end of input (/*) and end of job (/&) statement without first specifying a SELECT END or RETURN statement, Info/Analysis ends the job as though they had been included.

RETURN - End Current Function

Use the RETURN statement to end the current function. Each function you use must be requested by a SELECT statement and ended by a RETURN statement, before you can select another function.

```
RETURN
```

The RETURN statement has no operands. RETURN brings you back to the selection level where you may select another function or end your session. In the following sequence, RETURN ends the Dump Management function:

```
DUMP NAME SYSDUMP.F6.0000013
SELECT DUMP MANAGEMENT
PRINT DATA
RETURN
SELECT END
```

You must enter RETURN to complete one function before selecting another. If you end your control statement sequence with an end of input (/*) and end of job (/&) statement without first specifying a RETURN and/or SELECT END statement, Info/Analysis ends the job as if they had been included.

HELP - Print Tutorial

Use the HELP statement to print the Info/Analysis batch tutorial. The tutorial consists of texts that describe the control statements you may enter.

```
HELP
```

The HELP statement has no operands. You may include HELP at any point in a sequence of control statements except after a SELECT END statement. The tutorial is included in the output from your session.

An example help job is shown in

----> Figure 3-2 on page 3-4.

DUMP NAME - Specify or Add Current Dump

Specify the name of the dump you wish to process by entering the DUMP NAME statement. You must specify a dump before you perform any Info/Analysis function except the Dump Management UTILITY or PRINT functions and the HELP function.

```
DUMP NAME     dumpname
```

"dumpname" is a variable representing the dump name. The dump you specify on a dump name statement is considered to be the current dump; that is, all subsequently selected functions process that dump until you enter another dump name or until it has been deleted.

When you specify a dump, Info/Analysis searches the dump management file for the name. If it finds the name, the dump is made current.

If the dump you specify is not identified in the dump management file and does not reside in a dump sublibrary, Info/Analysis adds the dump name and the information "TO BE ONLOADED" to the file. A dump must reside in one of the dump sublibraries before you can act on it using any Info/Analysis function other than Dump Onload.

You may specify the DUMP NAME statement at the selection level. In other words, the DUMP NAME statement may be placed:

- Immediately preceding a SELECT statement, as follows:

```
DUMP NAME SYSDUMP.F3.00000010
SELECT DUMP MANAGEMENT
PRINT DATA
RETURN
```

- After the SELECT DUMP MANAGEMENT statement and before the next RETURN statement, as follows:

```
SELECT DUMP MANAGEMENT
DUMP NAME SYSDUMP.F3.00000010
PRINT DATA
RETURN
```

DUMP MANAGEMENT

The Dump Management function enables you to manage your dump data sets by manipulating the contents of the dump management file. This file contains descriptive information about the dumps being managed by Info/Analysis. You can use this information to keep track of the dumps on your system, those you have offloaded, and those you plan to onload. For further information about Dump Management, refer to

------> Chapter 13, "Dump Management" on page 13-1.

To initiate Dump Management, use the following statement:

    SELECT DUMP MANAGEMENT

After this selection, specify the desired combination of the following control statements:

UTILITY - initialize a new or reallocated dump management file to contain the list of dumps managed by Info/Analysis.

DELETE - erase the current dump from the dump sublibrary, if it resides there, and delete information about the dump from the dump management file.

PRINT DATA - print the contents of the dump management file.

Info/Analysis responds by searching the dump sublibraries for dumps that are not yet identified in the dump management file. For each of these dumps, if any, the routine adds identifying information to the file.

UTILITY - Initialize Dump Management File

```
UTILITY
```

The UTILITY statement is intended for the system programmer at your installation who has responsibility for Info/Analysis. UTILITY initializes the dump management file at installation time or reinitializes the file when it is subsequently reallocated with more or less space.

The dump management file is allocated during installation of VSE/Advanced Functions. It is possible for this data set to become full. In VSE, disk data sets are allocated with a fixed size with no dynamic extensions possible. Extensions to the data set must be accomplished by specific allocation of a larger data set.

When the dump management file size has to be increased or decreased, do the following:

1. Dump the contents (if any) of the existing file to tape
2. Allocate a new disk extent that is larger or smaller
3. Restore the old file contents (if any) to the new extent

When these steps are done, the system programmer should invoke Info/Analysis in batch mode and specify the UTILITY statement as follows:

```
SELECT DUMP MANAGEMENT
UTILITY
RETURN
SELECT END
```

UTILITY sets up the control information in the file for use by the dump management function. The control record indicates the number of dumps currently being managed and the maximum number that will fit.

DELETE - Delete Current Dump

```
DELETE
```

Use the DELETE statement to erase the current dump from the library, if it resides there, and to delete the information about the dump from the dump management file.

If you specify DELETE and the current dump does not reside in the library (that is, it has never been onloaded or it has been offloaded), information about the dump is deleted from the dump management file. If a copy of the dump exists on tape, it is your responsibility to dispose that tape.

A delete job example is shown in

-----> Figure 3-9 on page 3-14.

PRINT - Print List of Managed Dumps

Use the PRINT statement to print the contents of the dump management file.

```
PRINT            DATA
```

Output from the PRINT statement is a list of the dumps being managed by Info/Analysis. For each dump, one line of information is printed. An example of a print data job and its output is shown under

-------> "List Names of Dumps Managed by Info/Analysis" on page 3-5.

DUMP SYMPTOMS

The Dump Symptoms function prints the failure information that is contained in the dump symptom record. You may use this information to identify duplicate problems locally and in an IBM maintenance data base. To initiate Dump Symptoms, enter the following statement:

```
SELECT DUMP SYMPTOMS
```

After making this selection, you may print the symptoms of the current dump using the PRINT statement.

PRINT - Print Dump Symptoms

```
PRINT            DATA
```

Use the PRINT statement to print sections 1 through 5 of the symptom record of the current dump. For a description of a symptom records, see

-------> Appendix C, "Symptom Record Overview" on page C-1.

An example of a print data job and its output is shown under

-------> "Print Dump Symptoms" on page 3-17.

DUMP VIEWING

Using Dump Viewing in batch mode, you may print dump data and analysis summary data. You may also call an analysis routine for execution. Unlike interactive users of Dump Viewing, you cannot locate particular data or mask data for security purposes. To initiate Dump Viewing, enter the following statement:

```
SELECT DUMP VIEWING
```

After making this selection, you may perform Dump Viewing functions by specifying the following control statements:

- PRINT - print dump data
- CALL - initiate an analysis routine

PRINT - Print Dump Data

Use the PRINT statement to print dump data or analysis summary data. You must specify either the area of the dump you wish to print in hexadecimal mode or that you wish to print all formatted and all analysis summary data. The operands for the PRINT statement are mutually exclusive.

```
PRINT           { from-addr           }
                { from-addr to-addr   }
                { from-addr END       }
                { from-addr FOR length}
                { FORMAT              }
```

The PRINT control statement for Dump Viewing requires either an address range or the FORMAT operand. The results of PRINT with the addr-range operand are printed in traditional hexadecimal format with EBCDIC translation. The FORMAT operand prints the analysis summary if stored during a previous analysis routine run, and formatted dump display information as described in Chapter 15, "Dump Viewing" on page 15-1." This data includes control blocks, text data, hexadecimal data, and control block linkage descriptors as defined in section 6 of the symptom records. For a description of the symptom records, see

-------> Appendix C, "Symptom Record Overview" on page C-1.

PRINT

from-addr
'from-addr' marks the beginning of the 8192 byte area to be printed.

from-addr to-addr
'from-addr' marks the beginning of the area to be printed and 'to-addr' marks the end.

from-addr END
'from-addr' marks the beginning of the area to be printed and 'END' indicates that the data up to the high address end of the dumped storage is to be printed.

from-addr FOR length
'from-addr' marks the beginning of the area to be printed and 'length' represents the number of bytes in hexadecimal which are to be printed. For example, if you specify 10 as a length, 16 bytes are printed.

All addresses are 1- to 4-character hexadecimal values representing valid addresses in the dump. Leading zeros are not required for an address specification.

FORMAT
causes the data to be printed with correlated field names and other identifiers. The data printed is

determined by information in section 6 of the symptom records.

Find print job examples in

-------> Figure 3-14 on page 3-22 and

-------> Figure 3-15 on page 3-24.

CALL - Initiate Analysis Routine

Use the CALL statement to invoke an analysis routine.

```
CALL              routine name
```

The routine name is required on a CALL statement and must be the name of an executable routine. These routines may be provided by system components or by your installation. It is the responsibility of your installation to maintain an external routines file containing the names of executable routines. Consult your system programmer for the names of these routines.

An call job example is given in

------> Figure 3-10 on page 3-15.

A description of the analysis routine IJBXDBUG can be found under

------> "The Stand-Alone Dump Analysis Routine IJBXDBUG" on page 4-8.

DUMP OFFLOAD

Dump Offload places a dump file that resides on the dump library onto tape for later retrieval. You may choose whether or not to maintain the copy that is on the dump library; the default is to erase the dump. To initiate Dump Offload, use the following statement:

```
SELECT DUMP OFFLOAD
```

After making this selection, you may offload the current dump by specifying any of the following control statements that are necessary:

- VOLID - specify the output volume and the logical unit number

- BYPASS - skip the write-to-tape operation

- ERASE NO - does not delete the library copy of the dump

Dump Offload is valuable if you need to increase the available online space. Dump Offload does not remove the information about the dump from the dump management file.

An example of a dump Offload job is given in

------> Figure 3-7 on page 3-11.

VOLID - Specify Output Volume

Use the VOLID statement to specify the identifier of the output tape.

```
VOLID          volume-id [logical unit number]
```

The volume id is a 6-character alphameric value that is added to the entry for the current dump contained in the dump management file. The VOLID statement is required if the dump you are offloading has not been previously onloaded or offloaded.

For subsequent offloads of the same dump, Info/Analysis can retrieve the volume id from the dump management file. To override the saved volume, use the VOLID statement. The most recent VOLID is the one saved in the dump management file.

The logical unit number (SYSnnn) can be assigned to a physical tape address via the job control 'ASSGN' statement. If you do not define a logical unit number allocation is done automatically by the system. The first available unit will be allocated and a message issued for the tape to be mounted on this unit.

BYPASS - Skip Offload

Use the BYPASS statement to free the library space used by the dump without writing the dump to tape. BYPASS is allowed only if a valid offloaded copy of the dump exists; that is, if both of the following conditions are met:

- The current dump has been previously offloaded, but is still in the library, and

- The current dump has not been modified by an analysis routine since it was last offloaded.

```
BYPASS
```

When BYPASS is processed, Info/Analysis checks for the above conditions. If they are not met, the offload function is not performed.

The DUMP Offload BYPASS statement and the Dump Management DELETE statement differ in the following ways:

- BYPASS checks for a copy of the dump on tape. DELETE does not.
- DELETE removes references to the dump from the dump management file. BYPASS does not.

Thus, use Dump Management with DELETE only if you no longer need a dump. Use Dump Offload with BYPASS if you wish to remove a dump from the dump library but want to maintain a copy on tape and keep information about the dump in the dump management file.

The BYPASS and ERASE NO statements are contradictory and thus mutually exclusive.

ERASE - Delete or Retain Library Copy of Dump

The ERASE statement specifies whether or not Info/Analysis should delete the dump library copy of the dump when doing an offload.

```
ERASE          [ YES ]
               [  NO ]
```

If you want to maintain a copy of the dump on the dump library as well as on tape, specify ERASE NO. ERASE YES is the default. Therefore, if you specify ERASE, ERASE YES, or do not specify the ERASE control statement during Dump Offload, Info/Analysis erases the dump from the dump library after a copy is offloaded to tape.

By specifying ERASE NO, you can offload more than one copy of the dump. For example, you may offload a copy of the dump, then run analysis routines, then offload the modified copy.

BYPASS and ERASE NO are contradictory and thus mutually exclusive statements.

DUMP ONLOAD

Dump Onload copies dumps which reside on tapes into the dump library so that they can be further processed by Info/Analysis. To initiate Dump Onload, use the following statement:

```
SELECT DUMP ONLOAD
```

After making this selection, you may onload the current dump by entering the VOLID and FILE control statements if necessary.

An example of a dump Onload job is given in

------> Figure 3-6 on page 3-9.

VOLID - Specify Input Volume

Use the VOLID statement to specify the volume identifier and (optional) the logical unit number of the tape on which the current dump resides.

```
VOLID            volume id [logical unit number]
```

The volume id is a 6-character alphameric value that is added to the entry for the current dump contained in the dump management file. The VOLID statement is required if you are onloading a dump for the first time.

For subsequent offloads and onloads of the dump, Info/Analysis retrieves the volume id from the dump management file. To override the saved value, use the VOLID statement. The most recent VOLID is the one saved in the dump management file.

The logical unit number (SYSnnn) can be assigned to a physical tape address via the job control 'ASSGN' statement. If you do not define a logical unit number allocation is done automatically by the system. The first available unit will be allocated and a message issued for the tape to be mounted on this unit.

FILE - Specify Dump on Multiple-Dump Tape

If the tape you are using contains more than one dump, use the FILE statement to specify the specific dump file you want to onload. In this way, you may onload more than one dump from a tape during a session. Keep in mind that you must leave Dump Onload and specify another dump name before onloading the next file.

```
FILE          file number  [ LAST ]
```

The default for the FILE statement is "1" if the file statement is omitted. Therefore, if you are onloading a dump from a single file tape or if you are onloading the first file from a multiple file tape, you need not specify the FILE statement. Also, for Stand-alone dumps the FILE statement need not be specified.

The file number must designate an existing file on the input tape. This sequence number is used during the positioning of the tape by Dump Onload when the input tape file is opened.

When onloading multiple dumps during an Info/Analysis session, their file numbers do not have to be in ascending order.

The LAST parameter indicates that this is the last file to be onloaded from the current volume. Specifying LAST deallocates the tape drive from Info/Analysis.

ENDING THE INFO/ANALYSIS JOB

You end an Info/Analysis batch job by submitting the SELECT END statement while you are at the selection level. The selection level is the point in a sequence after a RETURN statement and before a function is selected. If you wish to end your session and are at the function level, enter RETURN followed by SELECT END. The function level is the point in a sequence after a function is selected and before a RETURN statement is entered.

SELECT END should be followed by an end-of-input statement (/*) and an end-of-job statement (/&). If you enter an end-of-input or end-of-job statement at any point in the sequence, the job is canceled at that point. Any valid control statement sequences preceding the end-of-input or end-of-job statement are performed as specified.

CONTROL STATEMENT SEQUENCE EXAMPLES

The following are examples of batch execution sequences. Each example describes a possible sequence of functions and presents the control statements to perform those functions.

Each function and the statement that selects that function are labeled with the same letter so that you may make comparisons easily. The example in Figure 19-2 contains the following operations:

1. Select the Dump Management function and request the printing of the list of managed dumps.

2. On the selection level, specify SYSDUMP.F6.00000007 as the current dump.

3. Use the Dump Symptoms function to print the dump symptoms that are contained in the symptom record.

4. Use the Dump Viewing function to print selective areas of the dump. The assumed areas are written in the comments on each statement.

5. Use Dump Offload to offload SYSDUMP.F6.00000007 to the tape with VOLID T02512.

6. End your Info/Analysis session.

```
1.  SELECT DUMP MANAGEMENT
      PRINT DATA
      RETURN
2.  DUMP NAME SYSDUMP.F6.00000007
3.  SELECT DUMP SYMPTOMS
      PRINT DATA
      RETURN
4.  SELECT DUMP VIEWING
      PRINT 0 20880          * PRINT SUPERVISOR DATA
      PRINT C80000 END       * PRINT SVA
      PRINT 126000 FOR 1800  * PRINT JTB TABLE
      PRINT FORMAT           * PRINT ALL FORMATTED DATA
      RETURN
5.  SELECT DUMP OFFLOAD
      VOLID T02512
      RETURN
6.  SELECT END
```

Figure 19-2. Control Statement Sequence Example

The example in Figure 19-3 contains the following operations:

1. On the selection level, specify SYSDUMP.F4.00000006 as the current dump.

2. Use Dump Onload to load the current dump (file 3 on tape T300U1) into the a dump sublibrary so that you can work with it.

3. Use Dump Viewing to call routine IJBXDBUG to analyze the Stand-alone dump.. Results of the routine are printed.

4. On the selection level, specify SYSDUMP.F5.00000002 as the current dump.

5. Use Dump Offload to offload SYSDUMP.F5.00000002, specifying the output volume and choosing to bypass the write operation because a valid copy of the dump already exists on tape. (The information concerning this dump in the dump management file will be kept.)

6. On the selection level, specify SYSDUMP.F6.00000007 as the current dump.

7. Use Dump Onload to load the current dump (file 1 on tape T300U1) into a dump sublibrary, specifying LAST because it is the last dump to be onloaded from the tape.

8. End your Info/Analysis session.

```
1.  DUMP NAME SYSDUMP.F4.00000006
2.  SELECT DUMP ONLOAD
      VOLID T300U1
      FILE 003
      RETURN
3.  SELECT DUMP VIEWING
      CALL  IJBXDBUG
      RETURN
4.  DUMP NAME SYSDUMP.F5.00000002
5.  SELECT DUMP OFFLOAD
      VOLID T03417
      BYPASS
      RETURN
6.  DUMP NAME SYSDUMP.F6.00000007
7.  SELECT DUMP ONLOAD
      VOLID T300U1
      FILE 1 LAST
      RETURN
8.  SELECT END
```

Figure 19-3. Control Statement Sequence Example

CONTROL STATEMENT SUMMARY

This section contains a summary of the batch control statements for Info/Analysis. The statements are presented in alphabetical order. The "Valid Functions" column represents the functions during which the control statement may be entered as follows:

M	- Dump Management
S	- Dump Symptoms
V	- Dump Viewing
OF	- Dump Offload
ON	- Dump Onload
SEL	- Selection level
T	- Tutorial

Control Statement	Description	M	S	V	OF	ON	SEL
BYPASS	skip offload				X		
CALL {routine}	call analysis routine			X			
DELETE	delete dump	X					
DUMP NAME {dumpid}	specify dump	X					X
ERASE [YES] [NO]	delete/retain system copy of dump				X		
FILE {number} [LAST]	specify dump file					X	
HELP	print tutorial	X	X	X	X	X	X
PRINT {from-addr to-addr }	print dump data			X			
{from-addr END }				X			
{from-addr FOR length}				X			
{FORMAT }	print formatted dump			X			
{DATA }	print dump management file	X	X				
RETURN	end function	X	X	X	X	X	X
SELECT {DUMP MANAGEMENT} {DUMP SYMPTOMS } {DUMP VIEWING } {DUMP OFFLOAD } {DUMP ONLOAD } {END }	select function						X
UTILITY	initialize dump management file	X					
VOLID {volume[,logical unit nbr]}	specify input or output				X	X	

APPENDIX A. MESSAGE/SUBCOMPONENT CROSS-REFERENCE LIST

The table below shows the message prefixes related to the subcomponent identifiers and routine identifiers. See the VSE/Advanced Functions Messages and Codes for all further details, such as explanations and responses.

Prefix	Prefix	Subcomponent/Routine Name
0		Supervisor, IPL, and DOC
	C	Checkpoint
	D	DOC
	E	Emergency
	I/J	IPL
	P	PIOCS
	R	Restart
	S	EOJ
	T	MCAR/CCH
	V	EOJ
1		Job control
	A	ASSGN routine
	B	Buffer load program
	C	Job initiation and termination
	D/E	Library definition
	I	Attention routine
	L	Label error
	M/N	Cataloged procedure
	P	Attention routine
	Q/R	VSE/POWER
	S/T/U	Job control language
2		Linkage editor

Prefix	Prefix	Subcomponent/Routine Name
4		I/O and error recovery
	0	Retry
	1	Tape handling
	2-9	Disk handling:
	2	= indexed sequential file (and VSE/VSAM, if installed)
	3	= sequential input OPEN, including diskette
	4	= sequential output OPEN, including diskette
	5	= sequential CLOSE, including diskette
	6	= direct access input
	7	= direct access output
	8	= common OPEN/CLOSE routines
	9	= sequential disk work file
	A	VSE/VSAM
	B	BTAM-ES
	C	SDAID,
	D	PARSER routines
	E	Tape error statistics
	F	Hard copy file retrieval (stand-alone dump)
	G	DOSVSDMP
	M	MICR/OCR I/O
	P	Data check
	V	VTOC display/dump
5		ACF/VTAM
8		System utilities
	5	Copy and restore diskette
	6	Initialize tape
	B	Backup VSE, system
	C	Train cleaning
	F	Fast copy disk and VSE/Fast Copy Data Set
	M	Copy file and maintain object module
	R	Restore VSE, system and stand-alone
	V	VTOC display
	X	Common (for more than one utility program)
A		Assembler
E		VSE.OLTEP

Prefix	Prefix	Subcomponent/Routine Name
F		TOLTEP, CETAP
K		VSE/ICCF
L		Librarian
M		Maintain System History Program (MSHP)
P		3800 Printing subsystem support
	0	IEBIMAGE program
	1	SETDF routine
	2	Data management
	3	SETPRT routine
S		Stand-alone components
	1	Supervisor
	E	Format emulated extent
	R	Restore

APPENDIX B. SUPERVISOR CALL CODE (SVC) LIST

Code Dec	Code Hex	Issuing Macro	Generation Option	Function
0	00	EXCP	none	Execute channel program.
1	01	FETCH	none	Fetch a phase, except a transient phase.
2	02		none	Fetch a logical (B-) transient phase.
3	03		none	Quiesce I/O.
4	04	LOAD	none	Load any phase.
5	05	MVCOM	none	Modify the partition communication region.
		if issued by a phys. transient	none	Fetch another physical transient phase.
6	06	CANCEL	none	Cancel a problem program or a task.
7	07	WAIT	none	Wait for the posting of a CCB (or IORB) or a TECB.
8	08		none	Transfer control from a logical (B-) transient to a problem program.
9	09	LBRET	none	Return to a logical (B-) transient from the problem program after an SVC 8.
10	0A	SETIME	none	Set interval timer.
11	0B		none	Return from a logical (B-) transient.
12	0C		none	Reset switches in the partition communication region (COMREG).
13	0D		none	Set switches in the partition communication region (COMREG).
14	0E	EOJ	none	End a job and go to job control for end of job step processing.
15	0F	SYSIO	none	An SVC reserved for system tasks: headqueue an I/O request and execute the channel program.
16	10	STXIT PC	none	Establish linkage to user PC routine for program check interrupts.
17	11	EXIT PC	none	Return from the user PC routine.
18	12	STXIT IT	none	Establish linkage to user IT routine for interval timer

Code Dec	Code Hex	Issuing Macro	Generation Option	Function
				interrupt.
19	13	EXIT IT	none	Return from the user IT routine.
20	14	STXIT OC	none	Establish linkage to user OC routine for external or attention (operator command) interrupts.
21	15	EXIT OC	none	Return from the user OC routine.
22	16		none	Seize or release the system, enable or disable for external and I/O interrupts, set the key in a user PSW.
23	17		none	Store the load address of a phase at a defined user address.
24	18	SETIME	none	Establish linkage to user TECB, if any.
25	19		none	Issue an HIO for a telecommunication device or for any device if issued by OLTEP, with multiprogramming, dequeue an unstarted OLTEP I/O request to a shared device.
26	1A			Validate address limits.
27	1B		none	Issue an HIO for a telecommunication device without dequeueing the CCB.
28	1C	EXIT MR	MICR=type of SUPVR	Return from user MICR stacker select routine.
29	1D	WAITM	MICR=type of SUPVR if 370 mode; none if ECPS:VSE mode	Support usage of the WAITM macro.
33	21		none	Force task selection for a system task.
34	22	GETIME	none	Provide the time of the day and update the DATE field.
35	23		none	Hold a track for exclusive use by the requesting task.
36	24	FREE	none	Free a track held by the requesting task.
37	25	STXIT AB	none	Establish linkage to a user AB routine for abnormal termination of a task.
38	26	ATTACH	none	Initialize a subtask and establish its processing priority.
39	27	DETACH	none	Terminate a subtask, free tracks

Code Dec	Code Hex	Issuing Macro	Generation Option	Function
				that are held by the subtask.
40	28	POST	none	Indicate termination of an event and ready any waiting task.
41	29	DEQ	none	Indicate that a previously enqueued resource is available again.
42	2A	ENQ	none	Prevent two or more tasks from simultaneously manipulating a shared resource (e.g. data area)
44	2C		none	Support the creation of unit check records from outside the A or R-transient area.
45	2D	reserved		
46	2E		none	Allow OLTEP to run in supervisor state.
47	2F	WAITF	MICR=type of SUPVR	Support the multiple wait macro WAITF for MICR type I/O routines
48	30		DOC=type of FOPT	Fetch a CRT transient.
49	31		none	Allow VTAM to initiate the execution of a channel program.
50	32		none	Used by LIOCS when it encounters an error.
51	33		none	Make directory entry information for a phase available to the requesting task.
52	34	TTIMER	none	Return the remaining time interval or cancel a time interval.
53	35		none	Allow VTAM to schedule a user exit in an application program.
54	36		none	Release page frames to selection pool (applies only to /370 mode of operation).
55	37		none	Allow SDAID to acquire processor storage needed for program initialization (applies only to 370 mode of operation).
56	38	CPCLOSE	MODE=VM of SUPVR	Support the CPCLOSE command when VSE/Advanced Functions operates under VM/370 (applies only to 370 mode).
57	39	GETPRTY	none	Return partition priorities to the requesting task.
		SETPRTY		Change partition priorities as specified.
58	3A	INVPART	none	Acquire address space for program

Code Dec	Code Hex	Issuing Macro	Generation Option	Function
				execution.
59	3B	INVPAGE	none	Set one or more pages invalid.
60	3C	GETDADR	none	Return the virtual address of I/O area plus offset in the ERP and CRT routines.
61	3D	GETVIS	none	Request allocation of storage within the same partition or within the SVA.
62	3E	FREEVIS	none	Free storage requested through a GETVIS macro.
63	3F	USE	none	Use a system resource.
64	40	RELEASE	none	Release a system resource.
65	41	CDLOAD	none	Load a phase in the requesting partition GETVIS area unless that phase is already in the SVA.
66	42	RUNMODE	none	Return the system operating mode.
67	43	PFIX	none	Fix pages in processor storage.
68	44	PFREE	none	Free pages in processor storage.
69	45	REALAD	none	Return the real address corresponding to a given virtual ad - dress.
70	46	VIRTAD	none	Return the virtual address corresponding to a given real address.
71	47	SETPFA	none	Establish or terminate linkage to a user page-fault appendage routine.
72	48	GETCBUF	none	Get copy buffer for IDAL of tape ERP.
		FREECBUF		Free copy buffer for IDAL of tape ERP.
73	49	SETAPP	none	Allow linkage to channel-end appendage routines.
74	4A	PFIXREST	none	Fix page(s) in processor storage for restart.
		PFIXCHPT		Build parameter list for PFIXREST during checkpointing.
75	4B	SECTVAL	RPS=YES of FOPT	Calculate a sector value for a disk device with the RPS feature
76	4C		none	Initiate recording on the SYSREC file.
77	4D	TRANSCSW	none	Return the virtual address of an ERP CCW address copied from the pertinent CSW.

Code Dec	Code Hex	Issuing Macro	Generation Option	Function
78	4E	CHAP	none	Change the processing priority of the requesting subtask.
80	50	SETT	TTIME=part-id of FOPT	Set a task time interval.
81	51	TESTT	TTIME=part-id of FOPT	Return the remaining task time interval or cancel the time interval.
82	52		none	Set monitor call and/or branch for ICCF.
83	53		none	Allocate real or virtual address space.
84	54	SETLIMIT	none	Set partition sizes or PFIX limits or both.
85	55	RELPAG	none	Release the contents of one or more pages.
86	56	FCEPGOUT	none	Request a page-out operation for one or more pages.
87	57	PAGEIN	none	Request a page-in operation for one or more pages.
88	58	TPIN	none	Start TP balancing.
89	59	TPOUT	none	Stop TP balancing.
90	5A	PUTACCT	none	Provide interface with VSE/POWER for additional, user-provided account information.
91	5B		none	Provide interface with VSE/POWER for standard account information.
92	5C	XECBTAB	none	Define, delete, or check an entry in the cross-partition ECB table.
93	5D	XPOST	none	Set the traffic bit in a cross-partition ECB and ready any waiting tasks.
94	5E	XWAIT	none	Wait for a cross-partition ECB to be posted.
95	5F	EXIT AB	none	Return from a user abnormal termination routine.
96	60	EXIT TT	TTIME=part-id of FOPT	Return from a user task timer exit routine.
97	61	STXIT TT	TTIME=part-id of FOPT	Establish linkage to a user task timer exit routine for task time interval end.
98	62	EXTRACT	none	Extract system control information.
		MODCTB	none	Modify a PUB2 table entry.
		BOUNDARY	none	Retrieve partition boundary.
99	63	GETVCE	none	Return a specific volume charac-

Code Dec	Code Hex	Issuing Macro	Generation Option	Function
				teristics table entry.
100	64		MODE=E of SUPVR	Fix or free a page in the system GETVIS area.
101	65	MODVCE	none	Update the volume characteristics table.
102	66	GETJA	none	Update the fields in the requesting partition job accounting table.
103	67		none	Execute I/O operations for SYSFIL on an FBA device if supported.
104	68	EXTENT	none	Build, return, or delete DASD extent information.
105	69	SUBSID	none	Accept, return, delete subsystem identification information.
106	6A		none	Set the storage key for a specific area to the value in reg 0.
107	6B	GETFLD	none	Retrieve task-related information.
		MODFLD	none	Modify task-related information.
		TREADY	none	Post or cancel a task.
		TPOST	none	Deactivate current task or part.
108	6C	SECHECK	none	Check user authority for accessing the specified resource.
109	6D	PAGESTAT	none	Return the status of a page or a set of pages.
110	6E	LOCK UNLOCK	none	Protect a serially reusable resource against concurrent access by two or more tasks.
112	70	MSAT	none	Build, return, or delete stored assignment information.
113	71	XPCC	none	Execute cross-partition communication control function.
141	8D	VSIUCV	MODE=VM MODE=E	Provide subsystem support for VM/VCNA (VTAM Communication Network Application).

APPENDIX C. SYMPTOM RECORDS OVERVIEW

The symptom records contain a collection of failure-related symptoms in a standard format. At the time of problem detection, the failing component creates the symptom records. Subsequently, analysis routines may be run to collect additional symptoms and may add them to the symptom records. The ultimate goal of the symptom records is to reduce the amount of time necessary to analyze a dump.

SYMPTOM RECORDS STRUCTURE

The symptom records have six sections. Figure C-1 shows an overview of the symptom records contents and the information used in a dump.

```
SYMPTOM RECORDS:
                          Sections:
    1         2          3          4          5           6
 ______________________________________________________________________
| Environ | Pointer | Required | Optional | Optional | Logical Block |
|   ment  |         | Symptoms | Symptoms | Symptoms | Descriptors   |
|_________|_________|__________|__________|__________|_______________|
  |          |         |          |          |            |
  |     Pointer within |          |          |       Address Pointers
  |   the Symptom Record          |          |       to Dump Areas
  |                    |          |          |
  |     _______________|__________|_______   |
  |    |               |          |       |  |
  └──> |  ENVIRONMENT: |          |       |  |
       |      CPU MODEL |..........4331   |  |
       |      CPU SERIAL|..........022001 |  |
       |      ......   |          |       |  |
       |               V          |       |  |
       |  REQUIRED SYMPTOMS:      |       |  |
       |                          |       |  |
       |      AB/S0900            |       |  |
       |      REGS/0E014          |       |  |
       |      .......       ┌─────┘       |  |
       |                    v             |  |
       |  OPTIONAL SYMPTOMS (SDB):        |  |
       |                                  |  |
       |  OPTIONAL SYMPTOMS (NON-SDB): <──────┘
       |                                  |
       |      MODE=PAGING                 |
       |__________________________________|

        Symptom Part of a Dump
```

Figure C-1. Symptom Records

The sections of the symptom records are:

1. The environment section, which describes the operating system at the time the problem is detected.

2. The pointer section, which describes the offsets of each of the other sections from the beginning of the record and the length of each section. This data is used by the analysis tool and is not accessible to you.

3. The required symptoms section, which contains symptoms considered essential to your dump analysis tasks. Required symptoms are in the structured data base (SDB) format.

4. The optional symptoms (SDB) section, which contains additional symptoms in the SDB format.

5. The optional symptoms (non-SDB) section, which contains symptoms that do not conform to the SDB format but provide failure-related information.

6. The control block section, which contains descriptions and locations of control blocks that are necessary for problem analysis. This section also contains text and hexadecimal entries that may be related to the problem and descriptions of control block chains and arrays.

The contents of Sections 1, 3, 4, and 5 is displayed when you select Dump Symptoms. This information can help you determine the nature of the problem and where it occurred.

The contents of Section 6 is used by Info/Analysis to display dump information when you select Dump Viewing. The control block data presented via the Section 6 entries can help you determine why the problem occurred. See the discussion of Section 6 below.

Note: The structured data base format is used to standardize problem data so that searches for duplicate problems in the data base of existing problems used by customer engineering are accurate.

SYMPTOM RECORD CREATION

The symptom records are built when:

- A system component detects an error that may or may not result in a dump. The dumping component builds the symptom records, completing the required symptoms section and as much optional data as possible. The dumping component then calls a system dumping routine that fills out the environment section, merges it with the rest of the symptom records, and possibly takes a dump.

- A stand-alone dump is taken. The environment section is completed by the dump program.

ADDING DATA TO A SYMPTOM RECORD

The capability of Info/Analysis to access the symptom records can be used by analysis routines to retrieve and examine any part of the symptom records. The routines may then use the symptom records update service of Info/Analysis to add data to the symptom records. Analysis routines may not delete or modify the original information in the symptom records; only additions are possible.

SECTION 6

Section 6 acts as a table of contents for dump data. You can use it to locate certain control blocks without having to manually follow pointers through the dump. The component which originates the dump is responsible for providing information about the control blocks that are pertinent to the error.

The section provides for example:

- Names and locations of dumped control block storage.
- Descriptions of conditions present at the time of the dump.
- Hexadecimal data that may not be contained in the dumped storage (such as registers).
- Algorithms for control block relationships.
- The names of control block fields.

The location of Section 6 entries is in separate records of the dump that are classified as symptom record extensions.

At dump time, the failing component may designate the control blocks that are suspected of being in error or are necessary for problem determination. Ideally, the dump includes only the storage in use when the error occurred. Host system storage that provides pointers to the component address space or partition may not be needed. This practice reduces the volume of dump output.

The failing component may include descriptions of the related control blocks in Section 6. Keep in mind that the data for the control blocks is within the main body of the dump. The information in Section 6 describes the addressing method, the content, the format, and the chaining structure of these related dump areas.

Each of these descriptive entries in Section 6 is called a locating block descriptor (LBD). LBDs come in a variety of forms to describe the structure and relationships of control blocks. Each LBD consists of a header segment and, optionally, a variable segment. The header identifies the data being described by the LBD by providing a name, its length, and usually, its location in storage. The optional portion may be:

- One or more extensions
- A formatting descriptor
- A linkage descriptor

By including a variety of LBDs, Section 6 becomes a table of contents for dump data. Through the Dump Viewing function of Info/Analysis, you can use Section 6 to analyze the dump.

Locators

If a header portion of an LBD provides a control block address, a specific instance of a control block has been identified. An LBD may still be a locator if the address is not provided. This would be the case for a linkage descriptor, for example.

A locator may be simple or complex. A simple locator names a control block and defines its length, address qualifications, and other pertinent information. A simple locator is appropriate if there is one occurrence of a particular control block in a dump or in a linkage. A complex locator consists of a header portion resembling a simple locator and one or more extensions that:

- Provide a way to find all occurrences of the control block that is defined in the header.
- Associate additional data with the control block defined in the header.

There are five kinds of locator extensions:

- Chain extension - describes a string of all occurrences of one type of control block in a dump. If you select a linkage descriptor while viewing the analysis summary main display, and the locator for a particular type of control block has a chain extension, Info/Analysis uses the locator and its chain extension to display the chain of control blocks of that type.
- Array Extension - describes contiguous occurrences of one type of control block in a dump. If you select a linkage descriptor while viewing the analysis summary main display, and the locator for a particular type of control block has an array extension, Info/Analysis uses the locator and its array extension to display the array of control blocks of that type.

- Text Extension - contains character data added to the dump by the dumping component. The text is associated with the name defined in the locator's header. You may view text for which the header merely provides a name by selecting this text entry from the analysis summary main display. Alternatively, the header may describe a control block with which the text is associated. If you select a locator while viewing the analysis summary LOCATORS display, and the locator by that name has a text extension, the lines of text are displayed along with the control block data.

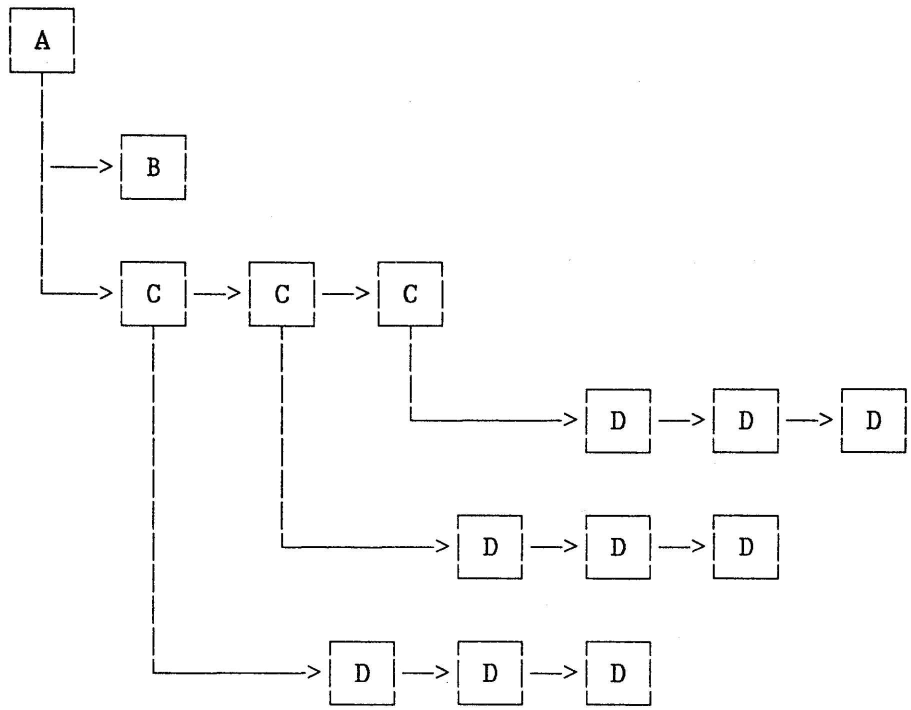
- Hexadecimal Extension - contains hexadecimal information such as register contents or storage data areas that may pertain to the software problem that caused the dump. This data is collected and associated with the name defined in the header portion of the locator. You may view hexadecimal data for which the header merely provides a name by selecting the hexadecimal entry from the analysis summary main display.

- Keyfield Extension - contains the location of a particular field (such as a completion code) that is significant or pertinent to the associated control block identified in the header portion of the locator. This field may be in (or apply to) the control block. According to the status of this field, you can decide whether or not to continue to examine the control block. The key-field is included in an analysis summary linkage or LOCATORS display.

Linkage Descriptors

A linkage descriptor is a special type of extension to a header that defines the relationships for a set of control blocks. That is, the control block named in the linkage is used to locate one or more control blocks that are logically related.

Linkage descriptors are used in conjunction with locators. Together, they enable Info/Analysis to display an ordered list of control blocks. For example, a component might create a linkage descriptor that defines control block A pointing to B and a chain of Cs, and each C pointing to a chain of Ds. When you select the linkage entry for that component while viewing the analysis summary main display of Dump Viewing, the displayed data depicts the following situation:



Info/Analysis shows the linkages by indenting control block names. To create the display, Info/Analysis uses both locators with their extensions, if any, and a linkage descriptor.

Formatting Descriptors

A formatting descriptor is a special type of extension to a header that defines a set of simple formatting instructions for a control block. The header portion of the formatting descriptor cannot be used to locate a control block in the dump.

A formatting descriptor for a control block maps out some or all of the fields of the control block named in the associated locator, their offsets from the beginning of the control block, and their lengths. When you select a control block for display while in the formatted mode of dump display, Info/Analysis displays the contents of each field, one or more per line. Depending on the options you have set, the field labels and offsets may appear with the data. To create such a display, Info/Analysis uses both a locator and a formatting descriptor.

APPENDIX D. OTHER SERVICE AIDS

This appendix describes various commands and facilities to process information relevant for problem analysis. The following information is contained in this chapter:

ACTION: Print Linkage Editor Map

ACTION MAP

To obtain a linkage editor map for a program, specify, for the program's linkage editor run, the linkage editor control statement ACTION MAP.

Figure D-1 on page D-3 is a sample output from this routine.

Linkage Editor Map Warning Messages

The following messages may be included in the map output, except when NOMAP was specified in the ACTION statement for the linkage editor run.

ROOT STRUCTURE OVERLAID BY SUCCEEDING PHASE: When this message appears, "OVEROOT" is printed to the left of the phase name that overlays the root phase.

POSSIBLE INVALID ENTRY POINT DUPLICATION IN INPUT: An entry label appeared at least twice in the input. At the second (or later) appearance it was not possible to validate it as being a true duplication. The most common reason for this message is submodular structure with (source) entry labels defined before the CSECT in which the entry point appears.

INVALID TRANSFER LABEL ON END OR ENTRY STATEMENT IGNORED: An overriding transfer label in the entry statement was not defined within the first phase, or a transfer label was not defined in an end statement in its own module.

CONTROL SECTIONS OF ZERO LENGTH IN INPUT: The COBOL, FORTRAN, RPG, AND PL/1 (D) compilers do not supply all of the information required by the linkage editor in the ESD records. Specifically, the control section length is provided in the end record. If a control section defined in the ESD information has a length of zero, it normally indicates that the length is to appear in the end record. It is possible to generate zero-length control sections through Assembler. Such a condition produces this message. This is not an invalid condition if it is not the last control section that is of zero length. If the last control section is of zero length, the length is implied to be in the end record and, if not present, causes an error condition.

UNRESOLVED EXTERNAL REFERENCES: These labels indicate external references that cannot be matched with a corresponding entry point. ESD items from unused control sections may also cause this message.

```
// JOB LNKPLNK1
// LIBDEF PHASE,CATALOG=IJSYSRS.SYSLIB
// LIBDEF OBJ,SEARCH=(IJSYSRS.SYSLIB,SERVLIB.S1$XE8)
// OPTION CATAL
 ACTION MAP
 INCLUDE INLPLNK1
// EXEC LNKEDT,PARM='MSHP'


JOB  LINKSL2   07/25/84   5666-301   LINKAGE EDITOR DIAGNOSTIC OF INPUT
-
ACTION TAKEN  MAP    <------------------------------------(A)
INCLUDE INLPLNK1     <------------------------------------(B)
      ** MODULE INLPLNK1 84-03-20 15.54 84-04-10       INCLUDED FROM SERVLIB .S1$XE8   VOLID=SYSRES
*******************************************************************
PHASE LIBR,*
*******************************************************************
INCLUDE INLPMAIN     <------------------------------------(B)
      ** MODULE INLPMAIN 84-03-20 15.54 84-04-10 12.10 INCLUDED FROM SERVLIB .S1$XE8   VOLID=SYSRES
INCLUDE INLPAREA
                             ┌----(C)         ┌----(D)                    ┌----(E)
                             V                V                           V
      ** MODULE INLPAREA 84-03-20 15.55 84-04-11 12.15 INCLUDED FROM SERVLIB .S1$XE8   VOLID=SYSRES

                    --------------------
                     -- continuation --
                    --------------------
INCLUDE INLPRDTP
      ** MODULE INLPRDTP 84-03-20 15.55                INCLUDED FROM SERVLIB .S1$XE8   VOLID=SYSRES
*******************************************************************
PHASE LIBRROLD,LIBRRES1,NOAUTO    <------------------(F)
*******************************************************************
INCLUDE INLPROLD
      ** MODULE INLPROLD 84-03-20 15.55 84-04-15 14.10 INCLUDED FROM SERVLIB .S1$XE8   VOLID=SYSRES
INCLUDE INLPDBLO
      ** MODULE INLPAREA 84-03-20 15.55 84-03-21 10.15 INCLUDED FROM SERVLIB .S1$XE8   VOLID=SYSRES
INCLUDE INLPRTI
      ** MODULE INLPRTI  84-03-20 15.55 84-04-11 16.15 INCLUDED FROM SERVLIB .S1$XE8   VOLID=SYSRES
INCLUDE IJJTCTL
      ** MODULE IJJTCTL  84-02-14 20.55 84-04-15 15.10 INCLUDED FROM IJSYSRS .SYSLIB   VOLID=SYSRES
ENTRY
```

Figure D-1 (Part 1 of 3). Sample: Linkage Editor Output (ACTION MAP)

(A) Option MAP was specified in the ACTION statement for the linkage editor run.
(B) Listing of control statements as submitted to linkage editor.
(C) Date and time the module has been cataloged the first time.
(D) Date and time of last update.
(E) Sublibrary from where the module is included.
(F) Phase statement from the included module. This statement defines the phase name, the load address (for example * to indicate relocatable) and for example, whether the phase has to be SVA eligible or the AUTOLINK feature has to be deactivated. The named phase is combined by the subsequent included modules.

Figure D-1 (Part 2 of 3). Sample: Linkage Editor Output (ACTION MAP)

```
        (A)     (B)        (C)        (D)       (E)      (F)   (G)      (H)    (I)          (J)

07/25/84 PHASE  XFR-AD  LOCORE  HICORE  CSECT/     LOADED RELOC. PARTIT. PHASE  TAKEN
                                        ENTRY      AT     FACTOR OFFSET  OFFSET FROM

         LIBR   060078  060078  06EBC8                                                   RELOCATABLE
                                         INLPMAIN 060078 060078 000000  000000 INLPMAIN
                                        +INLPEND  060096
                                        +INLPCACK 06009C
                                        +INLPCANC 0600A2
                                         INLPAREA 060FC0 060FC0 000F48  000F48 INLPAREA

                         --------------------
                          -- continuation --
                         --------------------

                                        +INLPRTIC 0750F8
                                        *INLPRTIM 0750FE
                                         IJJTCTL  075938 075938 0158C0  004DB0 IJJTCTL
PHASE(S) CATALOGED INTO SUBLIBRARY  IJSYSRS.SYSLIB   VOLID= SYSRES

1S55I LAST RETURN CODE WAS 0000
EOJ LNKPLNK1  MAX.RETURN CODE=0000
```

(A) Name of each phase.
(B) Address, where the phase is transferred to.
(C) Lowest and highest virtual storage location of the phase.
(D) Labels of all CSECT's which establish the phase in ascending order.
+ indicates an entry label, and * indicates that the CSECT is not referenced.
(E) CSECTs load address.
(F) Difference between the start of virtual storage and the CSECT start address
(If the CSECT was assembled to 0).

The length of a COMMON area (generated by a COM statement in a compiler source module) if such an area is present is also given in this column.
(G) Offset from the partition begin plus save area length to the CSECTs start location.
(H) Offset from the phase begin to the CSECTs start location.
(I) Name of the module from which the CSECT is taken.
(J) Indicates whether a phase is relocatable, self-relocatable, not relocatable, or SVA eligible (in this example, the linkage editor produces relocatable phases).

Figure D-1 (Part 3 of 3). Sample: Linkage Editor Output (ACTION MAP)

Device Support Facilities: Disk Surface Analysis

Device Support Facilities (DSF)

VSE includes a programming tool to aid in determining whether to start data recovery procedures (for disk surface problems) or to call IBM for assistance (for disk drive problems). Refer to the Device Support Facilities manual for details.

DITTO: Dump a Disk, Diskette, or Tape

DITTO

VSE/DITTO (VSE/Data Interfile Transfer, Testing and Operations), a program product, is a useful tool for the recovery of data that may have become inaccessible by VSE programs.

You can use VSE/DITTO to print, on a line printer, data stored on disk, diskette, or tape. The program dumps the data either in character-only format or in the character and vertical hexadecimal format.

For more detailed information about VSE/DITTO, refer to the VSE/DITTO, General Information, manual.

DSPLY/ALTER: Display or Alter Storage

You can display or alter 16 bytes of the real or the virtual address ranges of virtual storage at the operator console. You can do this either by the **DSPLY** or **ALTER** operator commands or by using the hardware storage display feature as described under "Hardware Aids via the Operator Console" on page E-6.

To request a **display** of storage, enter the command:

```
DSPLY [x,] address
```

To **alter**, enter:

```
ALTER [x,] address
```

followed by the new data in hexadecimal form (up to 16 characters).

For **x** specify the virtual address space to which the address belongs (R, 1, 2, or 3).

For **address**, specify a six-digit hexadecimal address (for an address of less than six significant digits, leading zeros are required). If you enter an odd number of characters for the alter data, the last character is ignored.

Figure D-2 shows an example of using the ALTER and DSPLY commands.

DSPLY 1,000300	display command
90F21028 18215811 00185801 00141200 1I40I READY	system displays
ALTER 1,000300	alter command
ENTER DATA	system responds
FFF2	new data

Figure D-2. Sample: DSPLY and ALTER Command

The example above shows the commands to display the contents of address X'000300' in space 1 and to alter two bytes beginning with the same address to X'FFF2'.

For a detailed description of the ALTER or DSPLY command see VSE/Advanced Functions System Control Statements.

Restriction:

If the specified address is within an invalid address space, the system issues a message on SYSLOG. If the 16 bytes cross from a valid to an invalid address space, or beyond the highest available address, only those bytes within the valid area are displayed or altered, and a message is issued.

Invalid addresses are:

- Locations beyond the end of virtual storage.
- Unused (not allocated) partition GETVIS space.

when operating in 370 mode:

- A location in the page pool
- A location in the system GETVIS area.
- A location within an unallocated area of the virtual address space.
- A location in the partition's virtual address area when a program in that partition is being executed in real mode, or vice versa.

LIBLIST: Display Library Chains

```
(//) LIBLIST  [libtype|*](,*|partition-id)
                          (,SYSxxx)
```

Library search chains established with the job control LIBDEF statement can be displayed either on the system console or on SYSLST with the LIBLIST job control statement.

```
libtype = Corresponds to the type operand of the
          LIBDEF statement
*       = specifies that library definitions of all
          LIBDEF statements (except DUMP) are to be
          displayed.

partition-id = BG or Fn: partition whose
               library chains are to be listed.
*            = The library chains of the partition
               processing the statement are to be
               listed (default).

SYSxxx = SYSLST or SYSLOG device for output.
         Default = SYSLOG if entered from SYSLOG,
                 = SYSLST if entered from SYSRDR.
```

Figure D-3 on page D-10 is an example of a listing of BG partition's active library search chains resulting from a LIBLIST command.

```
// LIBLIST *,BG

TYPE: PHASE

    BG-TEMP ** NO LIBRARY INFORMATION AVAILABLE

    BG-PERM LIBNAME SUBLIB      STATUS  -PARTITIONS-
    SEARCH  PRVLIB  TCLIB               01 34
            IJSYSRS SYSLIB              0
    CATALOG PRVLIB1 TCLIB               0 23

TYPE: OBJ

    BG-TEMP ** NO LIBRARY INFORMATION AVAILABLE

    BG-PERM LIBNAME SUBLIB      STATUS  -PARTITIONS-
    SEARCH  PRVLIB  TCLIB               01 34
            IJSYSRS SYSLIB              0
            PRVLIBS SLIB2       SEC SHR 0   4

TYPE: SOURCE

    BG-TEMP ** NO LIBRARY INFORMATION AVAILABLE

    BG-PERM LIBNAME SUBLIB      STATUS  -PARTITIONS-
    SEARCH  PRVLIB1 TCLIB               01 34
            IJSYSRS SYSLIB              0
            SERVLIB S1$XE8              0

TYPE: PROC

    BG-TEMP ** NO LIBRARY INFORMATION AVAILABLE

    BG-PERM LIBNAME SUBLIB      STATUS  -PARTITIONS-
    SEARCH  PRVLIB1 TCLIB               01 34
            IJSYSRS SYSLIB              0
```

No temporary search chain is defined in the above example.

The device on which PRVLIBS SLIB2 resides is shared by two or more CPUs - indicated by SHR in the STATUS column. SLIB2 is also a secured (SEC) library.

Figure D-3. Example: Library Chain Listing

LIST: Print Language Translator Source Code

```
// OPTION LIST
```

Normally, a language-translator source listing is requested for all language-translator runs by specifying option **LIST** in the STDOPT command as part of the IPL procedure. Should NOLIST have been specified in that command, you can override it by including // OPTION LIST in the job control statements for your language-translator run.

LISTIO: List I/O Device Assignments

```
LISTIO
```

You can list I/O device assignments before execution of the program begins by inserting the job control **LISTIO** statement or command in the program. The list will appear on SYSLOG if you insert the LISTIO command without //, or on SYSLST with //. The listing represents the device assignment status at the time the statement or command is being executed and not when an error occurred during a previous run.

For more information about the type of output obtained with the LISTIO options, refer to VSE/Advanced Functions System Control Statements.

LISTLOG: Display Console Communication

```
// EXEC LISTLOG
```

You can use the LISTLOG utility program to request a listing of all information collected for a specific job in the hard copy file. The program writes this list to the device assigned to SYSLST.

You invoke the program by inserting the statement // EXEC LISTLOG immediately following the /& statement for the job. VSE/Advanced Functions invokes the program automatically whenever a job is cancelled.

The printout provided by the LISTLOG utility program lists:

- Job control statements submitted for the job,
- All messages displayed on the console for this job,
- Any attention routine messages and commands that occurred while the job was being executed,
- Operator responses.

LOG: Print Job Control Statements

```
// STDOPT LOG=YES
// OPTION LOG
```

The system lists the job control statements of any job on SYSLST if the following command is not defined:

```
// STDOPT LOG=NO
```

If LOG=NO was specified, the system lists the job control statements for a particular job on SYSLST if you include the following command in the set of job control statements for that job.

```
// OPTION LOG
```

Later, after the job has been executed successfully several times, listing the job control statements for that job on SYSLST may be suppressed by including

```
// OPTION NOLOG
```

in the set of job control statements for that job if LOG=NO was not specified in the STDOPT command, or else by removing the // OPTION LOG statement, whichever applies.

LSERV: Display Label Information Area

LSERV, a system utility program, produces a printout of the system's label information area. You invoke this program by entering:

```
// EXEC  LSERV
```

via the console or by submitting control statements as follows.

```
// JOB   anyname
// EXEC  LSERV
/*
/&
```

A sample, partial output of an LSERV run for the above control statements, is shown in the example in Figure D-4 on page D-14. The output indicates the files whose labels have been stored in the label information areas for the various partitions, whether these labels have been stored permanently or temporarily, and (for the labels on a CKD volume) the label information area track.

The output shows the relationship between job control DLBL and EXTENT statements (for further information see the VSE/Advanced Functions System Management Guide).

For VSAM files only: there is an additional label information record following the VSAM label record if, in the DLBL statement, at least one of the operands DISP, RECORDS, and RECSIZE is specified.

> **Note:** A warning message is issued on SYSLST if you request LSERV while another partition is updating the label area or if the label area is being updated when the system terminates with an error (such as a hard wait or disabled loop).
>
> The free usage subarea, which is internally used by ICCF, is not formatted by the LSERV program.

```
                               LABEL INFORMATION DISPLAY   PAGE nnn
EXAMPLE
    FILE IDENTIFIER                         EXAMPLE
    FILE SERIAL NUMBER                      OMITTED
    VOLUME SEQUENCE NUMBER                  01
    CREATION DATE                           OMITTED
    RETENTION PERIOD (DAYS)                 0999
    FILE TYPE                               SEQUENTIAL

  EXTENT INFORMATION
    EXTENT SEQUENCE NUMBER                  000
    EXTENT TYPE                             1 (PRIME DATA)
    RELATIVE START ADDRESS                  002
    NUMBER OF TRACKS/BLOCKS                 045107
    SYMBOLIC UNIT                           SYSRES  LOGICAL UNIT FORMAT
                                            TYP#00,NUM#06
    VOLUME SERIAL NUMBER                    OMITTED

  ADDITIONAL INFORMATION
    DISPOSITION                             (OLD,KEEP)
    RECORDS                                 (0000000500,0000000100)
    RECORD SIZE                             0000000080

SALARY
    FILE IDENTIFIER                         SALARY.1983.FILE
    FILE SERIAL NUMBER                      DASD02
    VOLUME SEQUENCE NUMBER                  01
    CREATION DATE                           OMITTED
    EXPIRATION DATE                         83/365
    FILE TYPE                               SEQUENTIAL

  EXTENT INFORMATION
    EXTENT SEQUENCE NUMBER                  000
    EXTENT TYPE                             1 (PRIME DATA)
    RELATIVE START ADDRESS IN TRACKS/BLOCKS 010000
    NUMBER OF TRACKS/BLOCKS                 001000
    SYMBOLIC UNIT                           SYS019  LOGICAL UNIT FORMAT
                                            TYP#01,NUM#13
    VOLUME SERIAL NUMBER                    DASD02
```

Figure D-4. Sample: LSERV Output

LVTOC: Display Volume Table of Contents

```
// ASSGN SYS004,cuu
// ASSGN SYS005,cuu
// EXEC LVTOC
```

A volume table of contents (VTOC) is an index of all files, and the remaining space, on a DASD volume.

A VTOC display can be requested by executing the LVTOC program with SYS004 assigned to the applicable disk drive and SYS005 to a printer.

LVTOC lists the file labels contained in a VTOC in alphabetic sequence by file name. It also provides a listing of free space on the volume, with the start and end addresses and sizes of the unused space. The control statements needed to invoke that program may be submitted via SYSRDR or via the console as shown in Figure D-5 on page D-16.

A display of a VTOC can be requested also in response to messages as indicated in VSE/Advanced Functions Messages and Codes. Such a response is CANCELV or DSPLYV. Use CANCELV if you intend to cancel the job, or DSPLYV if the condition allows program execution to be continued after the VTOC display.

The LVTOC program is also described in VSE/Advanced Functions System Utilities.

```
Submission via SYSRDR              | Submission via the console
-----------------------------------------------------------------
// JOB    anyname                  | 1.  Press the Request key
// ASSGN  SYS004,cuu      (A)      | 2.  Enter:
// ASSGN  SYS005,cuu      (B)      |     PAUSE p.-id,EOJ          (C)
// EXEC   LVTOC                    | 3.  Wait for end-of-job in the
/&                                 |     specified partition.
                                   | 4.  Enter:
                                   |     // ASSGN  SYS004,cuu     (A)
                                   |     // ASSGN  SYS005,cuu     (B)
                                   |     // EXEC   LVTOC
```

(A) The unit address of the disk for which the VTOC is desired.

(B) The unit address of the device on which the VTOC is to be listed (normally a line printer).

(C) The identifier of the partition (F1 ...) in which LVTOC is to run.

Figure D-5. Control Statements to Invoke LVTOC

MAP: Map Storage/Partition Allocations

MAP

With the **MAP** command, the system lists the allocations of storage.

The format of the list of allocations is slightly different for 370 and ECPS:VSE mode, as shown in Figure D-6 on page D-18, but the information provided as MAP command output is self-explanatory to a great extent.

MAP command output in 370 mode of operation

SPACE	AREA	PRTY	V-SIZE	GETVIS	V-ADDR	R-SIZE	R-ADDR	NAME
S	SUP		320K		0	210K	0	$$A$SUPB
1	BG V	12	208K	48K	50000	0K		BATCH1
1	F3 V	6	2048K	1024K	90000	0K		CICS1
1	F5 V	7	3584K	1536K	390000	0K		CICS2
1	F7 R	3	80K	48K	890000	224K	76000	MICR
1	UNUSED		2688K		9B0000			
2	F8 V	8	392K	120K	50000	0K		BATCH2
2	F6 V	5	2048K	2048K	D0000	0K		SQL
2	UNUSED		6056K		4D0000			
S	F4 V	1	512K	256K	B40000	52K	69000	POWER
S	F2 V	4	1024K	512K	C00000	100K	50000	VTAM
S	F1 V	2	386K	128K	D80000	0K		OCCF
S	SVA		1280K	768K	E00000	70K		
	AVAIL		2112K			3440K		
	TOTAL		20480K			4096K		

MAP command output in ECPS:VSE and VM mode of operation:

AREA	PRTY	V-SIZE	GETVIS	V-ADDR	R-SIZE	NAME
SUP		224K		0	180K	$$A$SUPD
BG V	12	208K	48K	38000	0K	BATCH
F3 V	6	2048K	1024K	78000	0K	CICS1
F5 V	7	3584K	1536K	378000	0K	CICS2
F7 R	3	176K	48K	878000	224K	MICR
F4 V	1	512K	256K	8B0000	52K	POWER
F2 V	4	1024K	512K	970000	100K	VTAM
F1 V	2	384K	128K	AF0000	0K	OCCF
AVAIL		2624K		B70000	3470K	
SVA		1280K	768K	E00000	70K	
TOTAL		16384K			4096K	

Figure D-6. Sample: MAP Command Output

MSHP: Altering the Code of a Phase

MSHP

It may be desirable to alter (patch) the code of a phase stored in a library.

The CORRECT function of the MSHP program is available for this purpose.

For more information on the MSHP program, refer to the VSE/Advanced Functions Maintain System History Program Reference manual.

STOP/PAUSE: Suspend Program Execution

STOP/PAUSE

Suspending program execution between job steps can be of much help during hands-on diagnosis.

You suspend program execution with the **STOP** command either via the console or, if you use a card reader, via SYSRDR. Another possibility is to submit the job control **PAUSE** statement or command. Both methods result in program execution to be suspended when job control executes the statement or command. The PAUSE command is used to interrupt the jobstream. The operator may enter additional job control statements via SYSLOG at this time. The STOP command removes the partition from the system's task selection mechanism and no read is issued to the SYSLOG or SYSRDR device for that partition.

To resume program execution after a STOP command, issue an attention routine **START** command for the partition. To resume program execution after a PAUSE statement or command, simply press END/ENTER.

APPENDIX E. HARDWARE SERVICE AIDS

This appendix contains descriptions of the following commands or functions:

CONTROLLING THE RECOVERY MANAGEMENT FACILITY

The recording activity of RMS can be controlled via operator commands MODE and ROD, as follows:

- Use the **MODE** command to:
    - Set the recording mode for unlabeled or nonstandard labeled tapes to either individual or to combined (IR, CR options).
    - Set the recording mode for a particular device to intensive recording, diagnostic recording, or no recording (CE.cuu I, D, N).
    - Set the recording mode for the processor to quiet or to recording and specify for which type of processor errors -- hardware retries (HIR) or error correction code (ECC) -- the setting of recording mode is to apply. (NOTE: Does not apply to a 4300).
    - Set the error count threshold value to a value that is different from the IBM defined default value (T, E). (NOTE: Does not apply to a 4300).
    - Place the Model 145 or 148 control storage ECC recording activity in 'threshold mode' (TM).
    - Print a status report on SYSLOG.

- Use the **ROD** command to:
  - Add error statistics to the system recorder file.
  - Have RMS write MDR records into the SYSREC file for those devices that are equipped with an internal error log.
  - Have RMS build an end-of-day (EOD) record and write this record on SYSREC.
  - Write the Hard Copy buffer into the Hard Copy file.

The MODE and ROD commands are discussed in more detail in the following sections.

The MODE Command

Note: The length of a MODE command must not exceed 30 characters.

```
MODE {IR|CR}
```

Use this format to set the recording mode for unlabeled tapes and for tapes with nonstandard labels. Specify

IR (Individual Recording) to have RMS accumulate tape errors (in PUB2) individually per volume.

CR (Combined Recording) to have RMS accumulate all error statistics (in PUB2) from unlabeled and nonstandard labeled tape volumes on a tape drive and make one "combined recording" of those statistics when a tape volume with standard labels is opened on that drive.

```
MODE CE,cuu{,I[,xx,y]|,D[,xx,y]|,N}
```

Use this format when the error recording mode for a specific device is to be set or reset. The format of the command without any of the optional specifications resets the recording mode for the specified device to normal. Specify

CE,cuu,I[,xx,y] if you want RMS to do intensive recording.

When this format is used, RMS continues normal recording for the device specified by cuu, the channel and unit address. In addition, however, RMS records the next seven errors indicated by bit y (any number from 0 to 7) in byte xx (any from 0 to 31) of the sense data that is to be checked, or it records the next seven errors of any type if xx,y is omitted.

CE,cuu,D[,xx,y] if you want RMS to do diagnostic recording.

If this type of recording is requested for a device, RMS performs the same recording activity as for an intensive recording request plus a recording of the number of I/O retries required until successful execution of an originally failing I/O operation.

CE,cuu,N if, for any reason, you do not want RMS to do any error recording for a device.

Once you have turned off error recording by RMS for a device, this error recording cannot be turned on again, except through system start-up.

```
MODE {R|HIR,Q}
```

You can use this format to set RMS either in recording mode (R) or in quiet mode (HIR,Q).

```
MODE {HIR|ECC}[,M|,C]{,R|,Q|,TH}[,E=number]
                              [,T=number]
```

You can use this format when it is necessary to exercise close control over the recording activity of RMS. Specify

HIR{,R|,Q} (Hardware Instruction Retry) to set RMS in recording mode (R) for HIR type soft machine check interrupts or to set RMS in quiet (non-recording) mode (Q) for both HIR and ECC type soft machine check interrupts.

ECC{,R|,Q} to set RMS in recording mode (R) or quiet mode (Q) for ECC type soft machine check interrupts.

{HIR|ECC},R[,E=number][,T=number] to set an EFL (error frequency limit) for the recording of both HIR and ECC type soft machine checks (HIR,R does this for you) or only for ECC type soft machine checks. In the command, E=number specifies the number of errors and T=number specifies the number of hours within which the specified number of errors may occur before RMS switches to quiet mode of operation. For number, you can specify any decimal value from 8 to 9999.

The optional operands M and C, and also the option TH, apply only to Models 145 and 148 and are available for controlling the recording activity for ECC type soft machine checks. Specify

M (main) to apply the requested setting of controls to user-available processor storage.

C (control) to apply the requested setting to the central processor's control storage.

TH (threshold) to indicate "threshold mode," which causes RMS to enter quiet mode for control storage ECC type soft machine checks on next occurrence of this type of a check. TH is valid only if you specify also ECC,C.

```
MODE STATUS
```

Use this format of the command to request a display of the setting of the RMS recording controls.

The ROD Command

```
ROD
```

The ROD command has no operands. Issuing the ROD command causes the hard copy buffer written to the hard copy file and RMS to record, on SYSREC:

- Error statistics that were compiled for I/O devices (except telecommunication devices).
- An end-of-day record if RMS received an appropriate response to a prompting message via the console.

Retrieval and Analysis of RMS Information

The EREP Program

For the retrieval of information recorded by RMS on SYSREC, use the DOS/VSE EREP program. How to use this program is described in a separate publication, OS/VS, DOS/VSE, VM/370 Environmental Recording Editing and Printing (EREP) Program.

For a number of VSE messages, the recommended response in VSE/Advanced Functions Messages and Codes includes instructions to run EREP.

The SEREP Program

SEREP, a Stand-alone limited function version of EREP, can be used for the retrieval of information contained in the /370 central processor's logout area. The program edits the retrieved information and writes it to a line printer. A SEREP run should be executed only if:

- This is requested by IBM service personnel.
- This is indicated by the contents of bytes 1 and 2 of a hard-wait code in bytes 0 through 3 of processor storage.
- You are instructed to do so by a VSE message.

The procedure for the execution of a SEREP run is processor-model dependent, and is described in VSE/Advanced Functions Operations.

HARDWARE AIDS VIA THE OPERATOR CONSOLE

Current IBM processors provide a variety of hardware aids for hands-on diagnosis. The procedures for using these aids are, for the most part, processor-model dependent, and are described in detail in the operating procedures manuals for these processors. Therefore, this section discusses only aspects such as usefulness of the aids, when to use them, and requirements or precautions for their use.

The most important hardware aids available via the operator's console for system service and program diagnosis are:

- Alter/display feature.
- Instruction stepping feature.
- Stop on address compare feature.

CAUTION: When using one of the above mentioned hardware serviceability and debugging aids, you interfere with normal processing under VSE. Therefore, you should consider using these aids only (with your local management approval) in situations such as total system failure or a hard wait condition with no VSE-supported recovery possible.

Hardware Alter/Display

With the alter/display feature you can display the contents of storage areas and registers as indicated below. You can also alter any of these storage areas.

Note: The alter/display feature can be used only from the operator's console of your processor; the feature is not available, for example, from a channel-attached 3277 that you use as an operator's console.

Following is a list of storage areas (and registers) that you can display and alter by using this hardware aid:

- Any selected area of real or virtual storage.
- Contents of the general purpose registers.
- Contents of the floating point registers.
- Contents of the control registers.
- Current PSW.
- Storage protection key.

For detailed information on how to use this feature and on the areas that you can display or alter from your processor's console, refer to IBM's operating procedures manual for your central processor.

Instruction Stepping Feature

With the instruction stepping feature you can check and record the address of each instruction that is executed during program operation. By combined application of this feature and the alter/display feature, you can trace, for example, a short program loop. This approach of tracing executable code of a program is indicated when only short sections of code are to be traced or if, for any reason, the SDAID tracing facility cannot be used.

Refer to the operations manual for your processor for details of this feature.

Stop-on-Address-Compare Feature

This feature is provided primarily for IBM service personnel. It enables you, for example, to stop all system activity at a selected instruction address within a program. In combination with the alter/display feature (or commands ALTER, DSPLY, or DUMP), the stop-on-address-compare feature allows you to display or alter the contents of storage at this selected address. You can use the feature, for example, if there is a need for a dump of a specific area of virtual storage at a specific point of program execution.

Another use of this feature is the generation of a sync signal at a certain instruction address (this is primarily a hardware service aid).

If your installation uses a /370 Model 145 or 148, the stop-on-address-compare feature provides an additional function: the data compare trap. With this function you can locate the instruction that modified a byte in storage to a predefined pattern. When a store operation modifies the applicable byte in storage to the value set in switches A and B on the processor's console, the processor stops, and you can now display the current PSW. To find the address of the modifying instruction, subtract the instruction's length code from the instruction address in that PSW.

If your installation uses a /370 Model 158, the feature referred to in the operating procedures manual as "SAR (storage address register) compare select and set," provides additional flexibility in setting up stop or sync conditions. For more information refer to the operating procedures manual for your processor.

INDEX

D

E

F

G

H

I

J

K

L

N

P

S

T

U

V

W

Numerics

VSE/Advanced Functions
Service Aids
Order No. SC33-6195-1

READER'S
COMMENT
FORM

This form may be used to communicate your views about this publication. They will be sent to the author's department for whatever review and action, if any, is deemed appropriate. Comments may be written in your own language; use of English is not required.

IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation whatever. You may, of course, continue to use the information you supply.

Note: *Copies of IBM publications are not stocked at the location to which this form is addressed. Please direct any requests for copies of publications, or for assistance in using your IBM system, to your IBM representative or to the IBM branch office serving your locality.*

Possible topics for comments are:

Clarity   Accuracy   Completeness   Organization   Coding   Retrieval   Legibility

If you wish a reply, give your name and mailing address:

____

____

____

What is your occupation? ________________________

Number of latest Newsletter associated with this publication: ________________

Thank you for your cooperation. No postage stamp is necessary if mailed in the U.S.A. (Elsewhere, an IBM office or representative will be happy to forward your comments or you may mail directly to the address in the Edition Notice on the back of the title page).

Note: Staples can cause problems with automated mail sorting equipment.
Please use pressure sensitive or other gummed tape to seal this form.

SC33-6195-1

Reader's Comment Form

Fold And Tape | Please Do Not Staple | Fold And Tape

NO POSTAGE
NECESSARY
IF MAILED
IN THE
UNITED STATES

BUSINESS REPLY MAIL

FIRST CLASS PERMIT NO. 40 ARMONK, N.Y.

POSTAGE WILL BE PAID BY ADDRESSEE:

International Business Machines Corporation
Department 6R1
180 Kost Road
Mechanicsburg, PA 17055

Fold And Tape | Please Do Not Staple | Fold And Tape

IBM®

Cut or Fold Along Line

VSE/Advanced Functions Service Aids (File No. S370/4300-40) Printed in U.S.A. SC33-6195-1

IBM

SC33-6195-1