GC28-1147-3
File No. S370-36

Program Product

MVS/Extended Architecture System Programming Library: User Exits

MVS/System Product:

JES3 Version 2 5665-291
JES2 Version 2 5740-XC6

IBM

Fourth Edition (June, 1987)

This major revision applies to Version 2 Release 2.0 and to all subsequent releases of MVS/System Product (5665-291 or 5740-XC6) until otherwise indicated in new editions or Technical Newsletters.

See the Summary of Amendments following the contents for a list of the changes made to this manual.

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 Bibliography*, GC20-0001, for the editions that are applicable and current.

Preface

This book contains information about the user exits provided in the base control program (BCP) and various components of MVS/XA. System programmers who create and maintain user-supplied routines can use this book with the *MVS/XA SPL: System Modifications* and *MVS/XA SPL: System Macro and Facilities.* Functional areas where such modifications are primarily used are syntactical checking of input records as well as an installation's statistical accounting retrieval function, device allocation, and for specific performance needs. This book assumes that the user has an extensive knowledge of both MVS/XA and any special installation requirements in these areas.

This book consists of three sections, which include the following information:

- Section 1 - introduces the subject of exit points through an overview description of why and how you would use these exit points.
- Section 2 - documents the user exits that are in the BCP of MVS/XA.
- Section 3 - lists the user exits that are located throughout the components of MVS/XA.

Information about these exit points is available in a number of publications. Section 3 refers to those exits that remain in their original books. The following is a reference list of books that contain either user exit information or support information for one or more user exits.

- *MVS/Extended Architecture: System Management Facilities (SMF)*, LC28-1153
- *MVS/Extended Architecture: Service Aids*, GC28-1159
- *MVS/Extended Architecture Resource Measurement Facility (RMF) - Reference and User's Guide*, LC28-1138
- *A Guide to Using Multiple Virtual Storage Interface Facilities - Student Text*, SR20-4675
- *MVS/Extended Architecture: JES3 User Modifications and Macros*, SC23-0060
- *MVS/Extended Architecture: JES3 User Modifications and Macros*, LC28-1372
- *MVS/Extended Architecture: JES2 User Modifications and Macros*, LC23-0069

- *Advanced Communications Function for VTAM Version 2 Planning and Installation Reference*, SC27-0610
- *MVS/Extended Architecture: Supervisor Services and Macro Instructions*, GC28-1154
- *MVS/Extended Architecture Planning: Global Resource Serialization*, GC28-1062
- *MVS/Extended Architecture Data Areas (MVS and JES2) - Microfiche*, LYB8-1191
- *MVS/Extended Architecture Data Areas (MVS JES3) - Microfiche*, LYB8-1195
- *MVS/Extended Architecture Magnetic Tape Labels and File Structure Administration*, GC26-4003 (DFP Version 1), GC26-4145 (DFP Version 2)
- *MVS/Extended Architecture Message Library: Routing and Descriptor Codes*, GC28-1194
- *MVS/Extended Architecture: SYS1.LOGREC Error Recording*, GC28-1194
- *MVS/Extended Architecture: TSO*, GC28-1173

  Supplement for MVS/Extended Architecture with TSO/E, SD23-0267
- *MVS/Extended Architecture: System Macros and Facilities - Volume 1*, LC28-1150
- *MVS/Extended Architecture: System Macros and Facilities - Volume 2*, LC28-1151
- *MVS/Extended Architecture: 31-Bit Addressing*, GC28-1158
- *MVS/Extended Architecture: System Modifications*, GC28-1152
- *MVS/Extended Architecture Data Administration Guide*, GC26-4013 (DFP Version 1), GC26-4140 (DFP Version 2)
- *MVS/Extended Architecture: System-Data Administration*, GC26-4010 (DFP Version 1), GC26-4149 (DFP Version 2)
- *MVS/Extended Architecture: VSAM Administration: Macro Instruction Reference*, GC26-4016 (DFP Version 1), GC26-4152 (DFP Version 2)
- *MVS/Extended Architecture Data Administration: Utilities*,GC26-4018 (DFP Version 1), GC26-4150 (DFP Version 2)
- *MVS/Extended Architecture VSAM Administration Guide*, GC26-4015 (DFP Version 1), GC26-4151 (DFP Version 2)
- *Advanced Communications Function for TCAM Version 2 Networking Installation Guide*, SC30-3153

- *Advanced Communications Function for TCAM Version 2 Installation Reference*, SC30-3133
- *Advanced Communications Function for TCAM Version 2 Installation Sample Programs*, SC30-3134
- *Advanced Communications Function for TCAM Version 2 Utilities*, SC30-3138
- *Advanced Communications Function for TCAM Version 2 Base Installation Guide*, SC30-3132
- *SPL: Resource Access Control Facility*, SC28-1343
- *MVS/Extended Architecture Interactive Problem Control System (IPCS) User's Guide and Reference*, GC28-1297
- *MVS/Extended Architecture Message Library: System Messages Volume 1*, GC28-1376
- *MVS/Extended Architecture Message Library: System Messages Volume 2*, GC28-1377

Contents

Figures

Summary of Amendments

Summary of Amendments
for GC28-1147-3
as Updated June, 1987

This major revision, which supports MVS/System Product Version 2 Release 2.0, contains updates to IEAVMXIT, incorporating new routing codes for CTXT. Some maintenance changes have also been made to the publication.

Summary of Amendments
for GC28-1147-2
as Updated December 27, 1985
By TNL GN28-1069

This Technical Newsletter, which supports the JES3 component of MVS/System Product Version 2 Release 1.5, contains updates to the User Exit Directory, including the addition of two user exits to the dictionary: IATUX61 and IATUX62. Some maintenance changes have also been made to the dictionary.

Summary of Amendments
for GC28-1147-2
as Updated December 21, 1984
By TNL GN28-0945

This Technical Newsletter which supports the JES3 component of MVS/System Product Version 2 Release 1.2, contains updated to the Preface and the User Exit Directory.

Section 1: Introduction

In certain areas of processing you might want to customize your MVS/XA system to a greater extent than that available by standard options such as initialization parameters and operator commands. You could make extensive changes to a MVS/XA system by directly altering the source code. However this approach is dangerous. For example, you would have to adopt the changes to any IBM-supplied maintenance code before applying the updates to your altered source code; and, during migration, there would be no simple way of transferring your alterations to a new release. To avoid such problems IBM offers MVS/XA user exit facilities.

There are exit points located throughout the MVS/XA system code. When one such exit is invoked, control passes to a user-supplied routine that does its special processing. At the end of the user routine, control returns to a specified point in the system code.

User Exit Types

There are three types of user exits in the MVS/XA system:

- **User Exit Point** - is an instruction in the source code where a user exit routine is called via the implementation of a calling macro. The parameters are passed to the macro via the implementation of keywords supplied with the initialization statements. These parameters, therefore, are loaded at IPL or START time of the component involved. You can modify the macro keywords via keywords/parameters supplied with an operating command. In most cases the initialization statement defaults are set to cause the exit point to be ignored.

- **User Exit Name List** - is similar to a user exit point in that at a position in the source code a macro is implemented. However, rather than directly calling the user exit routine, a DSECT is addressed which contains a list of user exit routine names. The user exit routines are then invoked sequentially as they appear in the name list. This processing is terminated in one of two ways:

  – when one of the user exit routines issues a terminating return code.
  – when the end of the name list is reached.

  The installation provides the routine names by updating, reassembling and relink-editing the name list DSECT. (It is possible to temporarily update the DSECT using AMASPZAP).

If the exit is not to be used, the name list contains blank entries.

In both the user exit point and user exit name list methods, the installation defines the user exit routine names.

- **Replaceable Module** - is a load module containing a CSECT that the user installation is authorized to update, alter or completely replace. The name of the routine and its library location are predefined. Some replaceable modules are IBM-supplied code such as IEALIMIT (see Section 2) and others are dummy CSECTs containing only a BR14 branch.

Link-Editing a User Exit Routine into a Library

This example shows how to make a user exit routine available to the system by link-editing it into a system library.

```
//LKUSRPGM     JOB     MSGLEVEL=(1,1)
//             EXEC    PGM=HEWL,PARM='XREF,LET,LIST,NCAL'
//SYSPRINT     DD      SYSOUT=A
//SYSUT1       DD      UNIT=SYSDA,SPACE=(TRK,10)
//SYSLMOD      DD      DSNAME=SYS1.LINKLIB,DISP=OLD
//SYSLIN       DD      *
    object deck
         NAME        EXITNAME(R)
/*
```

In this example:

`EXEC Statement`

invokes the linkage editor and requests maximum diagnostic listing.

`SYSPRINT DD Statement`

defines the message data set.

`SYSLMOD DD Statement`

defines the output data set, in this case the link library, SYS1.LINKLIB. The output data set can also be a permanent library to be invoked later by a JOBLIB or STEPLIB DD statement; in that case, the SYSLMOD DD statement could be coded as follows:

```
//SYSLMOD  DD   DSNAME=MYLIB,UNIT=3330,VOL=SER=666666,
//              DISP=(NEW,KEEP),SPACE=(1024,(20,2,1))
```

`SYSLIN DD Statement`

defines the input data set, in this example the object code for the user program.

```
NAME Control Statement
```

specifies the member name, and thus the program name, to be assigned to the user program. In this example, the member name is EXITNAME.

Programming Considerations for User Exit Routines

Each MVS/XA component has individual and separate programming conventions. Components that have similar functions have similar requirements. Most requirements, however, are unique. Because of these differences among component requirements, we cannot provide across the board rules for coding user exit routines. The guidelines we can give are:

- One of the major configuration changes with the extended architecture release of MVS is the 31-bit addressing.
- Whenever a macro is provided to perform a service, use the macro.
- Upon entry to your exit routine, save all register contents and restore them before returning to your calling routine. There are exceptions to this guideline, such as register 15, which, in many cases, must contain a return code upon return to the calling routine.
- Under no circumstances should you assume an interface (such as contents of a register) that is not specifically documented.

Section 2: User Exits

The user exits described in this section of the book are functions supported in the MVS/XA base control program (BCP) code. The documentation for additional exit functions will be moved to this book in future editions. The remainder of the user exit functions found in various MVS/XA components are listed in Section 3: User Exit Directory.

ADYPSTD

Functional Description - (DAE Post-Dump Routine)

ADYPSTD is a DAE post-dump routine. ADYPSTD takes data created by the pre-dump module, ADYPRED, and schedules a transaction on the DAE transaction processor queue whenever an SDUMP dump or a SYSMDUMP dump completes or is suppressed.

The input to this exit routine consists of register 1 pointing to a parameter list that contains the address of the SVC dump exit parameter list (SDEPL). The SDEPL contains pointers to the pre-dump parameter data (DSPD) and the dump header record (AMDDATA).

The ADYPSTD exit routine analyzes the input to determine which transaction entries are required and which are to be built. If any transactions are required, ADYPSTD:

- Builds transaction entries that cause the transaction processor to update the DAE symptom data set with problem data and, if a new problem is found, to add an entry to the symptom queue.
- Places the transaction entry on the DAE transaction queue and posts the transaction processor.

Entry Specifications

Upon entry to the ADYPSTD routine, the register contents are as follows:

Register	Contents
0	irrelevant
1	address of a parameter list which contains the address of the SDEPL
2-12	irrelevant
13	address of a standard save area
14	return address
15	entry point address

Return Specifications

Upon return from the ADYPSTD routine, the register contents must be as follows:

Register	Contents
0-14	unchanged
15	return code

Programming Considerations

The DAE post-dump exit routine is called from IEAVTSEP as the last exit in the post-dump processing exit list IEAVTSEL.

ADYPSTD receives control when either an SDUMP dump or a SYSMDUMP dump finishes processing or when these dumps are suppressed by DAE. It establishes an ESTAE (label ADYPSTDR) exit recovery environment that:

- Takes an SDUMP for all unexpected errors, except errors that are potentially recursive.
- Retries whenever possible to cause a normal return to IEAVTSEP.
- Frees the local storage that is not task-related.
- Releases serialization (DEQ) from the transaction processor resource.

After scheduling all required transactions, ADYPSTD deletes the ESTAE recovery environment and returns to the caller.

Operational Considerations

The attributes of the user exit routine are:

- RMODE (ANY)
- Supervisor state
- Enabled
- AMODE (31)
- Key 0

IEALIMIT

Functional Description - (Limiting User Region Size)

An installation can enforce a region-size limit by writing an exit routine that is invoked once per step when the initiator is establishing region size. IEALIMIT can be used under MVS/XA to set non-extended region size and region limit only. The values set by IEALIMIT should be less than the size of the non-extended private area. If they are not, the control program uses the size of the non-extended private area.

If the step specifies a REGION value less than 16 megabytes, the non-extended region size and region limit are determined by the values set by IEALIMIT. The extended region size and limit are set by default. If the step specifies a REGION value greater that 16 megabytes, the non-extended region size and region limit is the smaller of the value set by IEALIMIT and the size of the non-extended private area. The extended region size and limit are determined by the value specified on the REGION parameter.

The values set by IEALIMIT have no effect on establishing the extended region size and limit. It is recommended that the exit routine IEFUSI be used to monitor jobs that specify a REGION value that is greater than 16 megabytes. See *MVS/XA: System Modifications* for more information related to IEALIMIT and IEFUSI.

Entry Specifications

Upon entry to the IEALIMIT routine, the register contents are as follows:

Register	Contents
0	If the REGION value specified is less that 16 megabytes, register 0 contains the number of bytes requested by the application program for its region (specified explicitly through the REGION parameter or implicitly through the default JCL value) If the REGION value specified is greater than 16 megabytes, register 0 contains a value that is equal to the size of the non-extended private area minus 64K.
1	same as register 0
13	address of standard save area
14	return address
15	entry point address for the IEALIMIT routine

Return Specifications

Upon return from the IEALIMIT routine, the register contents must be as follows:

Register	Contents
0	number of bytes to be used as the region parameter value. This number should be less than the size of the non-extended private area.
1	number of bytes to be used as the limit on all types of requests from subpool 0-127, 251 and 252. This number should be less than the size of the non-extended private area.
2-13	restored
14	return address
15	irrelevant

If register 1 is non-zero when the IBM-supplied IEALIMIT receives control, IEALIMIT adds 64K to the contents of register 1 and returns.

Register 0 remains unchanged. The register 1 value is used to limit the total allocation of storage from subpools 0-127, 251 and 252.

If the IBM-supplied IEALIMIT routine receives control and the input register 1 contains a zero, then IEALIMIT returns a zero in register 1 and no limit is assigned (to a job, a started program, or a TSO user). No limit is set only when the REGION parameter is not specified and the default value is zero.

Programming Considerations

When no limit is set, so much space within a region might be obtained (via repeated small GETMAINs or a single large GETMAIN) that no space remains in the private area for the system to use. This situation is likely to occur when a variable GETMAIN is issued that specifies such a large maximum value that most or all of the space remaining in the private area is allocated to the requestor. Therefore, it is strongly recommended that a region size be specified on the JOB or EXEC statement, or that the default region size for the job class not be zero.

After the IEALIMIT routine determines the appropriate limit, it must pass back to IGVGVRGN, via register 1, a numeric value that represents the imposed limit in bytes. As noted above, a zero returned in register 1 indicates that a limit is not imposed. The IEALIMIT routine should pass back, in register 0, a value that is less than the value in register 1. Both register 0 and 1 should be rounded to a multiple of 4K. IGVGVRGN stores register 0 as the REGION parameter value and register 1 as the IEALIMIT value for future reference (that is, for use in processing subsequent GETMAINs as described below).

The REGION parameter value (register 0) should be less than the IEALIMIT value (register 1) to protect against programs that issue variable length GETMAINs with very large maximums, and then do not immediately free part of that space or free such a small amount that a subsequent GETMAIN (possibly issued by a system service) causes the job to fail. As an example, suppose that a program issues a variable-length GETMAIN with maximum of 2^{31}-1 bytes. If the REGION parameter value is not less than the IEALIMIT value, all the space in the region up to the IEALIMIT value is allocated, and any subsequent GETMAIN that cannot be satisfied from free space in an already existing subpool causes the job to fail. If however, the REGION parameter value is made less than the IEALIMIT value, only space up to the REGION parameter value is allocated for the GETMAIN. Thus, an amount of space equal to the IEALIMIT value minus the REGION parameter value remains for subsequent GETMAINs.

The REGION parameter value minus the amount of storage currently allocated specifies the maximum amount of storage that can be allocated to a job by any single variable-length GETMAIN request. The IEALIMIT value specifies the maximum total storage that can be allocated to a job by any combination of GETMAINs. The relationship between the REGION parameter value and the IEALIMIT value and their effect upon both fixed-length and variable-length GETMAINs is shown in Figure 1.

Type of GETMAIN	**GETMAIN Request in Relation To Region Size & Limit Value**	**Result**
FIXED-LENGTH GETMAIN	Limit value minus currently - allocated space ≥ requested amount.	The GETMAIN is satisfied
	Limit value minus currently - allocated space < requested amount.	The GETMAIN fails
VARIABLE-LENGTH GETMAIN	Unallocated space[1] ≥ maximum amount requested.	The maximum amount is allocated.
	Minimum amount requested ≤ unallocated space, AND unallocated space ≤ maximum amount requested.	All unallocated space in the region is allocated.
	Unallocated space[1] ≤ minimum amount requested	The minimum is allocated unless the limit value would be exceeded, in which case the GETMAIN fails.
[1] Unallocated space is the region size minus the currently - allocated space.		

Figure 1. The Effects of IEALIMIT and REGION Values on Various GETMAINs

For example, assume that application program A has the following characteristics:

```
IEALIMIT value   150K
REGION-parameter value  100K
Currently allocated space  80K
```

Program A issues the following variable-length GETMAIN requests in the order indicated:

1. Request 5K-10K: 10K is allocated; currently allocated space is now 90K. Because the amount currently allocated (80K) does not exceed the REGION-parameter value (100K) and because the amount unallocated (20K - relative to the REGION-parameter value) is greater than the maximum amount requested (10K), the maximum is allocated.

2. Request 5K-100K: 10K is allocated; currently allocated space is now 100K. Because the amount unallocated (10K - relative to the REGION-parameter value) is between the minimum and maximum, the amount unallocated is allocated.

3. Request 40K-100K: 40K is allocated; currently allocated space is now 140K. Because the amount unallocated (0K - relative to the REGION-parameter value) is less than the minimum amount requested (40K), the minimum amount is allocated.

4. Request 15K-50K: the GETMAIN request fails. The amount unallocated (0K - relative to the REGION-parameter value) is less than the minimum amount requested (15K). If the minimum amount were allocated, the currently allocated amount would become 155K, which exceeds the IEALIMIT value (150K). Therefore, the request fails.

IBM-Supplied IEALIMIT Exit Routine

The following code is supplied with the BCP. It is link-edited with the nucleus. The exit is called by module IGVLIMIT.

```
* Save callers registers and establish addressability
                      STM         14,15,12(13)
                      STM         2, 12, 28(13)
                      BALR        9,0
* If this is not a request for an unlimited
* region, then increase the region limit by 64K.
* The region size is not changed.
                      LTR         1,1
                      BZ          EXIT
                      AL          1, INCRMENT
* Restore callers registers and return to caller.
  EXIT                LM          14,15,12(13)
                      LM          2,12,28(13)
                      BR          14
  INCRMENT            DC          F'65536'
```

Operational Considerations

- Function - set the region size and limit values for the non-extended private area.
- Environment - synchronous, enabled, AMODE = 24, KEY = 0, state = supervisor, mode = TCB, HOME address space.
- Recovery - run under an ESTAE established by IGVGVRGN or IGVGRRGN.
- Receives control - after an existing region has been freed and before the new region is initialized.
- Locks Held - local lock.
- Address Space - any.
- Task - initiators task.

Restrictions and Limitations

- Many subsystems request a region size of 0 (unlimited). If this exit is used to change such a subsystem request, unreliable results could be produced.
- If this exit routine requires dynamic storage, use subpool 229 or 230.

IEAVADFM

Functional Description - (Format SNAP, SYSABEND and SYSUDUMP Dumps)

This is a dump facility user exit routine name list. A user exit routine invoked from ABDUMP (SNAP/ABEND dump) allows an installation to gather information to be included in a SNAP/ABEND dump and format it to a data set described by a SYSABEND, SYSUDUMP or user-defined JCL DD statement.

The user exit routines named in IEAVADFM receive control automatically during the control block formatting phase of every SNAP or ABEND dump for which the CB option was requested. The user exit routines are given control in the AMODE of the exit. The exit provides an area in which the installation's exit routines can build a print line and the address of an IBM-supplied print routine to which the installation exit routine can pass the line for printing.

Entry Specifications

Upon entry to a user exit routine specified in the IEAVADFM name list, the register's contents are as follows:

Register	Contents
0	irrelevant
1	points to a parameter list
2-12	irrelevant
12	points to a standard 18-word save area
14	return address
15	formatting routine entry point address

The parameter list pointed to by general register 1 contains the following information that is useful for user formatting routines:

0	ADPLTCB - TCB of task being displayed
4	ADPLASID - Address space identifier
6	ADPLSBPL - Subpool used to get save area by component routine
7	ADPLFLAG - flag field
	ADPLSNPR - 0-Module loaded by SNAP
	1-Module loaded by PRDMP/IPCS
	ADPLSYTM - 0-System is OS/VS2
	1-System is OS/VS1
	ADPLDMGT - For data management formatter under SNAP:
	0--Format DEB only,
	1-Format DEB, DCB, IOB
	ADPLIPCS - Called by IPCS
	ADPLPRT - SYSPRINT data set
	ADPLSYNO - Exit given control for syntax checking only
	ADPLEJEC - Page eject
8	ADPLBUF - Pointer to output buffer
C	ADPLPRNT - Address of print routine
10	ADPLCVT - Address of CVT
14	ADPLMEMA - Address of memory access routine
18	ADPLFRMT - Address of format routine
1C	ADPLCOM1 - Reserved for component use
20	ADPLCOM2 - Reserved for component use
24	ADPLCOM3 - Reserved for component use
28	ADPLCOM4 - Reserved for component use
2C	ADPLFMT2 - Reserved for format routine
30	ADPLFMT1 - Reserved for format routine
34	ADPLEXT - Address of extension whose first word contains the address of the operands list or zero if none
38	ADPLABDA - Address of host internal parameter list
3C	ADPLTRFM - Address of trace control block (SNAP only)
40	reserved
44	ADPLLEV - Index indentation level
45	ADPLIDX - Entry code number corresponding to AMDMNDXT macro entries
46	ADPLLNCT - Lines per page
48	ADPLLNRM - Lines remaining on the current page
4A	ADPLDLEN - Storage access length
4C	ADPLOPLN - Length of operands
4E	ADPLPRDP - Address qualification
	ADPLVIRT - Virtual address
	ADPLREAL - Real address
	ADPLCPU - CPU data request
	ADPLHDR - Dump header request

The following bit governs the masking of register zero prior to its use by the storage access service as a virtual storage address. If it is off, X'7FFFFFFF' will be logically ANDed with register zero to obtain the requested address. If it is on, X'00FFFFFF' will be logically ANDed with register zero to obtain the requested address (PRDMP/IPCS only).

4F	ADPLSAMK - MVS/370 virtual address reserved
50	ADPLNDX - Address of the AMDPRNDX routine. Build index work entries
54	ADPLPGNO - Current page number
58	reserved

Note: This parameter list, is mapped by either the IHAABDPL or the BLSABDPL system mapping macro. The mapping list includes all the fields of the PRDMP service aid's parameter list so user formatting routines can be invoked by either SNAP/ABEND or print dump. The memory access service routine and the index routine are no-op functions for SNAP/ABEND. The memory access routine will check for a valid storage request. If the request is valid, the routine will zero register 15 and return. If the request is not valid, a return code of four is set in register 15. The index routine will clear the output buffer (ADPLBUFR) and return.

The exit receives control in protection key 0, supervisor state with no locks held.

Upon return from the user exit routine, the registers must contain:

Register	**Contents**	
0-14	same as on entry	
15	a return code, interpreted as follows:	
	0	continue processing
	12	suppress remainder of ABEND dump.

You should be aware that if the same user exit routine is executed under the PRDMP service aid, the print dump does not suppress the remainder of the dump for a return code of 12. The return should be made in protection key 0, supervisor state, with no locks held (the same state as when entered). For an example of a formatting routine, see module IEAVTFMT in your source code microfiche.

Programming Considerations

The user exit routines must observe the following programming conventions:

- The routines must be reentrant.

- One or more exit routines can be link-edited into SYS1.LPALIB, SYS1.LINKLIB or into a LNKLST library with any load module name. The installation must also add the load module's name to the IEAVADFM list of user exits. IEAVADFM is a CSECT in load module IGC0805A, and each entry is eight characters, padded to the right with blanks. If an entry is to be ignored, then the installation must change it to eight blanks. The end of the list is indicated by four bytes of hexadecimal zeroes. The IBM-supplied version of IEAVADFM has nine words of hexadecimal zeroes.

- In order to avoid an abnormal termination later in the SNAP/ABEND routine, the user's routines must not free either the entry parameter list or the print buffer.

- Recovery for SNAP is provided by an ESTAE routine. Each user formatting routine should set up its own recovery to handle any ABENDs encountered during the formatting process. This routine should either recover and continue formatting, or recover and return to SNAP with a zero return code. A non-zero return code is interpreted as a GETMAIN failure, resulting in the following message in the dump data set:

  ```
  USER/PP CONTROL BLOCKS UNAVAILABLE
  ```

  The dump is truncated because of lack of storage.

- The recovery routine should not continue formatting if an X'x37' ABEND occurs, because no space remains in the dump data set. Before the recovery routine returns to SNAP, it should free all the storage that it has obtained.

- If the formatting routine does not establish recovery, or if the recovery exit specifies continue-with-termination after an ABEND, SNAP terminates this control block formatter entirely and continues with the next portion of the dump, if any.

Interface to the print routine:

Entry: via BALR 14, 15 for each line to be written.

Environment: PSW key 0, supervisor state, no locks held.

Registers on entry:

Register 1	points to the parameter list
Register 13	points to the save area
Register 14	points to the return address
Register 15	points to the EPA of the print routine

Registers saved and restored: 14 through 12.

Return: via register 14.

To remove a formatting routine from the system, remove the entry from the IEAVADFM list of user exit routines by replacing the entry with eight blanks.

Operational Considerations

The user's formatting routine should build one print line at a time in the buffer provided, and should use BALR or call to the IBM-supplied print routine, which in turn prints the line on the dump data set. Offsets are recommended for all formatted control blocks that are longer than one output line. (One line generally formats 20 hexadecimal characters.) The print routine saves registers, prints the line, blanks the buffer, restores the registers, and returns control to the user's routine via register 14.

Because it works through the IBM-supplied print routine, the formatting routine has no direct access to the carriage controls. In order to cause a skipped line in the dump output, it is therefore necessary to pass a blank buffer to the print routine. Similarly, the print routine also handles page ejects. The user exit routine can use format patterns to format data in the output buffer, by using the IBM-supplied format service routine. The service routine can also convert data to printable hexadecimal. This service routine is the same routine as is provided by the PRDMP service aid, and the details of its interface are given in *MVS/XA SPL: Service Aids.*

IEAVADUS

Functional Description - (Select and Format Dump Data)

The IEAVADUS user exit provides the user with the ability to select and format data to be included in an ABDUMP (SNAP/ABEND dump) to a data set described by a SYSABEND, SYSUDUMP or a user-defined DD JCL statement.

This user exit routine is entered from module IEAVAD08 in the AMODE of the user exit routine. The user exit routine receives control automatically during the control block formatting phase of every SNAP and ABEND dump for which the CB option was requested. The exit provides an area in which the installation's routine can build a print line and the address of an IBM-supplied print routine to which the installation routine can pass the line for printing.

Entry Specifications

Upon entry to the user exit routine the register's contents are as follows:

Register	Contents
0	irrelevant
1	points to a parameter list
2-12	irrelevant
13	points to a standard 18-word save area
14	address to which the formatting routine should return control.
15	formatting routine entry point address

The parameter list pointed to by general register 1 contains the following information that is useful for user formatting routines.

0	ADPLTCB - TCB of task being displayed
4	ADPLASID - Address space identifier
6	ADPLSBPL - subpool used to get save area by component routine
7	ADPLFLAG - flag field
	ADPLSNPR - 0-Module loaded by SNAP
	1-Module loaded by PRDMP/IPCS
	ADPLSYTM - 0-System is OS/VS2
	1-System is OS/VS1
	ADPLDMGT - For data management formatter under SNAP:
	0--Format DEB only,
	1-Format DEB, DCB, IOB
	ADPLIPCS - Called by IPCS
	ADPLPRT - SYSPRINT data set
	ADPLSYNO - Exit given control for syntax checking only
	ADPLEJEC - Page eject
8	ADPLBUF - Pointer to output buffer
C	ADPLPRNT - Address of print routine
10	ADPLCVT - Address of CVT
14	ADPLMEMA - Address of memory access routine
18	ADPLFRMT - Address of format routine
1C	ADPLCOM1 - Reserved for component use
20	ADPLCOM2 - Reserved for component use
24	ADPLCOM3 - Reserved for component use
28	ADPLCOM4 - Reserved for component use
2C	ADPLFMT2 - Reserved for format routine
30	ADPLFMT1 - Reserved for format routine
34	ADPLEXT - Address of extension whose first word contains the address of the operands list or zero if none
38	ADPLABDA - Address of host internal parameter list
3C	ADPLTRFM - Address of trace control block (SNAP only)
40	reserved
44	ADPLLEV - Index indentation level
45	ADPLIDX - Entry code number corresponding to AMDMNDXT macro entries
46	ADPLLNCT - Lines per page
48	ADPLLNRM - Lines remaining on the current page
4A	ADPLDLEN - Storage access length
4C	ADPLOPLN - Length of operands
4E	ADPLPRDP - Address qualification
	ADPLVIRT - Virtual address
	ADPLREAL - Real address
	ADPLCPU - CPU data request
	ADPLHDR - Dump header request

The following bit governs the masking of register zero prior to its use by the storage access service as a virtual storage address. If it is off, X'7FFFFFFF' will be logically ANDed with register zero to obtain the requested address. If it is on, X'00FFFFFF' will be logically ANDed with register zero to obtain the requested address (PRDMP/IPCS only).

4F	ADPLSAMK - MVS/370 virtual address reserved
50	ADPLNDX - Address of the AMDPRNDX routine. Build index work entries
54	ADPLPGNO - Current page number
58	reserved

Note: This parameter list is mapped by either the IHAABDPL or the BLSABDPL system mapping macro. The parameter list includes all the fields of the PRDMP service aid's parameter list so user formatting routines can be invoked by either SNAP/ABEND or print dump. The memory access service routine and the index routine are no-op functions for SNAP/ABEND. The memory access routine will check for a valid storage request. If the request is valid, the routine will zero register 15 and return. If the request is not valid, a return code of four is set in register 15. The index routine will clear the output buffer (ADPLBUFR) and return.

The exit receives control in protection key 0, supervisor state with no locks held.

Return Specifications

Upon return from the user exit routine, the registers must contain:

Register	Contents	
0-14	same as on entry	
15	a return code, interpreted as follows:	
	0	continue processing
	12	suppress remainder of ABEND dump.

You should be aware that if the same exit routine is executed under the PRDMP service aid, the print dump does not suppress the remainder of the dump when it receives a return code of 12. The return should be made in protection key 0, supervisor state, with no locks held (the same state as when entered). For an example of a formatting routine, see module IEAVTFMT in your source code microfiche.

Programming Considerations

The user's formatting exit routine must observe the following programming conventions:

- The routines must be reentrant.
- The installation must link-edit the exit routine into SYS1.LPALIB, SYS1.LINKLIB or a LNKLST library. One user exit routine can be link-edited into SYS1.LPALIB with the load module name of IGC0905A replacing the IBM-supplied routine IEAVADUS. The SNAP/ABEND dump routine CSECT IEAVAD08 of load module IGC0805A loads IGC0905A, and then branches and links to it.
- In order to avoid an abnormal termination later in the SNAP/ABEND routine, the user's routines must not free either the entry parameter list or the print buffer.

- Recovery for SNAP is provided by an ESTAE routine. Each user formatting routine should set up its own recovery to handle any ABENDs encountered during the formatting process. This routine should either recover and continue formatting, or recover and return to SNAP with a zero return code. A non-zero return code is interpreted as a GETMAIN failure, resulting in the following message in the dump data set:

  ```
  USER/PP CONTROL BLOCKS UNAVAILABLE
  ```

The dump is truncated because of lack of storage. The recovery routine should not continue formatting if an X'x37' ABEND occurs, because no space remains in the dump data set. Before the recovery routine returns to SNAP, it should free all the storage that it has obtained.

If the formatting routine does not establish recovery, or if the recovery exit specifies continue-with-termination after an ABEND, SNAP terminates this control block formatter entirely and continues with the next portion of the dump, if any.

Interface to the print routine:

Entry: via BALR 14, 15 for each line to be written.

Environment: PSW key 0, supervisor state, no locks held.

To remove the formatting routine from the system, link-edit a copy of module IEFBR14 into SYS1.LPALIB with the name IGC0905A.

Operational Considerations

The user's formatting routine should build one print line at a time in the buffer provided, and should use BALR to branch to the IBM-supplied print routine, which in turn prints the line of the dump data set. Offsets are recommended for all formatted control blocks that are longer than one output line. (One line generally formats 20 hexadecimal characters). The print routine saves registers, prints the line, blanks the buffer, restores the registers, and returns control to the user's routine via register 14. No registers are necessary as input to this routine.

Because it works through the IBM-supplied print routine, the formatting routine has no direct access to the carriage controls. In order to cause a skipped line in the dump output, it is therefore necessary to pass a blank buffer to the print routine. Similarly, the print routine also handles page ejects. The user exit routine can use format patterns to format data in the output buffer, by using the IBM-supplied format service routine. The service routine can also convert data to printable hexadecimal. This service routine is the same routine as is provided by the PRDMP service aid, and the details of its interface are given in *MVS/XA SPL: Service Aids*.

IEAVTABX

Functional Description - (Change Options/Suppress Dump)

This is a SYSUDUMP/SYSABEND dumping services user exit routine name list. The installation can use this exit to change the dump options in effect or to suppress the dump that would be generated by an ABENDing task.

The user exit routines implemented by this exit facility receive control in key 0, supervisor state. They must be AMODE (31) and reside in (E)PLPA and their load module names (8 byte names/padded to right with blanks) must be included in the list in CSECT IEAVTABX, load module IEAVTABX. (The count field in the CSECT must reflect the number of entries in the list.) A zero entry (8C'00000000') must be the end of list indicator. The IEAVTABX load module CSECT is shown in Figure 2.

The user exit routine should establish an ESTAE for recovery. The user exit routine will receive control in 31-bit addressing mode and in an enabled state with no locks held. The input parameter list (IHAABEPL) is the major communication area and contains a copy of the following information for each dump:

- Job name
- ABEND code
- Address of the SDWA
- Module name
- Options in effect (parameter list)
- Parameter list level indicator
- Return code from the previous user exit routine

The user exit routine runs under the ABENDing task and in the current address space. The user exit routines are given control prior to taking the dump. If one of the user exit routines suppresses the dump, message IEA848I is issued, indicating that the dump suppression has taken place.

Entry Specifications

Upon entry to a user exit routine specified in the IEAVTABX name list, the register's contents are as follows:

Register	Contents
0	irrelevant
1	address of IHAABEPL
2-12	irrelevant
13	address of save area
14	return address
15	entry point address

Return Specifications

Upon return from the user exit routine the registers must contain:

Register	Contents	
0-14	same as on entry	
15	a return code, which is interpreted as follows:	
	0	continue processing with current options
	4	change options as indicated in IHAABEPL
	8	suppress the dump

Programming Considerations

The user exit routine must observe the following programming conventions:

- The number of user exit routines that IEAVTABX gives control can affect the amount of time it takes to complete the dump. Each user exit routine must establish an ESTAE and request a tailored dump if any errors occur during its processing.

  Note: A request for a ABDUMP will cause recursion and no dump will be produced for the user exit routine error. It is suggested that you take an SDUMP and issue a SETRP DUMP=NO. This will result in a retry attempt to a return point in order to prevent ABDUMP's recursion recovery routine from getting control.

- Obtain storage from specific subpools and these subpools should be requested in the tailored dump.

- Before returning control to ABDUMP, delete the ESTAE and free all storage the user exit routine obtained.

- The user exit routine must reside in (E)PLPA with AMODE (31).

- The user exit routine must not free the entry parameter list.

- When adding a user exit routine take the following steps:
  - Link-edit exit routine
  - Add the load module name for the user exit routine to the list in CSECT IEAVTABX.
  - Update the count field in IEAVTABX to reflect the number of user exit routines provided.
  - Ensure a zero entry (8C'00000000') to indicate the end of list.
  - Assemble and relink-edit IEAVTABX into the system.
  - IPL the system.

Note: The AMASPZAP can be used to update the IEAVTABX CSECT.

Operational Considerations

The attributes of the user exit routine are:

- Supervisor state
- Key 0
- AMODE (31)

```
  TABX      IEAVTABX TABLE CSECT FOR LOAD MODULE IEAVTABX                                                  PAGE    2

    LOC  OBJECT CODE    ADDR1 ADDR2  STMT   SOURCE STATEMENT                                   ASM H V 06 19.00 04/22/82
                                        2 * /* START OF SPECIFICATIONS ****                                     00100000
                                        3 *                                                                     00150000
                                        4 *     MODULE-NAME = IEAVTABX                                          00200000
                                        5 *                                                                     00250000
                                        6 *     DESCRIPTIVE-NAME = TABLE OF EXITS TO BE TAKEN BY ABDUMP         00300000
                                        7 *                                                                     00350000
                                        8 *     COPYRIGHT = 5740-XC6 COPYRIGHT IBM CORP 1980, 1981,             00375000
                                        9 *                 LICENSED MATERIAL-PROGRAM, PROPERTY OF IBM,         00400000
                                       10 *                 REFER TO COPYRIGHT INSTRUCTIONS FORM NUMBER G120-2083 00425000
                                       11 *                                                                     00450000
                                       12 *     STATUS = OS/VS2 HBB2102                                         00500000
                                       13 *                                                                     00550000
                                       14 *     LOAD MODULE = IEAVTABX                                          00600000
                                       15 *                                                                     00650000
                                       16 *     FUNCTION = CONTAIN NAME(S) OF EXIT(S) TO BE TAKEN.              00675000
                                       17 *                                                                     00700000
                                       18 *     OPERATION= USER CAN ADD/REPLACE ANY EXIT WITH THEIR OWN         00706200
                                       19 *                INSTALLATION EXIT. EXITS WILL BE TAKEN BY THE        00709900
                                       20 *                SPECIFIED ORDER.  THE COUNT FIELD MUST BE UPDATED TO 00713600
                                       21 *                REFLECT THE NUMBER OF EXIT ROUTINE NAMES IN THE      00717300
                                       22 *                TABLE.  AN ENTRY CONTAINING ALL ZEROES MUST INDICATE 00719800
                                       23 *                THE END OF THE LIST.                                 00722300
                                       24 *                                                                     00725000
                                       25 *     NOTES =                                                         00750000
                                       26 *                                                                     00800000
                                       27 *       CHARACTER-CODE-DEPENDENCIES = THIS MODULE IS EBCDIC CHARACTER 00880000
                                       28 *                                     CODE DEPENDENT.                 00900000
                                       29 *                                                                     00950000
                                       30 *       DEPENDENCIES = NONE                                           01000000
                                       31 *                                                                     01050000
                                       32 *       RESTRICTIONS = NONE                                           01250000
                                       33 *                                                                     01300000
                                       34 *       PATCH-LABEL = NONE                                            01350000
                                       35 *                                                                     01400000
                                       36 *     MODULE-TYPE = MODULE                                            01450000
                                       37 *                                                                     01500000
                                       38 *       PROCESSOR = ASSEMBLER                                         01550000
                                       39 *                                                                     01600000
                                       40 *       MODULE-SIZE =  VARIABLE                                       01650000
                                       41 *                                                                     01700000
                                       42 *       ATTRIBUTES = AMODE=31, RMODE=ANY,                             01733300
                                       43 *                    KEY 0, SUPERVISOR STATE, PLPA                    01766600
                                       44 *                                                                     01800000
                                       45 *     ENTRY = IEAVTABX                                                01850000
                                       46 *                                                                     01900000
                                       47 *       PURPOSE = SEE OPERATION.                                      01916600
                                       48 *                                                                     01933200
                                       49 *     EXTERNAL-REFERENCES                                             01950000
                                       50 *                                                                     02000000
                                       51 *       ROUTINE = NONE                                                02025000
```

Figure 2 (Part 1 of 2). CSECT for Load Module IEAVTABX

```
TABX     IEAVTABX TABLE CSECT FOR LOAD MODULE IEAVTABX                                                  PAGE    3

  LOC  OBJECT CODE    ADDR1 ADDR2  STMT   SOURCE STATEMENT                                  ASM H V 06 19.00 04/22/82

                                     52 *                                                                   02050000
                                     53 *      TABLES =                                                     02058300
                                     54 *        EACH ENTRY IN THE LIST OF MODULES HAS ONE OF TWO FORMATS:  02066600
                                     55 *          1) DC CL8'XXXXXXXX' NAME OF THE MODULE                   02074900
                                     56 *          2) DC CL8'        ' BLANKS INDICATING NULL ENTRY         02083200
                                     57 *        TO INDICATE THE END OF LIST USE:                           02088800
                                     58 *          1) DC CL8'00000000' ZEROES INDICATING LAST ENTRY         02094400
                                     59 *                                                                   02100000
                                     60 *      MACROS = NONE                                                02125000
                                     61 *                                                                   02150000
                                     62 *      CHANGE-ACTIVITY = NEW FOR HBB2102. ABDUMP PREDUMP EXIT SUPPORT.  02183300
                                     63 *                        02/11/81                                   02216600
                                     64 *                                                                   02250000
                                     65 **** END OF SPECIFICATIONS ***/                                     03700000
000000                               66 IEAVTABX CSECT ,                                                    03713300
                                     67 IEAVTABX AMODE 31                     31-BIT ADDRESSING MODE        03726600
                                     68 IEAVTABX RMODE ANY                    31-BIT RESIDENCE              03739900
000000 00000000                      69          DC    F'0'                   NUMBER OF EXITS TO BE TAKEN   03753200
000004 4040404040404040              70          DC    CL8'        '         USER EXIT ROUTINE MODULE @ZMC2101 03766500
00000C 4040404040404040              71          DC    CL8'        '              NAMES SHOULD BE           03779800
000014 4040404040404040              72          DC    CL8'        '                INSERTED IN THESE       03793100
00001C 4040404040404040              73          DC    CL8'        '                      AREAS.            03806400
000024 4040404040404040              74          DC    CL8'        '                                        03819700
00002C 4040404040404040              75          DC    CL8'        '                                        03833000
000034 4040404040404040              76          DC    CL8'        '                                        03846300
00003C 4040404040404040              77          DC    CL8'        '                                        03859600
000044 4040404040404040              78          DC    CL8'        '                                        03872900
00004C 4040404040404040              79          DC    CL8'        '                                        03886200
                                     80          MODID BRANCH=NO                                   @ZMC2101 03893100
000054 C9C5C1E5E3C1C2E7              81+         DC    CL8'IEAVTABX'          MODULE NAME          @G38DP11 01-MODID
00005C F0F461F2F261F8F2              82+         DC    CL8'04/22/82'          DATE                 @G38DP11 01-MODID
000064 4040404040404040              83+         DC    CL8' '                 VERSION (PTF NUMBER  @G38DP11 01-MODID
                                     84+*                                      OR PRODUCT)         @G38DP11
00006C                               85+         DS    0H                                          @ZA15314 01-MODID
000000                               86          END   IEAVTABX                                             03900000
```

Figure 2 (Part 2 of 2). CSECT for Load Module IEAVTABX

IEAVTSEL

Functional Description - (Post Dump Exit Name List)

IEAVTSEL is an SDUMP and SYSMDUMP post dump exit name list. It contains a list of module names to be given control as the dump processing ends. At the end of each SDUMP, each module in this list of installation exit routines gains control in order to perform post dump exit processing. The IEAVTSEL exit routine list is placed in SYS1.LINKLIB.

IEAVTSEL has 12-byte entries made up of an 8-byte name and a 4-byte flag field. The high order bit of the flag field indicates the conditions under which control is to be passed to the corresponding exit routine. If the bit is off (X'00000000'), the exit is only passed control when a dump has actually been taken. If the dump has been suppressed by dump analysis and elimination (DAE) the exit is not passed control. If the bit is set on (X'80000000'), the exit is passed control unconditionally. The list may contain blank entries (eight EBCDIC blanks, X'40') and is terminated by an entry of all zeros. The version of IEAVTSEL shipped with the MVS/XA system contains nine blank entries followed by an entry for ADYPSTD (DAE post-dump exit). This entry is followed by a terminating entry of zeroes. See ADYPSTD in this manual for more information concerning this post dump exit.

The exit routines specified by the installation in the IEAVTSEL name list are given control sequentially as they appear. The exit routines can reside in either SYS1.LINKLIB or SYS1.LPALIB. The input to exit routines consists of register 1 pointing to a parameter list that contains a pointer to the SDUMP exit parameter list (SDEPL) mapped by IHASDEPL in SYS1.MACLIB. The SDEPL contains the following fields:

Field Name	Length	Offset	Description
SDEPL	32	0	Exit PARMLIST
SDEPLCHA	4	0	EBCDIC identifier
SDEPLFLG	1	4	Exit status flags
SDEPLEXE			Bit on error occurred in preceding exit
SDEPLERR			Bit on error occurred in any previous exit
SDEPLRES	3	5	Reserved
SDEPLHD	4	8	Address of the dump header record mapped by the AMDDATA mapping macro
SDEPLWA	4	12	Address of exit work area (200 decimal bytes)
SDEPLEXT	4	16	Address of exit interface area
SDEPLEXL	4	20	Length of interface area
SDEPLDSP	4	24	Pointer to the DSPD.
	4	28	Reserved

Entry Specifications

Upon entry to a user exit routine specified in the IEAVTSEL name list the register's contents are as follows:

Register	Contents
1	address of parameter list
13	address of standard 72 byte save area
14	return address
15	entry address

Return Specifications

Upon return from the user exit routine the registers must contain:

Register	Contents
0-14	same as on entry
15	a return code which is interpreted as follows:

0	exit was successful. SDEPLEXE is turned off before calling the next exit.
nonzero	exit was unsuccessful SDEPLEXE and SDEPLERR will be turned on before calling the next exit.

Programming Considerations

The exit work area (pointed to by SDEPLWA) is a general work area for use by the user exit routine. It is cleared to zeros before each user exit routine is given control.

The exit interface area (pointed to by SDEPLEXT) is intended as a communication area that a user exit routine can use to pass information to successive user exit routines. This area is not zeroed between calls to user exit routines but is zeroed before the call to the first user exit routine. If an installation chooses, it can use this area as a work area.

The copy of the dump header record (pointed to by SDEPLHD) contains all the information that the post dump user exit routines should require. The following data is contained in the header record if the dump has not been suppressed:

PRDDUMPT	dump type - tells whether the dump is an SDUMP, SYSMDUMP, or an SDUMP for a SLIP request.
PRDDSNAM	dump data set name. Either SYS1.DUMPxx, or UNIT=uuu, or the name of the SYSMDUMP data set.
PRDERRID	Error ID from this dump.
PRDSDWA	A copy of the SDWA of the caller of SDUMP. *Note:* This is not present on SLIP dumps. From this SDWA you can obtain the failing module name, the ABEND code, and any other diagnostic data. See mapping macro IHASDWA for fields available.
PRD*****	The header contains other information that might be useful. See the mapping macro AMDDATA in the MVS/XA Data Areas microfiche.

If the dump is suppressed by DAE, the first 2K of the header record and portions of the second 2K contain zeroes. In this event, the DAE section of the header record (located in the second 2K) contains data gathered by DAE about this dump. Note that the DAE section of the header record is present regardless of whether the dump is suppressed or not. The following fields in the DAE section contain diagnostic information:

ADSSDAE	Start of the DAE section of the header record.
DAESSMVS	MVS format symptom string.
DAECRIT	Criteria for symptom string considered unique by DAE.
DAESTAT	DAE status flags mapped by ADYDSTAT.
DAEERID	ERROR-ID from the original occurrence of the dump.
DAEDCNT	The number of occurrences of the dump.

See the mapping macro AMDDATA for a complete list of the header records fields.

Operational Considerations

The user exit routines receive control in addressing mode AMODE (31) or (24) depending on how the user exit routine is link-edited, supervisor state, key 0, and task mode in the DUMPSRV (dumping services) address space. If an installation does not wish to have the user exit routine run in supervisor state or key 0, the user exit routine must issue a MODESET macro to obtain the desired state. The user exit routine must save and restore the caller's registers and return to the caller in the same state as they were in at entry.

Possible Uses

The following are examples of possible applications for which a post dump user exit routine might be written:

- To extract certain information from the header record such as dump title, ERROR-ID, time of dump, ABEND code and failing module name. This information could then be written to a log data set, using DISP=MOD to append the new entries on the end.
- To offload a DASD dump data set to a tape, for later processing, and log the information in a log data set.
- To start a PRDMP job that would print a small portion of the dump (such as log data, summary and/or SUMDUMP).

IEECVXIT

Functional Description - (Alter WTO/WTOR Message Routing and Descriptor Codes)

The IEECVXIT user exit routine provides a way to alter routing and/or descriptor codes for some WTO or WTOR messages. This user exit routine cannot alter the message text. If the user exit routine sets a nonzero routing code to zero, the message will not be displayed at any console, but will be recorded on the written log. If the user exit routine attempts to delete a WTOR message, the WTOR message will be sent to the master console. Entry to this exit is from the module IEAVVWTO.

Note: In addition to the IEECVXIT exit, there are other user exits (IEAVMXIT and exits that you specify on the USEREXIT parameter in an MPFLSTxx member of SYS1.PARMLIB) that you can use to modify message processing. These exits have greater capability than IEECVXIT. For information on these exits, see IEAVMXIT in this book.

Suppressing Unnecessary Messages

To reduce the number of messages the operator must deal with, your installation can suppress certain messages. The message processing facility (MPF) should be used to suppress messages that begin with particular message IDs. The IEECVXIT exit can be used to suppress messages if MPF is not suitable for the function. See *MVS/XA SPL: System Modifications* for more information on the operation of MPF.

Entry Specifications

Upon entry to the IEECVXIT user exit routine, the register contents are as follows:

Register	Contents
0	irrelevant
1	address of a full word pointing to a parameter list
2-12	irrelevant
13	address of standard save area
14	address that user exit returns control to at exit time
15	address of user exit IEECVXIT

Return Specifications

Upon return from the user exit routine, the registers must contain:

Register	Contents
0-15	same as upon entry

Programming Considerations

A WTO/WTOR exit routine can modify the standard routing codes and descriptor codes. This routine becomes part of the control program. When MVS uses a message's routing code field to route the message, the routine receives control before the message is routed. When the routine receives control, register 1 contains a pointer to a word that points to the first word of the message text. The message text field is 128 bytes long, followed by a four-byte routing code field and a four-byte descriptor code field. The exit routine can examine but not modify the message text.

A message is sent only to those locations specified in the modified routing codes. All messages with modified routing codes are sent to the written log when the log is active. When the written log is not active, the exit routine must not suppress messages that contain a routing code of 1, 2, 3, 4, 7, 8, or 10 because messages with these codes are necessary for system maintenance.

Programming Conventions for the WTO/WTOR Exit Routine

The programming conventions for the WTO/WTOR exit routine are:

- The exit routine's name must be IEECVXIT.
- The exit routine can be any size.
- The exit routine can allow interruptions. The routine receives control with no locks held; it must return control with no locks held.
- The exit routine must be reenterable and serially reusable. Macro instructions with expansions that store information into an inline parameter list must not be used.
- Registers must be saved at entry and restored before returning.
- The exit routine can issue WTO macro instructions. Note that the WAIT macro instruction must not be used when the exit routine is entered under the console communications task. (Doing so might permanently terminate console communications.)
- The exit routine is part of the WTO SVC. If the exit routine terminates abnormally, the WTO request is terminated.
- Exit from the routine is done using a branch and return via register 14. An abnormal termination can occur when there is no currently owned region.

- GETMAIN should not be issued for subpools that represent space within a region (0 through 127, 240, or 250 through 252). Because the exit routine executes as a part of the control program, subpools such as 229 or 230 can be used.

- The format of text and codes is:

  - Message text (128 characters padded with blanks).
  - Routing codes (4 bytes).
  - Descriptor codes (4 bytes).

In the routing code field, a bit setting of "1" indicates that the WTO or WTOR was assigned that particular routing code. Bit assignments and their meaning are:

Bit	**Assignment**	**Meaning**
Byte 0		
Bit 0	Routing code 1	Master console action
Bit 1	Routing code 2	Master console information
Bit 2	Routing code 3	Tape pool
Bit 3	Routing code 4	Direct access pool
Bit 4	Routing code 5	Tape library
Bit 5	Routing code 6	Disk library
Bit 6	Routing code 7	Unit record pool
Bit 7	Routing code 8	Teleprocessing control
Byte 1		
Bit 0	Routing code 9	System security
Bit 1	Routing code 10	System error/maintenance
Bit 2	Routing code 11	Programmer information
Bit 3	Routing code 12	Reserved
Bit 4	Routing code 13	Available for customer use
Bit 5	Routing code 14	Available for customer use
Bit 6	Routing code 15	Available for customer use
Bit 7	Routing code 16	Reserved
Byte 2		Reserved
Byte 3		Reserved

In the descriptor code field, a bit setting of "1" indicates that the WTO or WTOR was assigned that particular descriptor code. Bit assignments and their meanings are:

Bit	**Assignment**	**Meaning**
Byte 0		
Bit 0	Descriptor code 1	System failure
Bit 1	Descriptor code 2	Immediate action required
Bit 2	Descriptor code 3	Eventual action required
Bit 3	Descriptor code 4	System status
Bit 4	Descriptor code 5	Immediate command response
Bit 5	Descriptor code 6	Job status
Bit 6	Descriptor code 7	Application program/processor
Bit 7	Descriptor code 8	Out-of-line message
Byte 1		
Bit 0	Descriptor code 9	DISPLAY or TRACK command response
Bit 1	Descriptor code 10	Dynamic status displays
Bit 2	Descriptor code 11	Critical eventual action required
Bits 3-7		Reserved
Byte 2		Reserved
Byte 3		Reserved

Messages That IEECVXIT Will Not Receive

Messages issued by authorized programs that specify MCSFLAGS or MSGTYP on the WTO or WTOR macro will not be passed to the user exit routine. Connecting multiple-line WTOs issued by system programs will not be passed to the user exit routine. Responses to commands will not be passed to the exit. Routing of these messages depends on criteria other than the routing codes; therefore, the system bypasses the exit routine.

Adding a WTO/WTOR Exit Routine to the Control Program

The IBM-supplied module IEECVXIT is used as the WTO/WTOR exit routine unless you supply your own. The IBM-supplied module is a dummy routine that contains only a BR 14 instruction. It does no processing of messages.

To enter your own exit routine into the control program before system generation, use the linkage editor to replace the dummy WTO/WTOR exit routine IEECVXIT in SYS1.AOSC5 with your WTO/WTOR exit routine.

To enter your exit routine into the control program after system generation, use the linkage editor to replace the dummy WTO/WTOR exit routine in load module IEECVXIT in SYS1.LPALIB with your WTO/WTOR exit routine. Placing the module on a page boundary might reduce the number of page faults incurred, thus improving overall system performance.

Operational Considerations

The user exit routine must not enter an MVS WAIT or invoke any service that can issue an MVS WAIT.

IEAVMXIT and User-Specified Exits

Functional Description - (General and User-Specified WTO/WTOR Exit Routines)

The IEAVMXIT user exit (or an exit that you specify on the USEREXIT parameter in an MPFLSTxx member of SYS1.PARMLIB) allows you to modify message processing beyond the capabilities of the IEECVXIT user exit. You can use the IEAVMXIT user exit (or the user-specified WTO/WTOR exit (specified on the USEREXIT parameter) to:

- Change the message text, routing codes, descriptor codes, and console on which the message is displayed.

  Note: If you are using consoles with color capabilities, changing the descriptor code might alter the color in which the message is displayed.

- Queue messages to a particular active console, unconditionally to any console, and by routing codes only.
- Direct messages to hardcopy only, or to the console only, or to both hardcopy and console.
- Delete the message.
- Indicate whether or not the action message retention facility is to retain an action message.
- Indicate whether or not to broadcast a message to active consoles.
- Override message processing facility (MPF) suppression.
- Issue SVCs.
- Reply to WTORs.

For information on the MPFLSTxx member of SYS1.PARMLIB, see *MVS/XA SPL: Initialization and Tuning*. For information on reactivating user exits after they are modified, see *MVS/XA SPL: System Modifications.*

Entry Specifications

Upon entry to the user exit routine, the register contents are as follows:

Register	Contents
0	irrelevant
1	pointer to the address of the communications task exit parameter list (CTXT)
2-12	irrelevant
13	address of standard save area
14	return address
15	entry point address

The parameter list (CTXT), which can be located using the standard linkage from general register 1, contains the following information:

Field Name	Offset	Length	Bit Pattern	Description
CTXT	0 (0)	64		Communications task exit parameter list
CTXTACRN	0 (0)	4		Acronym 'CTXT'
CTXTVRSN	4 (4)	1		Version level
CTXTS212				Level of OS/VS2 JBB2125
CTXTS220				Level of OS/VS2 JBB2220
CTXTVERN				Current version level
CTXTMCS				MCS Console class
CTXTRSV1	5 (5)	3		Reserved
CTXTTXPJ	8 (8)	4		Pointer to text of major line
CTXTTXPN	12 (C)	4		Pointer to text of minor line
CTXTSEQN	16 (10)	4		WTO sequence number
CTXTRSV2	20 (14)	1		Reserved
CTXTMLID	21 (15)	3		Multiple-line WTO ID
CTXTRPID	24 (18)	2		Reply ID
CTXTMTYP	26 (1A)	2		Message type flags
CTXTMTY1	26 (1A)	1		First byte of message type flags
CTXTMTYA			X'80'	Monitor jobnames
CTXTMTYB			X'40'	Monitor status
			X'20'	Reserved
			X'10'	Reserved
			X'08'	Reserved
CTXTMTYF			X'04'	Monitor SESS
			X'02'	Reserved
			X'01'	Reserved
CTXTMTY2	27 (1B)	1		Second byte of message type flags
CTXTRCLN	28 (1C)	2		Length of routing codes in bytes
CTXTDCLN	30 (1E)	2		Length of descriptor codes in bytes
CTXTRCP	32 (20)	4		Pointer to routing codes
CTXTDCP	36 (24)	4		Pointer to descriptor codes
CTXTCIDP	40 (28)	4		Pointer to 1-byte console ID
CTXTSFLG	44 (2C)	4		Status flags (input to user exit)
CTXTSFB1	44 (2C)	1		Status flags byte one
CTXTSQPC			X'80'	Queue to a particular active console
CTXTSQUN			X'40'	Queue to a particular console unconditionally
CTXTSSUP			X'20'	Suppressed by MPF
CTXTSFHC			X'10'	Hardcopy
CTXTSNHC			X'08'	No hardcopy
CTXTSHCO			X'04'	Hardcopy only
CTXTSRSP			X'02'	Command response
CTXTSBCA			X'01'	Broadcast to active consoles
CTXTSFB2	45 (2D)	1		Status flag byte two
CTXTSRET			X'80'	Message to be retained by action message retention facility
CTXTSFB3	46 (2E)	1		Reserved
CTXTSFB4	47 (2F)	1		Reserved
CTXTRFLG	48 (30)	4		Request flags (from the user exit to the system)
CTXTRF3B	48 (30)	3		Request flags - three bytes
CTXTRFB1	48 (30)	1		Request flags byte one

Field Name	Offset	Length	Bit Pattern	Description
CTXTRCMT			X'80'	Change the message text
CTXTRCRC			X'40'	Change the routing codes
CTXTRCDC			X'20'	Change the descriptor codes
CTXTRQPC			X'10'	Queue to a particular active console
CTXTRQUN			X'08'	Queue to a particular console unconditionally
CTXTRQRC			X'04'	Queue by routing codes only
CTXTRCCN			X'02'	Change the 1-byte console ID
CTXTRPML			X'01'	Process minor lines
CTXTRFB2	49 (31)	1		Request flags byte two
CTXTRDTM			X'80'	Delete the message (no written and no display)
CTXTROMS			X'40'	Override MPF suppression
CTXTRFHC			X'20'	Force hardcopy
CTXTRNHC			X'10'	Force no hardcopy
CTXTRHCO			X'08'	Force hardcopy only
CTXTRBCA			X'04'	Broadcast message to all active consoles
CTXTRBCN			X'02'	Do not broadcast message to all active consoles
CTXTRNRT			X'01'	Action message retention facility is not to retain this message
CTXTRFB3	50 (32)	1		Request flags byte three
CTXTRRET			X'80'	Action message retention facility is to retain this message
CTXTRCKY			X'40'	Change the retrieval key
CTXTRCFC			X'20'	Change the 4-byte console id
CTXTRCMF			X'10'	Change the message type flags
	51 (33)	1		Reserved for communications task use
CTXTFCNP	52 (34)	4		Pointer to 4-byte console id
CTXTPREQ	56 (38)	4		Pointer to request flags, reflecting changes done to this file message previously
CTXTTOKN	60 (3C)	4		Token value
CTXTKEY	64 (40)	8		Retrieval key
CTXTJBNM	72 (48)	8		Name of job that issued message
CTXTRSV3	80 (50)	8		Reserved
CTXTSYSN	88 (58)	8		Name of the system on which WTO originated
CTXTRSV4	96 (60)	12		Reserved

The following structure (CTXTATTR) is pointed to by CTXTTXPJ or CTXTTXPN, depending on whether the WQE is a major or a minor line. For a major line, the field CTXTTXPJ contains a nonzero value and the CTCTTXPN contains zero. For a minor line, both fields contain nonzero value. This structure describes the attributes of the message currently being processed. The text of the message is contained within this structure. For a single-line WTO or the first line of a multiple-line WTO, the text field (CTXTTMSG) begins with the first character of the message ID.

Field Name	Offset	Length	Bit Pattern	Description
CTXTATTR	0	132		Communications task exit message attributes
CTXTTLEN	0	2		Text length in bytes
CTXTTLMX	2	2		Maximum length of text in bytes
CTXTTFLG	4	2		Message type flags
CTXTTFB1	4	1		Type flags byte one
CTXTTFSL			X'80'	A single-line message
CTXTTFWR			X'40'	A WTOR
CTXTTFMJ			X'20'	A multiple-line message
CTXTTFMC			X'10'	A control line
CTXTTFML			X'08'	A label line
CTXTTFMD			X'04'	A data line
CTXTTFME			X'02'	An end line
			X'01'	Reserved
CTXTTFB2	5	1		Reserved
CTXTTMSG	6	126		Text of message

The following structure (CTXTROUT), which is pointed to by CTXTRCP, maps the routing codes associated with the message.

Field Name	Offset	Length	Bit Pattern	Description
CTXTROUT	0	16		Routing codes
CTXTR001	0	1		First byte of routing codes
CTXTR01			X'80'	Code 1 – Master console action
CTXTR02			X'40'	Code 2 – Master console information
CTXTR03			X'20'	Code 3 – Tape pool
CTXTR04			X'10'	Code 4 – Direct access pool
CTXTR05			X'08'	Code 5 – Tape library
CTXTR06			X'04'	Code 6 – Disk library
CTXTR07			X'02'	Code 7 – Unit record pool
CTXTR08			X'01'	Code 8 – Teleprocessing control
CTXTR002	1	1		Second byte of routing codes
CTXTR09			X'80'	Code 9 – System security
CTXTR10			X'40'	Code 10 – System/error maintenance
CTXTR11			X'20'	Code 11 – Programmer information
CTXTR12			X'10'	Code 12 – Emulator information
CTXTR13			X'08'	Code 13 – User routing code
CTXTR14			X'04'	Code 14 – User routing code
CTXTR15			X'02'	Code 15 – User routing code
CTXTR16			X'01'	Code 16 – User routing code
CTXTR003	2	1		Third byte of routing codes
CTXTR17			X'80'	Code 17 – User routing code
CTXTR18			X'40'	Code 18 – User routing code
CTXTR19			X'20'	Code 19 – User routing code
CTXTR20			X'10'	Code 20 – User routing code
CTXTR21			X'08'	Code 21 – Reserved for JES usage
CTXTR22			X'04'	Code 22 – Reserved for JES usage
CTXTR23			X'02'	Code 23 – Reserved for JES usage
CTXTR24			X'01'	Code 24 – Reserved for JES usage
CTXTR004	3	1		Fourth byte of routing codes
CTXTR25			X'80'	Code 25 – Reserved for JES usage
CTXTR26			X'40'	Code 26 – Reserved for JES usage
CTXTR27			X'20'	Code 27 – Reserved for JES usage
CTXTR28			X'10'	Code 28 – Reserved for JES usage
CTXTR29			X'08'	Code 29 – Reserved
CTXTR30			X'04'	Code 30 – Reserved
CTXTR31			X'02'	Code 31 – Reserved
CTXTR32			X'01'	Code 32 – Reserved
CTXTR005	4	1		Fifth byte of routing codes
CTXTR33			X'80'	Code 33 – Reserved
CTXTR34			X'40'	Code 34 – Reserved
CTXTR35			X'20'	Code 35 – Reserved
CTXTR36			X'10'	Code 36 – Reserved
CTXTR37			X'08'	Code 37 – Reserved
CTXTR38			X'04'	Code 38 – Reserved
CTXTR39			X'02'	Code 39 – Reserved
CTXTR40			X'01'	Code 40 – Reserved
CTXTR006	5	1		Sixth byte of routing codes
CTXTR41			X'80'	Code 41 – Reserved
CTXTR42			X'40'	Code 42 – General information about JES2 or JES3
CTXTR43			X'20'	Code 43 – Reserved for JES usage
CTXTR44			X'10'	Code 44 – Reserved for JES usage
CTXTR45			X'08'	Code 45 – Reserved for JES usage
CTXTR46			X'04'	Code 46 – Reserved for JES usage
CTXTR47			X'02'	Code 47 – Reserved for JES usage
CTXTR48			X'01'	Code 48 – Reserved for JES usage

Field Name	Offset	Length	Bit Pattern	Description
CTXTR007	6	1		Seventh byte of routing codes
CTXTR49			X'80'	Code 49 – Reserved for JES usage
CTXTR50			X'40'	Code 50 – Reserved for JES usage
CTXTR51			X'20'	Code 51 – Reserved for JES usage
CTXTR52			X'10'	Code 52 – Reserved for JES usage
CTXTR53			X'08'	Code 53 – Reserved for JES usage
CTXTR54			X'04'	Code 54 – Reserved for JES usage
CTXTR55			X'02'	Code 55 – Reserved for JES usage
CTXTR56			X'01'	Code 56 – Reserved for JES usage
CTXTR008	7	1		Eighth byte of routing codes
CTXTR57			X'80'	Code 57 – Reserved for JES usage
CTXTR58			X'40'	Code 58 – Reserved for JES usage
CTXTR59			X'20'	Code 59 – Reserved for JES usage
CTXTR60			X'10'	Code 60 – Reserved for JES usage
CTXTR61			X'08'	Code 61 – Reserved for JES usage
CTXTR62			X'04'	Code 62 – Reserved for JES usage
CTXTR63			X'02'	Code 63 – Reserved for JES usage
CTXTR64			X'01'	Code 64 – Reserved for JES usage
CTXTR009	8	1		Ninth byte of routing codes
CTXTR65			X'80'	Code 65 – Processor related message
CTXTR66			X'40'	Code 66 – Processor related message
CTXTR67			X'20'	Code 67 – Processor related message
CTXTR68			X'10'	Code 68 – Processor related message
CTXTR69			X'08'	Code 69 – Processor related message
CTXTR70			X'04'	Code 70 – Processor related message
CTXTR71			X'02'	Code 71 – Processor related message
CTXTR72			X'01'	Code 72 – Processor related message
CTXTR010	9	1		Tenth byte of routing codes
CTXTR73			X'80'	Code 73 – Processor related message
CTXTR74			X'40'	Code 74 – Processor related message
CTXTR75			X'20'	Code 75 – Processor related message
CTXTR76			X'10'	Code 76 – Processor related message
CTXTR77			X'08'	Code 77 – Processor related message
CTXTR78			X'04'	Code 78 – Processor related message
CTXTR79			X'02'	Code 79 – Processor related message
CTXTR80			X'01'	Code 80 – Processor related message
CTXTR011	10	1		Eleventh byte of routing codes
CTXTR81			X'80'	Code 81 – Processor related message
CTXTR82			X'40'	Code 82 – Processor related message
CTXTR83			X'20'	Code 83 – Processor related message
CTXTR84			X'10'	Code 84 – Processor related message
CTXTR85			X'08'	Code 85 – Processor related message
CTXTR86			X'04'	Code 86 – Processor related message
CTXTR87			X'02'	Code 87 – Processor related message
CTXTR88			X'01'	Code 88 – Processor related message
CTXTR012	11	1		Twelfth byte of routing codes
CTXTR89			X'80'	Code 89 – Processor related message
CTXTR90			X'40'	Code 90 – Processor related message
CTXTR91			X'20'	Code 91 – Processor related message
CTXTR92			X'10'	Code 92 – Processor related message
CTXTR93			X'08'	Code 93 – Processor related message
CTXTR94			X'04'	Code 94 – Processor related message
CTXTR95			X'02'	Code 95 – Processor related message
CTXTR96			X'01'	Code 96 – Processor related message
CTXTR013	12	1		Thirteenth byte of routing codes
CTXTR97			X'80'	Code 97 – Device related message
CTXTR98			X'40'	Code 98 – Device related message
CTXTR99			X'20'	Code 99 – Device related message
CTXTR100			X'10'	Code 100 – Device related message
CTXTR101			X'08'	Code 101 – Device related message
CTXTR102			X'04'	Code 102 – Device related message
CTXTR103			X'02'	Code 103 – Device related message
CTXTR104			X'01'	Code 104 – Device related message

Field Name	Offset	Length	Bit Pattern	Description
CTXTR014	13	1		Fourteenth byte of routing codes
CTXTR105			X'10'	Code 105 – Device related message
CTXTR106			X'08'	Code 106 – Device related message
CTXTR107			X'04'	Code 107 – Device related message
CTXTR108			X'02'	Code 108 – Device related message
CTXTR109			X'01'	Code 109 – Device related message
CTXTR110			X'04'	Code 110 – Device related message
CTXTR111			X'02'	Code 111 – Device related message
CTXTR112			X'01'	Code 112 – Device related message
CTXTR015	14	1		Fifteenth byte of routing codes
CTXTR113			X'10'	Code 113 – Device related message
CTXTR114			X'08'	Code 114 – Device related message
CTXTR115			X'04'	Code 115 – Device related message
CTXTR116			X'02'	Code 116 – Device related message
CTXTR117			X'01'	Code 117 – Device related message
CTXTR118			X'04'	Code 118 – Device related message
CTXTR119			X'02'	Code 119 – Device related message
CTXTR120			X'01'	Code 120 – Device related message
CTXTR016	15	1		Sixteenth byte of routing codes
CTXTR121			X'10'	Code 121 – Device related message
CTXTR122			X'08'	Code 122 – Device related message
CTXTR123			X'04'	Code 123 – Device related message
CTXTR124			X'02'	Code 124 – Device related message
CTXTR125			X'01'	Code 125 – Device related message
CTXTR126			X'04'	Code 126 – Device related message
CTXTR127			X'02'	Code 127 – Device related message
CTXTR128			X'01'	Code 128 – Device related message

The following structure (CTXTDESC), which is pointed to by CTXTDCP, maps the descriptor codes associated with the message.

Field Name	Offset	Length	Bit Pattern	Description
CTXTDESC	0	2		Descriptor codes
CTXTDC1	0	1		First byte of descriptor codes
CTXTDC01			X'80'	Code 1
CTXTDC02			X'40'	Code 2
CTXTDC03			X'20'	Code 3
CTXTDC04			X'10'	Code 4
CTXTDC05			X'08'	Code 5
CTXTDC06			X'04'	Code 6
CTXTDC07			X'02'	Code 7
CTXTDC08			X'01'	Code 8
CTXTDC2	1	1		Second byte of descriptor codes
CTXTDC09			X'80'	Code 9
CTXTDC10			X'40'	Code 10
CTXTDC11			X'20'	Code 11
			X'10'-X'01'	Reserved

The following structure (CTXTCONS), which is pointed to by CTXTCIDP, contains the 1-byte console ID to which the message is being queued.

Field Name	Offset	Length	Description
CTXTCONS	0	9	Console identification
CTXTCIDR	0	1	Reserved
CTXTCNID	1	1	Console ID
	2	7	Reserved

The following structure (CTXTFBCN), which is pointed to by CTXTFCNP, contains the 4-byte console ID to which the message is being queued.

Field Name	Offset	Length	Description
CTXTFBCN	0	4	Console identification
CTXTFBCL	0	1	Console class
CTXTFBRV	1	1	Reserved
CTXTFBNM	2	2	Console number
	2	1	First byte of console number
CTXTFBMI	3	1	MCS portion of console identification

The following structure (CTXTPRFL) reflects changes made to this message.

Field Name	Offset	Length	Bit Pattern	Description
CTXTPRFL	0	4		Request flags (from the user exit to the system)
CTXTPF3B	48 (30)	3		Request flags - three bytes
CTXTPFB1	0	1		Request flags byte one
CTXTPCMT			X'80'	Change the message text
CTXTPCRC			X'40'	Change the routing codes
CTXTPCDC			X'20'	Change the descriptor codes
CTXTPQPC			X'10'	Queue to a particular active console
CTXTPQUN			X'08'	Queue to a particular console unconditionally
CTXTPQRC			X'04'	Queue by routing codes only
CTXTPCCN			X'02'	Change the 1-byte console ID
CTXTPPML			X'01'	Process minor lines
CTXTPFB2	1	1		Request flags byte two
CTXTPDTM			X'80'	Delete the message (no written and no display)
CTXTPOMS			X'40'	Override MPF suppression
CTXTPFHC			X'20'	Force hard copy
CTXTPNHC			X'10'	Force no hard copy
CTXTPHCO			X'08'	Force hard copy only
CTXTPBCA			X'04'	Broadcast message to all active consoles
CTXTPBCN			X'02'	Do not broadcast message to all active consoles
CTXTPNRT			X'01'	Action message retention facility is not to retain this message
CTXTPFB3	2	1		Request flags byte three
CTXTPRET			X'80'	Action message retention facility is to retain this message
CTXTPCKY			X'40'	Change the retrieval key
CTXTPCFC			X'20'	Change the 4-byte console id
CTXTPCMF			X'10'	Change the message type flags
	3	1		Reserved for communications task use

Return Specifications

Upon return from the user exit routine, the registers contain:

Register	Contents
0-15	same as upon entry

Programming Considerations

IEAVMXIT or a user-specified WTO/WTOR exit routine is called for all single-line messages. For a multiple-line message, the user exit routine is called only for the first line of the message, unless the exit requests otherwise. (The default is to bypass minor-line processing.)

An exit routine can convert a single-line message to a multiple-line message. To do this, the exit suppresses the original message and internally issues a multiple-line message with the original text as the major line and additional text as minor lines.

Note that there are two text pointers: the first one is used for a single-line message or for the major line of a multiple-line message; the second one is used for the minor line(s) of a multiple-line message. The first text pointer points to the text of a single-line message or to the text of the major line of a multiple-line message. The second text pointer points to the current minor line of a multiple-line message (while the first text pointer points to the associated major line).

IEAVMXIT is activated during communications task initialization by issuing a LOAD for IEAVMXIT and saving the address in the general WTO/WTOR user exit table (GENX). If a user-specified WTO/WTOR exit routine is desired, its address is located during the processing of the SET MPF command (and the associated processing of the specified MPFLSTxx parmlib member). The appropriate address is passed to the user exit interface, and the user exit routine is invoked via standard linkage.

WTO invokes user-specified WTO/WTOR exit routines for particular messages. During MPF table initialization processing, each user exit routine is loaded and its address saved in the MPF table. User-specified WTO/WTOR exit routines reside in the LNKLST concatenation (SYS1.LINKLIB or a data set concatenated to it).

IEAVMXIT or any user-specified WTO/WTOR exit routine must observe the following programming conventions:

- The exit routine's name is IEAVMXIT or a unique user-specified name.
- The exit routine can be any size.
- The exit routine can allow interruptions. The routine receives control with no locks held; it must return control with no locks held.
- The exit routine must be re-enterable and serially reusable. Macro instructions whose expansions store information into an in-line parameter list must not be used.
- The exit routine receives control in 31-bit mode and must use 31-bit addresses and assemble with AMODE 31 and RMODE ANY.
- Registers must be saved at entry and restored before returning.
- The exit routine can issue macro instructions.

- The WAIT macro instruction or any implied wait must not be used when the exit routine is entered under the console communications task. (Doing so might permanently terminate console communications.)

- Exit from the routine is via the RETURN macro instruction. An abnormal termination can occur when there is no currently owned region.

- GETMAIN should not be issued for subpools that represent space within a region (0 through 127, 240, or 250 through 252). Because the exit routine executes as a part of the control program, subpools such as 229 or 230 can be used.

- The exit routine should provide its own level of recovery because the system does not pass control to the exit if the exit abnormally terminates. See *MVS/XA SPL: System Modifications* for information on how to reactivate the exit if it abnormally terminates.

Operational Considerations

An installation **must carefully** consider the message processing that it wants for a particular message, and what it needs to do to achieve the desired processing. One step might be sufficient to obtain your desired result. However, in other cases, several steps might be necessary to obtain the desired result.

The following are examples of situations that an installation might encounter:

- To request queueing of a message to a particular console and to eliminate queueing by routing codes, the steps are:

  - Request queueing to a particular console
  - Request a change to the console ID
  - Specify the desired console ID
  - Request a change to the routing codes
  - Change the routing codes to all zeros

- To request queueing a message by routing codes only and also change the text of the message, the steps are:

  - Request queueing by routing codes only
  - Change the routing to whatever is desired
  - Request change the message text
  - Specify the new length of the text
  - Supply the new text

Incompatible requests are handled in one of two ways. If IEAVMXIT or a user-specified WTO/WTOR user exit makes conflicting requests, the message is either (1) processed in its original state or (2) processed according to the request that is least detrimental to the message.

The following incompatible requests cause the message to be processed in its original state:

- A request to delete a message and a request that specifies processing other than message deletion
- A request to queue via routing codes only and a request with any of the following:
  - Queue to a particular active console
  - Queue unconditionally to a console
  - Queue to hardcopy only
- A request to queue a message to hardcopy only and a request to broadcast the message to active consoles

The following incompatible requests cause the message to be processed according to the request that is least detrimental to the message:

- A request to send a message to hardcopy and a request to not send a message to hardcopy results in a hardcopy of the message.
- A request to send a message to hardcopy while allowing display at a console and a request to send a message only to hardcopy causes the message to sent to hardcopy as well as displayed at any console to which it might have been queued.
- A request to send a message to only hardcopy and a request not to send a message to hardcopy results in only sending a message to hardcopy.
- A request to queue a message to a particular active console and a request to queue the message unconditionally to a console results in queueing the message to the particular active console.
- A request to broadcast a message and a request to not broadcast a message results in not broadcasting the message.

Restrictions and Limitations

There are several restrictions that must be observed when using IEAVMXIT or any user-specified exit routine.

- Do not use the name IEAVMXIT as the name of a WTO/WTOR exit that you specify in an MPFLSTxx member of SYS1.PARMLIB.
- The user exit must not enter an MVS WAIT or invoke any service that can issue an MVS WAIT.
- The user exit must not issue a WTOR with a WAIT from the console communications task address space.
- The exit routine must reside in an authorized library on the LNKLST concatenation.

- Do not code a user exit that receives control for a message that the exit issues; this causes an endless loop. The exit must be coded so that when it receives control for that message, it does not issue the message again.
- When replying to a WTOR, several restrictions should be noted. They are as follows:
  - If an exit is coded to reply to a WTOR via the MGCR macro, the exit must obtain storage for the MGCR parameter list from virtual storage below 16 megabytes.
  - A user exit should reply to a suppressed WTOR; otherwise the WTOR remains outstanding and will not be displayed unless the operator issues a DISPLAY R command.
  - A user exit cannot request deletion of a WTOR; a request for deletion results in suppression of the WTOR. The operator will not be aware of the WTOR unless the operator issues the DISPLAY R command.
- Do not specify a text length or message that exceeds the maximum length allowed for that type of message. If you do, the system truncates the message.
- When processing minor lines of a multiple-line WTO, the user exit can change only the message text.
- For a multiple-line message, the user exit does not receive control for minor lines unless the exit requests minor-line processing. When such a request is made, the exit receives control for each minor line until the exit indicates that no more minor lines are to be processed.
- Some messages (such as $HASP373) have their text completed when WTO calls the subsystem interface. This call occurs after the user exit completes its processing.

Examples of User-Specified WTO/WTOR Exit Routines

This topic contains examples of user-specified WTO/WTOR exit routines that can be used to modify message processing. Each example describes a condition for which you might choose to code the exit, the statement(s) to be placed in the active MPFLSTxx member of SYS1.PARMLIB, and a coded example of the exit.

The examples of user exits described in this topic are:

- CANCWAIT - used to cancel jobs that are waiting for volumes
- OPERCANC - used to cancel jobs that are waiting for data sets
- KEYTRACK - used to route status messages, for critical jobs, to a particular work station
- JOBTRACK - used to track JES2 jobs that are started during a given period

Note: The structure of any of the following examples can be used when writing the general WTO/WTOR exit, IEAVMXIT. However, be aware that IEAVMXIT must do explicit checking for message IDs.

CANCWAIT User Exit Example

When the system cannot satisfy one or more volume requests for a job, the system issues two messages:

- Message IEF690I informs the operator that the volume(s) are not available.
- Message IEF235D indicates that the job will wait for the volume(s) to become available unless the operator wants to cancel the job by responding 'NO' to this message.

If your installation wants to cancel jobs that would otherwise have to wait for volumes to become available, you can code the CANCWAIT exit shown in this example.

The CANCWAIT exit:

- Changes the message routing code for message IEF690I (from code 2 to code 11, meaning that the message appears in the job's system output message class instead of appearing at the master console)
- Suppresses message IEF235D
- Cancels the job (instead of allowing it to wait for the volume(s) to become available or requiring the operator to cancel the job by replying 'NO' to message IEF235D)
- Issues a user-written message to indicate that the system cancelled the job

Note: For this example, you would place the following statements in the active MPFLSTxx member of SYS1.PARMLIB:

```
IEF690I,SUP(NO),USEREXIT(CANCWAIT)
IEF235D,SUP(NO),USEREXIT(CANCWAIT)
```

Coded Example of the CANCWAIT User Exit

```
 TITLE 'CANCWAIT - SAMPLE COMMUNICATIONS TASK USER EXIT FOR MESSAGES   X
                  IEF690I AND IEF235D'
***********************************************************************
*                                                                     *
* MODULE NAME         =  CANCWAIT                                     *
*                                                                     *
* DESCRIPTIVE NAME    =  SAMPLE COMMUNICATIONS TASK USER EXIT         *
*                        FOR MESSAGES IEF690I AND IEF235D.            *
*                                                                     *
* FUNCTION            =  ROUTES MESSAGE IEF690I VIA ROUTING CODE 11,  *
*                        AND SUPPRESSES AND REPLIES TO MESSAGE        *
*                        IEF235D.                                     *
*                                                                     *
*   OPERATION         =  DETERMINES WHICH MESSAGE IS BEING PROCESSED. *
*                        IF THE MESSAGE IS IEF690I, THIS EXIT ROUTINE *
*                        REQUESTS THE ROUTING CODES BE CHANGED TO     *
*                        ROUTING CODE 11 ONLY.                        *
*                                                                     *
*                        IF THE MESSAGE IS IEF235D, THIS EXIT ROUTINE *
*                        REQUESTS THE WTOR BE SUPPRESSED AND          *
*                        REPLIES TO IT INDICATING THAT THE JOB        *
*                        SHOULD BE CANCELLED.                         *
*                                                                     *
*                        A MESSAGE IS THEN ISSUED INDICATING THAT THE *
*                        JOB IS BEING CANCELLED.                      *
*                                                                     *
* ENTRY POINT         =  CANCWAIT                                     *
*                                                                     *
*   PURPOSE           =  TO PROCESS MESSAGES IEF690I AND IEF235D      *
*                                                                     *
*   LINKAGE           =  BALR                                         *
*                                                                     *
*   INPUT DATA        =  REG1 POINTS TO THE ADDRESS OF THE CTXT       *
*                        REG13 ADDRESS OF STANDARD SAVE AREA          *
*                        REG14 RETURN POINT                           *
*                        REG15 ENTRY POINT                            *
*                                                                     *
*   REGISTERS SAVED   =  REG0 - REG15                                 *
*                                                                     *
*   REGISTER USAGE    =  REG5  - POINTER TO THE ADDRESS OF THE CTXT   *
*                        REG11 - MODULE DATA REGISTER                 *
*                        REG12 - MODULE BASE REGISTER                 *
*                        REG13 - POINTER TO A STANDARD SAVE AREA      *
*                        REG14 - RETURN POINT                         *
*                                                                     *
*   REGISTERS RESTORED = REG0 - REG15                                 *
*                                                                     *
*   CONTROL BLOCKS    =                                               *
*     NAME     MAPPING MACRO   REASON USED                 USAGE      *
*     ----     -------------   -----------                -------     *
*     CTXT      IEZVX100        WTO USER EXIT PARAMETER LIST R,W      *
*     MGCR      IEZMGCR         SVC 34 PARAMETER LIST        C,D      *
*                                                                     *
*   KEY = R-READ, W-WRITE, C-CREATE, D-DELETE                         *
*                                                                     *
*   TABLES            =  NONE                                         *
*                                                                     *
*   MACROS            =  GETMAIN, FREEMAIN, MGCR, WTO                 *
*                                                                     *
***********************************************************************
```

```
CANCWAIT CSECT
CANCWAIT AMODE 31                        31-BIT ADDRESSING MODE
CANCWAIT RMODE ANY                       31-BIT RESIDENCE
***********************************************************************
*                                                                     *
*          REGISTER ASSIGNMENTS                                       *
*                                                                     *
***********************************************************************
REG0     EQU   0
REG1     EQU   1
REG2     EQU   2
REG3     EQU   3
REG4     EQU   4
CTXTPTR  EQU   5
REG11    EQU   11
REG12    EQU   12
REG13    EQU   13
REG14    EQU   14
REG15    EQU   15
SPINPRVT EQU   230
***********************************************************************
*                                                                     *
*         STANDARD ENTRY LINKAGE                                      *
*                                                                     *
***********************************************************************
         STM   REG14,REG12,12(REG13)    SAVE CALLER'S REGISTERS
         BALR  REG12,REG0               ESTABLISH MODULE BASE
         USING *,REG12                  REGISTER
         L     CTXTPTR,0(REG1)          ESTABLISH ADDRESSABILITY
         USING CTXT,CTXTPTR             TO THE CTXT
***********************************************************************
*                                                                     *
*        DYNAMIC STORAGE FOR THIS MODULE IS BEING OBTAINED BELOW      *
*        THE 16-MEG LINE BECAUSE SVC 34 REQUIRES THE MGCR PARAMETER   *
*        LIST TO BE IN 24-BIT ADDRESSABLE STORAGE.                    *
*                                                                     *
***********************************************************************
         GETMAIN  RU,LV=DATAEND,SP=SPINPRVT,LOC=BELOW  OBTAIN DYNAMIC  X
                                                 STORAGE
         LR    REG11,REG1               ADDRESS RETURNED IN REG1
         USING DATAAREA,REG11           ADDRESSABILITY TO DYNAMIC      X
                                        STORAGE
         ST    REG13,SAVEAREA+4         SET BACKWARD PTR
         LA    REG15,SAVEAREA           GET ADDRESS OF SAVE AREA
         ST    REG15,8(REG13)           SET FORWARD PTR
         LR    REG13,REG15              SET REG13 TO POINT TO          X
                                        STANDARD SAVE AREA
***********************************************************************
*                                                                     *
*        DETERMINE WHICH MESSAGE IS TO BE PROCESSED.  IEF690I OR      *
*        IEF235D?                                                     *
*                                                                     *
***********************************************************************
         L     REG2,CTXTTXPJ            ESTABLISH ADDRESSABILITY
         USING CTXTATTR,REG2            TO THE MSG ATTRIBUTES
         LA    REG4,CTXTTMSG            ADDRESS OF TEXT AREA
         USING MSGTEXT,REG4             BASE TEXT MAPPING
         CLC   MSGID,IEF690I            MESSAGE IEF690I?
         BNE   MSG235D                  NO, THEN MESSAGE IEF235D
```

```
***********************************************************************
*                                                                     *
*        WORKING WITH MESSAGE IEF690I THEREFORE REQUEST TO CHANGE     *
*        THE ROUTING CODES.                                           *
*                                                                     *
***********************************************************************
         OI    CTXTRFB1,CTXTRCRC        REQUEST TO CHANGE              X
                                        ROUTING CODES
         L     REG3,CTXTRCP             ESTABLISH ADDRESSABILITY
         USING CTXTROUT,REG3            TO THE ROUTING CODES
         XC    CTXTROUT(2),CTXTROUT     ZERO THE ROUTING CODES
         OI    CTXTR002,CTXTR11         QUEUE TO ROUTING CODE 11
         DROP  REG3                     DROP ADDRESSABILITY TO         X
                                        THE ROUTING CODES
         DROP  REG2,REG4                DROP ADDRESSABILITY TO         X
                                        THE MESSAGE ATTRIBUTES
         B     FINISHED                 EXIT MODULE
***********************************************************************
*                                                                     *
*        WORKING WITH MESSAGE IEF235D THEREFORE REPLY TO THE          *
*        MESSAGE AND ISSUE ANOTHER STATING A JOB IS BEING CANCELLED   *
*                                                                     *
***********************************************************************
MSG235D  EQU   *
         OI    CTXTRFB2,CTXTRHCO        SUPPRESS THE MESSAGE
         BAL   REG14,ISSUMGCR           REPLY TO IEF235D 'R XX,NO'
         BAL   REG14,ISSUWTO            ISSUE THE MESSAGE
***********************************************************************
*                                                                     *
*        STANDARD EXIT LINKAGE                                        *
*                                                                     *
***********************************************************************
FINISHED EQU   *
         L     REG13,4(REG13)           RESTORE REG 13
         FREEMAIN  RU,LV=DATAEND,A=(REG11),SP=SPINPRVT FREE THE STORAGE
         LM    REG14,REG12,12(REG13)    RESTORE CALLER'S               X
                                        REGISTERS
         BR    REG14                    RETURN TO CALLER
***********************************************************************
*                                                                     *
*   PROCEDURE - ISSUMGCR                                              *
*                                                                     *
*   FUNCTION  - REPLIES TO MESSAGE IEF235D WITH A NEGATIVE RESPONSE   *
*                                                                     *
*   INPUT     - NONE                                                  *
*                                                                     *
*   OUTPUT    - A REPLY COMMAND IS ISSUED VIA SVC 34                  *
*                                                                     *
*   NOTES     - THE MGCR MACRO CAN BE USED TO ISSUE A REPLY COMMAND   *
*                                                                     *
***********************************************************************
ISSUMGCR EQU   *
         XC    MGCRPL(MGCRLTH),MGCRPL   CLEAR THE PARAMETER LIST
         MVC   MGCRTEXT(L'TXTINSRT),TXTINSRT   MOVE IN THE REPLY BUFFER
         MVC   REPLY,CTXTRPID           INSERT THE REPLY ID
         LA    REG1,(MGCRTEXT-MGCRPL)+L'TXTINSRT      GET MGCRPL LENGTH
         STC   REG1,MGCRLGTH            SAVE LENGTH IN THE MGCRPL
         SR    REG0,REG0                CLEAR REGISTER ZERO
         MGCR  MGCRPL                   ISSUE THE COMMAND
         BR    REG14                    RETURN TO CALLER
```

```
***********************************************************************
*                                                                     *
*   PROCEDURE - ISSUWTO                                               *
*   FUNCTION  - ISSUES A MESSAGE INFORMING OPERATOR THAT A JOB IS     *
*               BEING CANCELLED                                       *
*   INPUT     - NONE                                                  *
*   OUTPUT    - MESSAGE STATING JOB CANCELLED                         *
*                                                                     *
***********************************************************************
ISSUWTO  EQU   *
         L     REG2,CTXTTXPJ            ESTABLISH ADDRESSABILITY
         USING CTXTATTR,REG2            TO MESSAGE ATTRIBUTES
         LA    REG4,CTXTTMSG            ADDRESS OF TEXT AREA
         USING MSGTEXT,REG4             BASE TEXT MAPPING
         MVC   USERAUTO,USERSTAT        MOVE WTO LIST FORM FROM        X
                                        STATIC TO DYNAMIC
         MVC   USERAUTO+24(8),JOBDATA   PLACE JOB DATA INTO MSG
         DROP  REG2,REG4                DROP ADDRESSABILITY TO         X
                                        THE MESSAGE ATTRIBUTES
         WTO   MF=(E,USERAUTO)          ISSUE THE MESSAGE
         BR    REG14                    RETURN TO CALLER
***********************************************************************
*                                                                     *
*        CHARACTER CONSTANTS                                          *
*                                                                     *
***********************************************************************
IEF690I  DC    CL8'IEF690I '            MESSAGE IEF690I
TXTINSRT DC    CL11'REPLY XX,NO'        WTOR REPLY
***********************************************************************
*                                                                     *
*        LIST FORM OF WTO MACRO(STATIC)                               *
*                                                                     *
***********************************************************************
USERSTAT WTO 'USER001I CANCELLING ???????? - WAITING FOR VOLUMES',     X
               ROUTCDE=(11),DESC=(6),MF=L
CNCLMSG  EQU   *-USERSTAT
***********************************************************************
*                                                                     *
*        STORAGE DEFINITIONS                                          *
*                                                                     *
***********************************************************************
DATAAREA DSECT
         DS    0F
SAVEAREA DS    18F                      STANDARD SAVE AREA
         DS    0F
USERAUTO DS    CL(CNCLMSG)              DYNAMIC FORM OF WTO
MGCR     IEZMGCR  DSECT=NO
         ORG   MGCRTEXT
COMMAND  DS    CL6                      THE REPLY COMMAND
REPLY    DS    CL2                      REPLY ID
REPLYMSG DS    CL3                      WTOR RESPONSE
         ORG
DATAEND  EQU   *-DATAAREA
***********************************************************************
*                                                                     *
*         MAPPING OF THE MESSAGE TEXT                                 *
*                                                                     *
***********************************************************************
MSGTEXT  DSECT
MSGID    DS    CL8                      MESSAGE ID
JOBDATA  DS    CL8                      JOB DATA(ID AND NAME)
         ORG   MSGTEXT
         DS    CL4
WTORID   DS    CL8
         IEZVX100
         END   CANCWAIT
```

OPERCANC User Exit Example

When the system cannot satisfy an application program's request for one or more data sets, the system issues message IEF099I to inform the operator that the job will wait until the data sets become available, and then processing for the job will continue. (Although the message does not require a response, the operator can issue the CANCEL command to cancel the job.)

If your installation does not want jobs to wait for data sets to become available, you can use the OPERCANC exit shown in this example.

The OPERCANC exit:

- Changes the descriptor code for message IEF099I (from code 6 to code 2, meaning that the message requires the operator to perform an action instead of just informing the operator of job status)

- Changes the message text to indicate that the operator is to issue the CANCEL command to cancel the job

Notes:

1. *The OPERCANC uses a job-naming convention in which the first three characters of the job name indicate the program type, where 'TTT' specifies an application program. If you use this exit, you can modify its code to meet your installation's job-naming convention(s).*

2. *For this example, you would place the following statement in the active MPFLSTxx member of SYS1.PARMLIB:*

   ```
   IEF099I,SUP(NO),USEREXIT(OPERCANC)
   ```

Coded Example of the OPERCANC User Exit

```
 TITLE 'OPERCANC - SAMPLE COMMUNICATIONS TASK USER EXIT FOR MESSAGE    X
                  IEF099I'
***********************************************************************
*                                                                     *
* MODULE NAME          =  OPERCANC                                    *
*                                                                     *
* DESCRIPTIVE NAME     =  SAMPLE COMMUNICATIONS TASK USER EXIT        *
*                         FOR MESSAGE IEF099I.                        *
*                                                                     *
* FUNCTION             =  ISSUES A MESSAGE TO THE OPERATOR REQUESTING *
*                         THAT APPLICATION JOBS BE CANCELLED          *
*                         RATHER THAN WAIT FOR THE DATASETS THEY      *
*                         NEED.                                       *
*                                                                     *
*   OPERATION          =  JOBNAMES FOR APPLICATION TEST WORK BEGIN    *
*                         WITH THE PREFIX 'TTT'.  IF THE FIRST THREE  *
*                         CHARACTERS OF THE JOBNAME BEGIN WITH        *
*                         THIS PREFIX, THE TEXT OF MESSAGE IEF099I    *
*                          'IEF099I JOB JJJ WAITING FOR DATASETS'     *
*                         WILL BE CHANGED TO SAY:                     *
*                          'IEF099I JOB JJJ WAITING - ISSUE CANCEL'   *
*                         THE MESSAGE WILL BE QUEUED VIA DESCRIPTOR   *
*                         CODE 2. THE DESCRIPTOR CODE WAS PREVIOUSLY  *
*                         6.                                          *
*                                                                     *
* ENTRY POINT          =  OPERCANC                                    *
*                                                                     *
*   PURPOSE            =  TO DETERMINE IF A JOB IS APPLICATION TEST   *
*                         WORK, AND IF IT IS, CHANGE THE TEXT OF      *
*                         MESSAGE IEF099I TO INFORM THE OPERATOR TO   *
*                         CANCEL THE JOB.                             *
*                                                                     *
*   LINKAGE            =  BALR                                        *
*                                                                     *
*   INPUT DATA         =  REG1  POINTS TO THE ADDRESS OF THE CTXT     *
*                         REG13 ADDRESS OF A STANDARD SAVE AREA       *
*                         REG15 ENTRY POINT                           *
*                                                                     *
*                                                                     *
*   REGISTERS SAVED    =  REG0 - REG15                                *
*                                                                     *
*   REGISTER USAGE     =  REG5  - POINTER TO THE ADDRESS OF THE CTXT  *
*                         REG12 - MODULE BASE REGISTER                *
*                         REG13 - POINTER TO A STANDARD SAVE AREA     *
*                         REG14 - RETURN POINT                        *
*                                                                     *
*   REGISTERS RESTORED =  REG0 - REG15                                *
*                                                                     *
*   CONTROL BLOCKS     =                                              *
*     NAME     MAPPING MACRO   REASON USED                  USAGE     *
*     ----     -------------   -----------                 -------    *
*                                                                     *
*     CTXT       IEZVX100       WTO USER EXIT PARAMETER LIST   R,W    *
*                                                                     *
*   KEY = R-READ, W-WRITE, C-CREATE, D-DELETE                         *
*                                                                     *
*   TABLES             =  NONE                                        *
*                                                                     *
*   MACROS             =  NONE                                        *
*                                                                     *
***********************************************************************
```

```
OPERCANC CSECT
OPERCANC AMODE 31                     31-BIT ADDRESSING MODE
OPERCANC RMODE ANY                    31-BIT RESIDENCE
***********************************************************************
*                                                                     *
*          REGISTER ASSIGNMENTS                                       *
*                                                                     *
***********************************************************************
REG0     EQU   0
REG1     EQU   1
REG2     EQU   2
REG3     EQU   3
REG4     EQU   4
CTXTPTR  EQU   5
REG12    EQU   12
REG13    EQU   13
REG14    EQU   14
MSGLNTH  EQU   L'MESSAGE
***********************************************************************
*                                                                     *
*         STANDARD ENTRY LINKAGE                                      *
*                                                                     *
***********************************************************************
         STM   REG14,REG12,12(REG13)    SAVE CALLER'S REGISTERS
         BALR  REG12,REG0               ESTABLISH MODULE BASE
         USING *,REG12                  REGISTER
         L     CTXTPTR,0(REG1)          ESTABLISH ADDRESSABILITY
         USING CTXT,CTXTPTR             TO THE CTXT
***********************************************************************
*                                                                     *
*         DETERMINE IF THE JOB BEING PROCESSED IS APPLICATION TEST    *
*         WORK.  IF IT IS CHANGE THE TEXT AND THE DESCRIPTOR CODE.    *
*                                                                     *
***********************************************************************
         L     REG2,CTXTTXPJ            ESTABLISH ADDRESSABILITY
         USING CTXTATTR,REG2            TO THE MSG ATTRIBUTES
         LA    REG4,CTXTTMSG            GET ADDRESS OF TEXT AREA
         USING MSGTEXT,REG4             BASE TEXT MAPPING
         CLC   JOBPREFX,JOBCLASS        APPLICATION TEST WORK?
         BNE   FINISHED                 NO, THEN EXIT LINKAGE
         L     REG3,CTXTDCP             ESTABLISH ADDRESSABILITY
         USING CTXTDESC,REG3            TO THE DESCRIPTOR CODES
         OI    CTXTRFB1,CTXTRCMT        REQUEST TO CHANGE THE         X
                                        MESSAGE TEXT
         MVC   NEWTEXT,TEXTCHNG         MOVE IN THE NEW TEXT
         MVI   CTXTTLEN+1,MSGLNTH       INSERT LENGTH OF MESSAGE
         OI    CTXTRFB1,CTXTRCDC        REQUEST TO CHANGE THE         X
                                        DESCRIPTOR CODE
         XC    CTXTDESC,CTXTDESC        CLEAR OUT THE DESCRIPTOR CODES
         OI    CTXTDC1,CTXTDC02         CHANGE TO DESCRIPTOR CODE 2
         DROP  REG2,REG3,REG4           DROP ALL USINGS
***********************************************************************
*                                                                     *
*         STANDARD EXIT LINKAGE                                       *
*                                                                     *
***********************************************************************
FINISHED EQU   *
         LM    REG14,REG12,12(REG13)    RESTORE CALLER'S              X
                                        REGISTERS
         BR    REG14                    RETURN TO CALLER
JOBCLASS DC    CL3'TTT'                 JOB CLASSIFICATION
TEXTCHNG DC    CL14'- ISSUE CANCEL'     TEXT TO BE INSERTED IN MESSAGE
```

```
**********************************************************************
*                                                                    *
*          MAPPING OF THE MESSAGE TEXT                               *
*                                                                    *
**********************************************************************
MSGTEXT  DSECT
MESSAGE  DS    0CL43                   LENGTH OF ENTIRE MESSAGE
MSGID    DS    CL8                     MESSAGE ID
         DS    CL4
JOBPREFX DS    CL3                     JOB CLASSIFICATION
         DS    CL14
NEWTEXT  DS    CL14                    TEXT AREA TO BE REPLACED
         IEZVX100
         END   OPERCANC
```

KEYTRACK User Exit Example

There are times when an installation might want to closely monitor certain jobs that it considers "key" or critical jobs. To help in monitoring such jobs, an installation can direct where status messages for the jobs will appear. Thus, the person(s) responsible for monitoring the jobs receive the messages at a particular work station (the specified active console).

If your installation wants status messages that are issued for certain jobs it considers critical to appear at a particular work station, you can use the KEYTRACK exit shown in this example.

Notes:

1. *The KEYTRACK exit uses a job-naming convention in which critical jobs have a '$' as the first character of the job name. If you use this exit, you can modify its code to meet your installation's job-naming convention(s).*

   Similarly, you can modify this exit's code to direct messages for critical jobs to the console of your choice rather than console ID 4, which is the console specified in this example.

2. *For this example, you would place the following statements in the active MPFLSTxx member of SYS1.PARMLIB:*

```
IEF402I,SUP(NO),USEREXIT(KEYTRACK)
IEF403I,SUP(NO),USEREXIT(KEYTRACK)
IEF404I,SUP(NO),USEREXIT(KEYTRACK)
IEF450I,SUP(NO),USEREXIT(KEYTRACK)
IEF451I,SUP(NO),USEREXIT(KEYTRACK)
IEF452I,SUP(NO),USEREXIT(KEYTRACK)
IEF453I,SUP(NO),USEREXIT(KEYTRACK)
```

Coded Example of the KEYTRACK User Exit

```
 TITLE 'KEYTRACK - SAMPLE COMMUNICATIONS TASK USER EXIT'
***********************************************************************
*                                                                     *
* MODULE NAME          =  KEYTRACK                                    *
*                                                                     *
* DESCRIPTIVE NAME     =  SAMPLE COMMUNICATIONS TASK USER EXIT        *
*                                                                     *
* FUNCTION             =  ROUTES KEY CRITICAL JOBS TO A SPECIFIC      *
*                         DISPLAY STATION.                            *
*                                                                     *
*   OPERATION          =  THE MESSAGE TEXT IS SCANNED FOR THE         *
*                         JOBNAME PREFIX CHARACTER ('$' IN THIS       *
*                         EXAMPLE) TO DETERMINE WHAT THE              *
*                         DESTINATION OF THE MESSAGE SHOULD BE.       *
*                         IF THE JOBNAME PREFIX IS FOUND THE MESSAGE  *
*                         IS QUEUED TO A SPECIFIC MONITORING WORK     *
*                         STATION.  IN THIS PARTICULAR CASE, THE      *
*                         WORK STATION IS CONSOLE ID 4.               *
*                                                                     *
* ENTRY POINT          =  KEYTRACK                                    *
*                                                                     *
*   PURPOSE            =  TO SCAN THE FOLLOWING MESSAGE ID'S FOR      *
*                         A JOB PREFIX KEY:                           *
*                            - IEF402I, IEF403I, IEF404I, IEF450I,    *
*                            - IEF451I, IEF452I, IEF453I              *
*                                                                     *
*   LINKAGE            =  BALR                                        *
*                                                                     *
*   INPUT DATA         =  REG1  POINTER TO THE ADDRESS OF THE CTXT    *
*                         REG13 ADDRESS OF STANDARD SAVE AREA         *
*                         REG15 ENTRY POINT                           *
*                                                                     *
*   REGISTERS SAVED    =  REG0 - REG15                                *
*                                                                     *
*   REGISTER USAGE     =  REG5  - POINTS TO THE CTXT                  *
*                         REG12 - MODULE BASE REGISTER                *
*                         REG13 - POINTER TO A STANDARD SAVE AREA     *
*                         REG14 - RETURN POINT                        *
*                                                                     *
*   REGISTERS RESTORED =  REG0 - REG15                                *
*                                                                     *
*   CONTROL BLOCKS     =                                              *
*     NAME     MAPPING MACRO   REASON USED                  USAGE     *
*     ----     -------------   -----------                 -------    *
*     CTXT      IEZVX100       WTO USER EXIT PARAMETER LIST  R,W      *
*                                                                     *
*   KEY = R-READ, W-WRITE, C-CREATE, D-DELETE                         *
*                                                                     *
*   TABLES             =  NONE                                        *
*                                                                     *
*   MACROS             =  NONE                                        *
*                                                                     *
***********************************************************************
```

```
KEYTRACK CSECT
KEYTRACK AMODE 31                    31-BIT ADDRESSING MODE
KEYTRACK RMODE ANY                   31-BIT RESIDENCE
***********************************************************************
*                                                                     *
*          REGISTER ASSIGNMENTS                                       *
*                                                                     *
***********************************************************************
REG0     EQU   0
REG1     EQU   1
REG2     EQU   2
REG3     EQU   3
REG4     EQU   4
CTXTPTR  EQU   5
REG12    EQU   12
REG13    EQU   13
REG14    EQU   14
***********************************************************************
*                                                                     *
*        CHARACTER EQUATES                                            *
*                                                                     *
***********************************************************************
JOBKEY   EQU   C'$'                   JOB NAME KEY
***********************************************************************
*                                                                     *
*        NUMERIC EQUATES                                              *
*                                                                     *
***********************************************************************
CNID04   EQU   X'04'                  CONSOLE ID TO RECEIVE MESSAGE
***********************************************************************
*                                                                     *
*        STANDARD ENTRY LINKAGE                                       *
*                                                                     *
***********************************************************************
         STM   REG14,REG12,12(REG13)    SAVE CALLER'S REGISTERS
         BALR  REG12,REG0             ESTABLISH MODULE BASE
         USING *,REG12                REGISTER
         L     CTXTPTR,0(REG1)        ESTABLISH ADDRESSABILITY
         USING CTXT,CTXTPTR           TO THE CTXT
***********************************************************************
*                                                                     *
*        DETERMINE IF THE MESSAGE PREFIX KEY IS SPECIFIED AS          *
*        PART OF THE JOBNAME.                                         *
*                                                                     *
***********************************************************************
         L     REG2,CTXTTXPJ          ESTABLISH ADDRESSABILITY
         USING CTXTATTR,REG2          TO THE MSG ATTRIBUTES
         LA    REG4,CTXTTMSG          ADDRESS OF TEXT AREA
         USING MSGTEXT,REG4           BASE TEXT MAPPING
***********************************************************************
*                                                                     *
*        IS THIS A CRITICAL JOB BEING PROCESSED?                      *
*                                                                     *
***********************************************************************
         CLI   STARTKEY,JOBKEY        JOB KEY PREFIX MATCH?
         BNE   FINISHED               NO, THEN EXIT LINKAGE
         OI    CTXTRFB1,CTXTRQPC+CTXTRCCN REQUEST TO QUEUE TO A        X
                                        PARTICULAR CONSOLE AND         X
                                        REQUEST TO CHANGE CONSOLE ID
         L     REG3,CTXTCIDP          ESTABLISH ADDRESSABILITY
         USING CTXTCONS,REG3          TO THE CONSOLE ID
         MVI   CTXTCNID,CNID04        INSERT CONSOLE ID
         DROP  REG2,REG3,REG4         DROP ALL USINGS
```

```
***********************************************************************
*                                                                     *
*        STANDARD EXIT LINKAGE                                        *
*                                                                     *
***********************************************************************
FINISHED EQU   *
         LM    REG14,REG12,12(REG13)     RESTORE CALLER'S              X
                                         REGISTERS
         BR    REG14                     RETURN TO CALLER
***********************************************************************
*                                                                     *
*         MAPPING OF THE MESSAGE TEXT                                 *
*                                                                     *
***********************************************************************
MSGTEXT  DSECT
MSGID    DS    CL8                       MESSAGE ID
STARTKEY DS    CL1                       JOB PREFIX KEY
         IEZVX100
         END   KEYTRACK
```

JOBTRACK User Exit Example

If your installation's job entry subsystem is JES2 and you want to know what jobs were started by JES2, you can code the JOBTRACK exit shown in this example. Such an exit makes it possible to obtain job information from the JES2 control blocks.

When JES2 issues message $HASP373 to indicate that a job was started, the JOBTRACK exit obtains the job name and job identifier. The exit creates an entry (for the job) in a table that the installation can use to determine what jobs JES2 started during a given period. This wrap-around table, which resides in the extended common storage area (ECSA), can contain a maximum of 1000 entries.

Note: For this example, you would place the following statement in the active MPFLSTxx member of SYS1.PARMLIB:

```
$HASP373,SUP(NO),USEREXIT(JOBTRACK)
```

Coded Example of the JOBTRACK User Exit

```
 TITLE 'JOBTRACK - SAMPLE COMMUNICATIONS TASK USER EXIT                X
                   FOR $HASP373'
***********************************************************************
* MODULE NAME          =  JOBTRACK                                    *
*                                                                     *
* DESCRIPTIVE NAME     =  SAMPLE COMMUNICATIONS TASK USER EXIT        *
*                         FOR MESSAGE $HASP373.                       *
*                                                                     *
* FUNCTION             =  OBTAINS THE JOB NAME AND JOB ID FROM THE    *
*                         SUBSYSTEM JOBS BLOCK (THIS DATA HAS NOT BEEN *
*                         INSERTED INTO MESSAGE $HASP373 YET)         *
*                         AND INSERTS THEM INTO THE JOBSRUN           *
*                         TRACKING TABLE.                             *
*                                                                     *
*   OPERATION          =  IF THE JOBSRUN TABLE DOES NOT ALREADY EXIST *
*                         OBTAIN STORAGE FOR IT.  BUILD THE TABLE IF  *
*                         NECESSARY, WHICH ENTAILS INSERTING THE      *
*                         JOB NAME AND JOB ID IN EACH ENTRY.          *
*                         THE TABLE WILL BE POINTED TO OUT OF THE CVT. *
*                         THE ANCHOR POINTER IS CVTUSER.              *
*                                                                     *
* ENTRY POINT          =  JOBTRACK                                    *
*                                                                     *
*   PURPOSE            =  TO BUILD A TABLE OF ALL JOBS WHICH WERE     *
*                         STARTED IN THE SYSTEM.  THE TABLE WILL      *
*                         CONTAIN JOB NAMES AND JOB ID'S AND WILL     *
*                         RESIDE IN THE EXTENDED COMMON STORAGE AREA. *
*                         AREA (ECSA).                                *
*                         THE MAXIMUM NUMBER OF ENTRIES IS ONE        *
*                         THOUSAND.  THE TABLE WILL WRAP AROUND WHEN  *
*                         THAT MAXIMUM IS REACHED.                    *
*                                                                     *
*   LINKAGE            =  BALR                                        *
*                                                                     *
*   INPUT DATA         =  REG1  POINTS TO THE ADDRESS OF THE CTXT     *
*                         REG13 ADDRESS OF STANDARD SAVE AREA         *
*                         REG15 ENTRY POINT                           *
*                                                                     *
*   REGISTERS SAVED    =  REG0 - REG15                                *
*                                                                     *
*   REGISTER USAGE     =  REG5  - POINTS TO THE CTXT                  *
*                         REG12 - MODULE BASE REGISTER                *
*                         REG13 - POINTER TO A STANDARD SAVE AREA     *
*                         REG14 - RETURN POINT                        *
*                                                                     *
*   REGISTERS RESTORED =  REG0 - REG15                                *
*                                                                     *
*   CONTROL BLOCKS     =                                              *
*     NAME     MAPPING MACRO   REASON USED                  USAGE     *
*     ----     -------------   -----------                  -------   *
*     CTXT       IEZVX100        WTO USER EXIT PARAMETER LIST R,W     *
*     CVT        CVT             GET PTR TO TCB                R      *
*     JSCB       IEZJSCB         GET PTR TO ACTIVE JSCB        R      *
*     PSA        IHAPSA          GET PTR TO ACTIVE TCB         R      *
*     SJB        $SJB            PICK UP THE JOB ID AND NAME   R      *
*     SSIB       IEFJSSIB        GET PTR TO SJB                R      *
*     TCB        IKJTCB          GET PTR TO JSCB               R      *
*                                                                     *
*   KEY = R-READ, W-WRITE, C-CREATE, D-DELETE                         *
*                                                                     *
*   TABLES             =  JOBSRUN JOB TRACKING TABLE                  *
*                                                                     *
*   MACROS             =  GETMAIN,FREEMAIN                            *
***********************************************************************
```

```
JOBTRACK CSECT
JOBTRACK AMODE 31                          31-BIT ADDRESSING MODE
JOBTRACK RMODE ANY                         31-BIT RESIDENCE
***********************************************************************
*                                                                     *
*         REGISTER ASSIGNMENTS                                        *
*                                                                     *
***********************************************************************
REG0     EQU   0
REG1     EQU   1                           ADDRESS OF THE JOBSRUN TABLE
REG2     EQU   2
REG3     EQU   3
REG4     EQU   4
CTXTPTR  EQU   5                           POINTS TO THE CTXT
REG6     EQU   6
REG8     EQU   8
REG9     EQU   9
REG10    EQU   10
REG12    EQU   12                          MODULE BASE REGISTER
REG13    EQU   13
REG14    EQU   14
***********************************************************************
*                                                                     *
*        NUMERICAL EQUATES                                            *
*                                                                     *
***********************************************************************
ONE      EQU   1                            GENERAL PURPOSE VARIABLE ONE
ONEK     EQU   1000                         MAXIMUM NUMBER OF ENTRIES
SPINECSA EQU   241                          SUBPOOL WHERE JOBSRUN TABLE    X
                                            WILL RESIDE
***********************************************************************
*                                                                     *
*        STANDARD ENTRY LINKAGE                                       *
*                                                                     *
***********************************************************************
         STM   REG14,REG12,12(REG13)        SAVE CALLER'S REGISTERS
         BALR  REG12,REG0                   ESTABLISH MODULE BASE
         USING *,REG12                      REGISTER
         L     CTXTPTR,0(,REG1)             ESTABLISH ADDRESSABILITY
         USING CTXT,CTXTPTR                 TO THE CTXT
***********************************************************************
*                                                                     *
*        ESTABLISH ADDRESSABILITY TO THE NECESSARY CONTROL BLOCKS     *
*        FOR RETRIEVAL OF THE JOB NAME AND ITS ID.                    *
*                                                                     *
***********************************************************************
         LA    REG6,0                       ESTABLISH ADDRESSABILITY
         USING PSA,REG6                     TO THE PSA
         L     REG6,PSATOLD                 GET THE ADDRESS OF THE         X
                                            ACTIVE TCB
         USING TCB,REG6                     ADDRESSABILITY TO THE TCB
         L     REG6,TCBJSCB                 GET THE ADDRESS OF THE JSCB
         USING IEZJSCB,REG6                 GET THE ADDRESS OF
         L     REG6,JSCBACT                 THE ACTIVE JSCB SO THAT
         L     REG6,JSCBSSIB                ADDRESSABILITY TO THE SSIB CAN X
                                            BE ESTABLISHED
         USING SSIB,REG6                    USE THE SSIB TO GET TO THE SJB
         L     REG6,SSIBSUSE                GET THE ADDRESS OF THE SJB
         USING SJBDSECT,REG6                START USING THE SJB
```

```
***********************************************************************
*                                                                     *
*        CALL A ROUTINE TO BUILD THE JOB TRACKING TABLE               *
*                                                                     *
***********************************************************************
         BAL   REG14,BLDTABLE           GO BUILD OR UPDATE THE TABLE
***********************************************************************
*                                                                     *
*        STANDARD EXIT LINKAGE                                        *
*                                                                     *
***********************************************************************
         LM    REG14,REG12,12(REG13)     RESTORE CALLER'S                X
                                         REGISTERS
         BR    REG14                     RETURN TO CALLER
***********************************************************************
*                                                                     *
*    PROCEDURE - BLDTABLE                                             *
*                                                                     *
*    FUNCTION  - BUILDS A TABLE OF JOB ID'S AND JOB NAMES             *
*                                                                     *
*    INPUT     - CVTUSER FIELD OF THE CVT                             *
*                                                                     *
*    OUTPUT    - JOB TRACKING TABLE                                   *
*                                                                     *
***********************************************************************
BLDTABLE EQU   *
         L     REG8,16                  ESTABLISH ADDRESSABILITY
         USING CVTMAP,REG8              TO THE CVT
         L     REG1,CVTUSER             GET ADDRESS OF THE TABLE FROM    X
                                        THE CVT
         LTR   REG1,REG1                DOES THE TABLE EXIST?
         BNZ   TABLEBLT                 YES, THEN UPDATE THE TABLE
***********************************************************************
*                                                                     *
*   TABLE HAS NOT BEEN BUILT YET, THEREFORE GET STORAGE FOR IT.       *
*                                                                     *
***********************************************************************
         L     REG0,STORGAMT
         GETMAIN RU,LV=(0),SP=SPINECSA,LOC=ANY
         ST    REG1,CVTUSER             PUT ADDRESS OF TABLE             X
                                        INTO THE CVT
         DROP  REG8                     DROP ADDRESSABILITY TO THE CVT
***********************************************************************
*                                                                     *
*   CLEAR THE STORAGE OBTAINED BY THE GETMAIN                         *
*                                                                     *
***********************************************************************
         LR    REG2,REG1                GET THE ADDRESS OF THE TABLE     X
                                        IN STORAGE
         L     REG3,STORGAMT            GET THE LENGTH TO BE CLEARED
         LR    REG8,REG1                GET ADDRESS OF STORAGE
         SR    REG9,REG9                CLEAR THE PADDING BYTE
         MVCL  REG2,REG8                CLEAR THE STORAGE
         USING TRAKTABL,REG1            GET ADDRESSABILITY TO THE        X
                                        JOBSRUN TRACKING TABLE
         MVC   TRKACRO,JOBSRUN          INSERT EBCDIC TABLE NAME
         LA    REG8,ENTRYLTH            GET THE SIZE OF AN ENTRY
         ST    REG8,TRKLNGTH            AND STORE IT IN THE TABLE HEADER
         LA    REG8,ONE                 SET THE ENTRY COUNT TO ONE
         ST    REG8,TRKNUM              AND STORE IT IN THE HEADER
         LA    REG4,TRKSTART            GET ADDRESS OF FIRST ENTRY
         ST    REG4,TRKENTRY            AND STORE IT INTO THE HEADER
         ST    REG4,TRKCURR             CURRENT ENTRY AT THIS TIME
         LA    REG8,ONEK                STORE THE MAXIMUM ENTRIES
         ST    REG8,TRKMXENT            ALLOWED IN THE TABLE HEADER
         B     FILENTRY                 FILL IN A TABLE ENTRY
```

```
***********************************************************************
*                                                                     *
*   TABLE HAS BEEN BUILT ALREADY THEREFORE JUST UPDATE IT.           *
*                                                                     *
***********************************************************************
TABLEBLT EQU   *
         L     REG3,TRKNUM              GET CURRENT NUMBER OF ENTRIES
         C     REG3,TRKMXENT            EQUAL TO THE MAXIMUM?
         BE    WRAPTABL                 YES, THEN WRAP AROUND
         LA    REG3,ONE(REG3)           INCREMENT IT BY ONE
         ST    REG3,TRKNUM              STORE THE NEW VALUE
         L     REG4,TRKCURR             GET ADDRESS OF CURRENT ENTRY
         LA    REG4,ENTRYLTH(REG4)      BUMP BY LENGTH OF AN ENTRY
         ST    REG4,TRKCURR             POINT TO THE NEXT SLOT
         B     FILENTRY                 FILL IN A TABLE ENTRY
***********************************************************************
*                                                                     *
*   THE TABLE IS FULL THEREFORE WRAP AROUND                           *
*                                                                     *
***********************************************************************
WRAPTABL EQU   *
         LA    REG4,ONE                 RESET THE ENTRY COUNT TO ONE
         ST    REG4,TRKNUM              AND STORE IT IN THE HEADER
         L     REG4,TRKENTRY            GET ADDRESS OF FIRST ENTRY
         ST    REG4,TRKCURR             MAKE IT THE CURRENT
***********************************************************************
*                                                                     *
*   FILL IN A TABLE ENTRY WITH THE JOB ID AND THE JOB NAME FROM       *
*   THE SJB                                                           *
*                                                                     *
***********************************************************************
FILENTRY EQU   *
         USING TRAKENTY,REG4            CURRENT ENTRY
         MVC   TRKJOBID,SJBJOBID        SAVE THE JOB ID
         MVC   TRKJOBNM,SJBJOBNM        SAVE THE JOB NAME
         DROP  REG1,REG4,REG6           DROP ADDRESSABILITY
         BR    REG14                    RETURN TO CALLER
***********************************************************************
*                                                                     *
*        CHARACTER CONSTANTS                                          *
*                                                                     *
***********************************************************************
IEF099I  DC    CL8'$HASP373'             MESSAGE ID
JOBSRUN  DC    CL8'JOBSRUN '             TABLE ID
***********************************************************************
*                                                                     *
*        NUMERIC CONSTANTS                                            *
*                                                                     *
***********************************************************************
STORGAMT DC    A(ENTRYLTH*1000+HDRLTH)   AMOUNT OF STORAGE NEEDED FOR   X
                                         ONE THOUSAND TABLE ENTRIES
```

```
***********************************************************************
*                                                                     *
*          JOBS RUN TRACKING TABLE                                    *
*                                                                     *
***********************************************************************
TRAKTABL DSECT
TRKACRO  DS    CL8
TRKENTRY DS    A                                ADDRESS OF FIRST ENTRY
TRKCURR  DS    A                                ADDRESS OF CURRENT ENTRY
TRKMXENT DS    F                                MAXIMUM NUMBER OF ENTRIES
TRKNUM   DS    A                                NUMBER OF ENTRIES INSERTED
TRKLNGTH DS    F                                LENGTH OF EACH ENTRY
HDRLTH   EQU   *-TRAKTABL                       LENGTH OF THE TABLE HEADER
TRKSTART EQU   *                                BEGINNING OF THE FIRST ENTRY
TRAKENTY DSECT                                  TABLE ENTRY
TRKJOBID DS    CL8                              JOB ID
TRKJOBNM DS    CL8                              JOB NAME
ENTRYLTH EQU   *-TRAKENTY                       LENGTH OF ONE ENTRY
         IEZVX100
         CVT   DSECT=YES
         IHAPSA
         IKJTCB
         IEZJSCB
         IEFJSSIB
         $SJB
         END   JOBTRACK
```

IEFDB401

Functional Description - (Allocation Input Validation Routine)

The IEFDB401 user exit from the allocation control routine provides for a user-written routine to either validate or alter any request to SVC 99. The user exit routine is entered for all system and user SVC 99 requests. The routine must be coded so as not to interfere with system requests made to the SVC 99 macro. Refer to *MVS/XA SPL: System Macros and Facilities - Volume 1* for more information on the SVC 99 macro.

The user validation routine can test and modify the SVC 99 input request, and it might indicate through a return code whether or not processing of the request is to continue. For example, the user routine might perform the following functions:

- Control the amount of direct access space requested.
- Check for authorization to use specified units.
- Check for authorization to use certain data sets.
- Check for authorization to hold certain resources for reuse.

Entry Specifications

Upon entry to the user written routine, the register contents are as follows:

Register	Contents
0	irrelevant
1	points to a parameter list
2-12	irrelevant
13	points to a save area
14	points to the return address
15	entry address

Return Specifications

Upon return from the user exit routine, the registers must contain:

Register	Contents	
0-14	same as on entry	
15	a return code defined as follows:	
	0	the SVC 99 request processing is to continue.
	nonzero	the SVC 99 request is to be terminated.

Programming Considerations

The user validation routine, which can reside above or below 16 megabytes, must observe the following programming conventions and must receive the following input:

- It must use standard entry and exit linkages.
- Its CSECT name must be IEFDB401 and must reside in its own load module IEFDB401 in SYS1.LPALIB.
- If processing of the request is to continue, the routine must use register 15 to return a code of zero to SVC 99, or any other code if processing is to terminate.
- It receives control in supervisor state under the scheduler's protection key (KEY=1). At entry, register 1 points to a list of addresses for the following parameters:
  - A copy of the SVC 99 input request block, text unit pointers, and text units in scheduler-key fetch-protected storage.
  - The address of a work area for the use of the routine. This area is contiguous with the text unit pointer list so that it can be used to extend the list and provide additional text units.
  - A fullword that contains the length of the work area (500 bytes).
  - The eight-character job name.
  - The twenty-byte programmer name
  - An area that contains accounting information from the JOB statement. The first byte of this area contains the number of accounting fields; the accounting fields follow this byte. Each entry for an accounting field contains the length of the field (one byte, binary), followed by the field itself. The entry for a null field contains a length of zero.
  - The eight-character step name
  - The eight-character program name
  - An area containing accounting information from the EXEC statement. The first byte of this area contains the number of accounting fields (0 for no fields); the accounting fields follow this byte. Each entry for an accounting field contains the length of the field (one byte, binary), followed by the field itself. The entry for a null field contains a length of zero.

The IBM-supplied routine that your routine can replace allows all requests to continue processing.

ISGGREX0

Functional Description - (Scanning the ENQ/DEQ/RESERVE Resource Name Lists)

The ISGGREX0 user exit is supplied in support of the global resource serialization component of MVS/XA as well as MVS/370.

The ISGGREX0 exit is invoked by ENQ/DEQ/RESERVE mainline routines (ISGLNQDQ and ISGGQWBI). The routine scans the input resource name list (RNL) for the resource name specified in the input parameter element list (PEL). A return code is provided by the exit routine indicating whether or not the input resource name is contained in the RNL.

The ISGGREX0 exit routine provided by IBM scans any of the three RNL's. Even though one routine is provided to perform the scan, it is invoked using three different external names:

- ISGGSIEX scans the SYSTEM inclusion RNL.
- ISGGSEEX scans the SYSTEM exclusion RNL.
- ISGGRCEX scans the RESERVE conversion RNL.

Each of these external entry point names and its corresponding RNL address have been defined in the global resource serialization vector table (GVT). The ENQ/DEQ/RESERVE mainline routines invoke the appropriate exit routine although each entry point maps to the common scan routine provided by global resource serialization.

You can change these exits to perform processing necessary for your particular system. When making these changes, you can use the source version of the exit routine provided as a member named ISGGREXS in SYS1.ASAMPLIB. Unlike the version provided in SYS1.NUCLEUS, this version of the exit routine contains logic for excluding temporary data sets from global resource serialization.

Entry Specifications

Upon entry to the ISGGREX0 routine, the register contents are as follows:

Register	Contents
0	for ISGGSIEX, the address of the SYSTEM inclusion RNL. for ISGGRCEX, the address of the RESERVE conversion RNL. for ISGGSEEX, the address of the SYSTEMS exclusion RNL.
1	address of PEL
2-12	irrelevant
13	savearea address
14	return address
15	entry point address

Return Specifications

Upon return from the ISGGREX0 routine, the registers must contain:

Register	Contents
0-14	same as on input
15	return code 0 name found 4 name not found

Your code must restore all general registers.

Programming Considerations

The installation can replace the exit routine, ISGGREX0, without changes to GVT or the ENQ/DEQ/RESERVE mainline routines, as long as the following three entry point names are defined:

- ISGGSIEX
- ISGGSEEX
- ISGGRCEX

The installation can either have three separate routines (one routine for each entry point) or one common routine.

The recovery routine for this module is ISGGFRR0. It depends on the following labels being defined:

ISGGREX0	CSECT name
ISGGREXE	end of executable code
ISGGREXM	label on the MODID macro

If an error occurs which causes ISGGFRR0 to receive control, the recovery routine compares the address in the PSW at the time of the error to the addresses of ISGGREX0 and ISGGREXE. If the failing PSW address lies within the two, ISGGFRR0 assumes that ISGGREX0 failed.

The recovery routine uses only the product level and compile date generated by the MODID macro. The CSECT name is assumed to be ISGGREX0.

Restrictions and Limitations

Your code must not issue any SVC's except ABEND.

To include your code:

1. Use ISGGREX0 as the CSECT name of your code, ISGGREXE as the end of module name, and ISGGREXM as the label of the MODID macro instruction, where ISGGREXE and ISGGREXM are identified as entry symbols. Recovery routine ISGGFRR0 depends on these names.

2. Assemble CSECT ISGGREX0.

3. Link ISGGREX0 with the REPLACE option replacing it with your CSECT into DISTLIB AOSC5.

4. Link IEANUC01 from SYS1.NUCLEUS, include ISGGREX0 from DISTLIB AOSC5 and replace in SYS1.NUCLEUS.

5. Update SMPCDS to reflect the change for regression messages.

The exit routine in each system belonging to the same global resource serialization complex must yield the same scan results, otherwise, resource integrity cannot be guaranteed.

Each entry point is invoked in the requestor's address space (the address space in which the ENQ/DEQ/RESERVE macro instruction is issued). At entry the local lock of the requestor's address space is held as well as the CMSEQDQ class lock. The CMSEQDQ class lock prevents subsequent ENQ/DEQ/RESERVE requests from being processed. Thus, for system performance reasons, keep the processing performed by the exit routine to a minimum.

Operational Considerations

The attributes of the search routine are:

- Nucleus resident
- Supervisor state
- Key 0
- Work unit, task
- With or without PT and SSAR authority
- Linkage, BALR
- AMODE (24)
- RMODE (any)

Section 3: User Exit Directory

This section contains a list of user exits that are coded into the various components of MVS/XA. Each user exit has an entry in the directory that contains its id and component names. A short description or title of each user exit is also included. The name and order number of the book that contains the complete documentation of the user exit are the final items in each entry in the directory.

Those user exits included in Section 2 of this book also have entries in this directory.

Exit	Component	Description	Source/Order No.
PRDMP user defined verbs	BCP	Format and print dump data sets created by standalone, SVC or SYSMDUMP dump services	*MVS/XA SPL: Service Aids*, GC28-1159
AMDUSRxx	BCP	Allows user to format all user trace records of a specified type	*MVS/XA SPL: Service Aids*, GC28-1159
Interpret Table	ACF/VTAM	Use an interpret table to (1) to determine the application program to which a nonstandard logon is directed or (2) to determine the actual name of an application program from the symbolic name specified in a standard logon. Entries in the table can contain either the actual name of the application program or the name of a logon - interpret routine (see number 4) that will determine the name	*Advanced Communications Function for VTAM Version 2 Planning and Installation Reference*, SC27-0610
Logon Interpret Routine	ACF/VTAM	A logon-interpret table routine is pointed to by an entry in an interpret table (see number 3). It determines the name of the application program that is to receive the logons.	*Advanced Communications Function for VTAM Version 2 Planning and Installation Reference*, SC27-0610
EDIT EXIT	TSO	Installation exits for RENUM, MOVE and COPY subcommands of EDIT	*MVS/XA SPL: TSO*, GC28-1173
ERBMFDUC	RMF	Internal processing	*MVS/XA Resource Measurement Facility Reference and User Guide*, LC28-1138
ERBMFEVT	RMF	User sampler	*MVS/XA Resource Measurement Facility, Reference and User Guide*, LC28-1138
ERBMFIUC	RMF	Monitor I session initialization	*MVS/XA Resource Measurement Facility Reference and User Guide*, LC28-1138
ERBMFPUS	RMF	Post processor	*MVS/XA Resource Measurement Facility Reference and User Guide*, LC28-1138
ERBMFRUR	RMF	Report writer	*MVS/XA Resource Measurement Facility Reference and User Guide*, LC28-1138
ERBMFTUR	RMF	Termination Exit	*MVS/XA Resource Measurement Facility Reference and User Guide*, LC28-1138
ERBTRACE	RMF	Field tracing	*MVS/XA Resource Measurement Facility Reference and User Guide*, LC28-1138
EXIT 0[1]	JES2	Pre-initialization process	*MVS/XA SPL: JES2 User Modifications and Macros*, LC23-0069
EXIT 1[1]	JES2	Print/Punch separator	*MVS/XA SPL: JES2 User Modifications and Macros*, LC23-0069
EXIT 2[1]	JES2	Job statement scan	*MVS/XA SPL: JES2 User Modifications and Macros*, LC23-0069
EXIT 3[1]	JES2	Job accounting field scan	*MVS/XA SPL: JES2 User Modifications and Macros*, LC23-0069
EXIT 4[1]	JES2	JCL and JES2 control statement scan	*MVS/XA SPL: JES2 User Modifications and Macros*, LC23-0069
EXIT 5[1]	JES2	JES2 command processor	*MVS/XA SPL: JES2 User Modifications and Macros*, LC23-0069
EXIT 6[1]	JES2	Internal text scan	*MVS/XA SPL: JES2 User Modifications and Macros*, LC23-0069
EXIT 7[1]	JES2	JCT read/write (JES2)	*MVS/XA SPL: JES2 User Modifications and Macros*, LC23-0069

[1] JES2 exit names are user defined.

Exit	Component	Description	Source/Order No.
EXIT 8[1]	JES2	JCT read/write (USER)	*MVS/XA SPL: JES2 User Modifications and Macros*, LC23-0069
EXIT 9[1]	JES2	Job output overflow	*MVS/XA SPL: JES2 User Modifications and Macros*, LC23-0069
EXIT 10[1]	JES2	$WTO screen	*MVS/XA SPL: JES2 User Modifications and Macros*, LC23-0069
EXIT 11[1]	JES2	Spool partitioning allocation ($TRACK)	*MVS/XA SPL: JES2 User Modifications and Macros*, LC23-0069
EXIT 12[1]	JES2	Spool partitioning allocation ($TRAK)	*MVS/XA SPL: JES2 User Modifications and Macros*, LC23-0069
EXIT 13[1]	JES2	TSO/E interactive data transmission facility screening and notification	*MVS/XA SPL: JES2 User Modifications and Macros*, LC23-0069
EXIT 14[1]	JES2	Job queue work select - $QGET	*MVS/XA SPL: JES2 User Modifications and Macros*, LC23-0069
EXIT 15[1]	JES2	Output data set/copy select	*MVS/XA SPL: JES2 User Modifications and Macros*, LC23-0069
EXIT 16[1]	JES2	Notify	*MVS/XA SPL: JES2 User Modifications and Macros*, LC23-0069
EXIT 17[1]	JES2	RJE SIGNON/SIGNOFF	*MVS/XA SPL: JES2 User Modifications and Macros*, LC23-0069
EXIT 18[1]	JES2	SNA RJE LOGON/LOGOFF	*MVS/XA SPL: JES2 User Modifications and Macros*, LC23-0069
EXIT 19[1]	JES2	Initialization statement	*MVS/XA SPL: JES2 User Modifications and Macros*, LC23-0069
EXIT 20[1]	JES2	End of input	*MVS/XA SPL: JES2 User Modifications and Macros*, LC23-0069
EXIT 21[1]	JES2	SMF record	*MVS/XA SPL: JES2 User Modifications and Macros*, LC23-0069
EXIT 22[1]	JES2	Cancel/status	*MVS/XA SPL: JES2 User Modifications and Macros*, LC23-0069
EXIT 23[1]	JES2	JSPA modification for data set separator	*MVS/XA SPL: JES2 User Modifications and Macros*, LC23-0069
EXIT 24[1]	JES2	Post initialization data set separator	*MVS/XA SPL: JES2 User Modifications and Macros*, LC23-0069
IATUX01	JES3	Reserved name	
IATUX02	JES3	Determine action to take when JES3 is unable to find procedure name	*MVS/XA SPL: JES3 User Modifications and Macros*, LC28-1372 or LC28-1371[2]
IATUX03	JES3	Examine or modify internal text created from JCL	*MVS/XA SPL: JES3 User Modifications and Macros*, LC28-1372 or LC28-1371[2]
IATUX04	JES3	Examine the job information from the JCL	*MVS/XA SPL: JES3 User Modifications and Macros*, LC28-1372 or LC28-1371[2]
IATUX05	JES3	Examine the step information from the JCL	*MVS/XA SPL: JES3 User Modifications and Macros*, LC28-1372 or LC28-1371[2]
IATUX06	JES3	Examine DD statement information from the JCL	*MVS/XA SPL: JES3 User Modifications and Macros*, LC28-1372 or LC28-1371[2]
IATUX07	JES3	Examine or substitute unit, type and volume serial information	*MVS/XA SPL: JES3 User Modifications and Macros*, LC28-1372 or LC28-1371[2]

[1] JES2 exit names are user defined.

[2] For JES3 complexes using any release of MVS/SP-JES3 Version 1 that includes JES3 SP 1.3.4

Exit	Component	Description	Source/Order No.
IATUX08	JES3	Examine setup information	*MVS/XA SPL: JES3 User Modifications and Macros*, LC28-1372 or LC28-1371[2]
IATUX09	JES3	Examine final job status, JST and JVT	*MVS/XA SPL: JES3 User Modifications and Macros*, LC28-1372 or LC28-1371[2]
IATUX10	JES3	Generate a message	*MVS/XA SPL: JES3 User Modifications and Macros*, LC28-1372 or LC28-1371[2]
IATUX11	JES3	Inhibit printing of the LOCATE request or response	*MVS/XA SPL: JES3 User Modifications and Macros*, LC28-1372 or LC28-1371[2]
IATUX15	JES3	Scan an initialization statement	*MVS/XA SPL: JES3 User Modifications and Macros*, LC28-1372 or LC28-1371[2]
IATUX16	JES3	Reserved name	
IATUX17	JES3	Define set of scheduler elements	*MVS/XA SPL: JES3 User Modifications and Macros*, LC28-1372 or LC28-1371[2]
IATUX18	JES3	Check console authority level	*MVS/XA SPL: JES3 User Modifications and Macros*, LC28-1372 or LC28-1371[2]
IATUX19	JES3	Examine or modify temporary OSE	*MVS/XA SPL: JES3 User Modifications and Macros*, LC28-1372 or LC28-1371[2]
IATUX20	JES3	Examine or modify data written on job header pages	*MVS/XA SPL: JES3 User Modifications and Macros*, LC28-1372 or LC28-1371[2]
IATUX21	JES3	Examine or modify data written on data set header labels	*MVS/XA SPL: JES3 User Modifications and Macros*, LC28-1372 or LC28-1371[2]
IATUX22	JES3	Examine or alter the forms alignment	*MVS/XA SPL: JES3 User Modifications and Macro*, LC28-1372 or LC28-1371[2]
IATUX23	JES3	Examine or modify data written to trailer pages	*MVS/XA SPL: JES3 User Modifications and Macros*, LC28-1372 or LC28-1371[2]
IATUX24	JES3	Examine the Net-id and the devices requested	*MVS/XA SPL: JES3 User Modifications and Macros*, LC28-1372 or LC28-1371[2]
IATUX25	JES3	Examine or modify volume serial number	*MVS/XA SPL: JES3 User Modifications and Macros*, LC28-1372 or LC28-1371[2]
IATUX26	JES3	Examine MVS scheduler control blocks	*MVS/XA SPL: JES3 User Modifications and Macros*, LC28-1372 or LC28-1371[2]
IATUX27	JES3	Examine or alter the JDAB, JCT and JMR	*MVS/XA SPL: JES3 User Modifications and Macros*, LC28-1372 or LC28-1371[2]
IATUX28	JES3	Examine the accounting information as provided by the JOB statement	*MVS/XA SPL: JES3 User Modifications and Macros*, LC28-1372 or LC28-1371[2]
IATUX29	JES3	Examine the accounting information as provided by the JCT, JDAB and JMR	*MVS/XA SPL: JES3 User Modifications and Macros*, LC28-1372 or LC28-1371[2]
IATUX30	JES3	Examine authority level for TSO terminal commands	*MVS/XA SPL: JES3 User Modifications and Macros*, LC28-1372 or LC28-1371[2]
IATUX31	JES3	Examine or modify destination or message text	*MVS/XA SPL: JES3 User Modifications and Macros*, LC28-1372 or LC28-1371[2]
IATUX32	JES3	Override the DYNALDSN initialization statement	*MVS/XA SPL: JES3 User Modifications and Macros*, LC28-1372 or LC28-1371[2]
IATUX33	JES3	JES3 control statement & JCL EXEC statement user exit	*MVS/XA SPL: JES3 User Modifications and Macros*, LC28-1372 or LC28-1371[2]

[2] For JES3 complexes using any release of MVS/SP JES3 Version 1 that includes JES3 SP 1.3.4

Exit	Component	Description	Source/Order No.
IATUX34	JES3	JCL DD statement user exit	*MVS/XA SPL: JES3 User Modifications and Macros*, LC28-1372 or LC28-1371[2]
IATUX35	JES3	Validity check network commands	*MVS/XA SPL: JES3 User Modifications and Macros*, LC28-1372 or LC28-1371[2]
IATUX36	JES3	Collect accounting information	*MVS/XA SPL: JES3 User Modifications and Macros*, LC28-1372 or LC28-1371[2]
IATUX37	JES3	Modify the JES3 networking data set header	*MVS/XA SPL: JES3 User Modifications and Macros*, LC28-1372 or LC28-1371[2]
IATUX38	JES3	Change SYSOUT class for networking data sets	*MVS/XA SPL: JES3 User Modifications and Macros*, LC28-1372 or LC28-1371[2]
IATUX39	JES3	Modify the data set header for a SYSOUT data set	*MVS/XA SPL: JES3 User Modifications and Macros*, LC28-1372 or LC28-1371[2]
IATUX40	JES3	Modify job header	*MVS/XA SPL: JES3 User Modifications and Macros*, LC28-1372 or LC28-1371[2]
IATUX41	JES3	Determines the disposition of job over JCL limit	*MVS/XA SPL: JES3 User Modifications and Macros*, LC28-1372 or LC28-1371[2]
IATUX42	JES3	TSO interactive data transmission facility screening and notification	*MVS/XA SPL: JES3 User Modifications and Macros*, LC28-1372 or LC28-1371[2]
IATUX43	JES3	Modify job header segments	*MVS/XA SPL: JES3 User Modifications and Macros*, LC28-1372 or LC28-1371[2]
IATUX44	JES3	Examine and modify the JCL	*MVS/XA SPL: JES3 User Modifications and Macros*, LC28-1372 or LC28-1371[2]
IATUX45	JES3	Examine and modify data sent to an output writer FSS	*MVS/XA SPL: JES3 User Modifications and Macros*, LC28-1372 or LC28-1371[2]
IATUX46	JES3	Select processors eligible for C/I processing	*MVS/XA SPL: JES3 User Modifications and Macros*, LC28-1372 or LC28-1371[2]
IATUX47	JES3	Delete or save held output datasets	*MVS/XA SPL: JES3 User Modifications and Macros*, LC28-1372 or LC28-1371[2]
IATUX48	JES3	Reserved name	
IATUX49	JES3	Override address space selected for C/I processing	*MVS/XA SPL: JES3 User Modifications and Macros*, LC28-1372 or LC28-1371[2]
IATUX50	JES3	Process user defined BSIDMOD codes	*MVS/XA SPL: JES3 User Modifications and Macros*, LC28-1372 or LC28-1371[2]
IATUX56	JES3	Authority check JES3 commands issued via BDTTSO/BDTBATCH	*MVS/XA SPL: JES3 User Modifications and Macros*, LC28-1372 or LC28-1371[2]
IATUX61	JES3	During MDS processing, chooses whether a job should be canceled or sent to the error queue	*MVS/XA SPL: JES3 User Modifications and Macros*, LC28-1372
IATUX62	JES3	Overrides the decision to accept a tape or disk mount	*MVS/XA SPL: JES3 User Modifications and Macros*, LC28-1372
IEALIMIT	VSM	Allows user to enforce a region size limit and a GETMAIN limit in the private area (below 16 megabytes)	*MVS/XA SPL: User Exits*, GC28-1147
IEAVADFM	RTM	Allows user to format SNAP, SYSABEND and SYSUDUMP dumps	*MVS/XA SPL: User Exits*, GC28-1147
IEAVADUS	RTM	Allows user to select and format dump data	*MVS/XA SPL: User Exits*, GC28-1147
IEAVTABX	RTM	Gives control to user pre-dump exit that can alter SNAP/ABEND dump options or suppress dump	*MVS/XA SPL: User Exits*, GC28-1147

[2] For JES3 complexes using any release of MVS/SP JES3 Version1 that includes JES3 SP 1.3.4

Exit	Component	Description	Source/Order No.
IEAVTSEL	RTM	Gives control to user exit routines found in IEAVTSEL after dump is taken; routines perform post dump processing	*MVS/XA SPL: User Exits*, GC28-1147
IEECVXIT	BCP	Alter routing or descriptor codes for WTO and codes for WTO and WTOR messages	*MVS/XA SPL: User Exits*, GC28-1147
IEFACTRT	SMF	Termination exit	*MVS/XA SPL: SMF*, GC28-1153
IEFDB401	BCP	Allows user to validate and alter an SVC 99 request	*MVS/XA SPL: User Exits*, GC28-1147
IEFUJI	SMF	Job initiation	*MVS/XA SPL: SMF*, GC28-1153
IEFUJP	SMF	Job purge	*MVS/XA SPL: SMF*, GC28-1153
IEFUJV	SMF	Job validation	*MVS/XA SPL: SMF*, GC28-1153
IEFUSI	SMF	Step initiation	*MVS/XA SPL: SMF*, GC28-1153
IEFUSO	SMF	SYSOUT limit	*MVS/XA SPL: SMF*, GC28-1153
IEFUTL	SMF	Time limit	*MVS/XA SPL: SMF*, GC28-1153
IEFU29	SMF	SMF dump	*MVS/XA SPL: SMF*, GC28-1153
IEFU83	SMF	SMF record	*MVS/XA SPL: SMF*, GC28-1153
IEFU84	SMF	SMF record	*MVS/XA SPL: SMF*, GC28-1153
IEFXVNSL	BCP	Allows user to replace label with non-standard format	*MVS/XA Tape Labels*, GC26-4003
IKJEFLD	TSO	Customize LOGON procedure for installation users	*MVS/XA SPL: TSO*, GC28-1173 (Supp with TSO/E - SD23-0267)
IKJEFF10	TSO	SUBMIT exit-allows terminal user to initiate background job	*MVS/XA SPL: TSO*, GC28-1173 (Supp. with TSO/E - SD23-0267)
IKJEFF53	TSO	Controls conditions under which CANCEL, STATUS and OUTPUT commands are allowed	*MVS/XA SPL: TSO*, GC28-1173 (Supp. with TSO/E - SD23-0267)
ISGGREX0	SCSDS	Controls scan of Global Resource Serialization resource name lists (RNL)	*MVS/XA SPL: User Exits*, GC28-1147
ISTAUCAG	ACF/VTAM	Calculates and records time during which a terminal user or application program is logged onto an application program	*Advanced Communications Function for VTAM Version 2 Planning and Installation Reference*, SC27-0610
ISTAUCAT	ACF/VTAM	Validates a logon request to application program	*Advanced Communications Function for VTAM Version 2 Planning and Installation Reference*, SC27-0610
ISTEXCVR	ACF/VTAM	Provides ACF/VTAM with ordered list of virtual routes for path selection to transmit data through network	*Advanced Communications Function for VTAM Version 2 Planning and Installation Reference*, SC27-0610
ISTINCDT	ACF/VTAM	User supplies supplementary tables to session-link unformatted system services (USS) table which ACF/VTAM uses to handle command input and message output.	*Advanced Communications Function for VTAM Version 2 Planning Installation and Reference*, SC27-0610
ISTINCLM	ACF/VTAM	User can change the IBM-supplied logon mode table which contains parameters contains parameters representing protocols for telecommunications session or supply supplementary tables	*Advanced Communications Function for VTAM Version 2 Planning Installation and Reference*, SC27-0610
ISTINCNO	ACF/VTAM	User supplies supplementary tables to operation-level USS tables which handle commands from ACF/VTAM operator and messages to an ACF/VTAM operator	*Advanced Communications Function for VTAM Version 2 Planning and Installation Reference*, SC27-0610

Exit	Component	Description	Source/Order No.
ISTMGC00	ACF/VTAM	Communication network management (CNM) table routes unsolicited network service requests to application programs to record and report maintenance statistics	*Advanced Communications Function for VTAM Version 2 Planning and Installation Reference*, SC27-0610
ISTPUCWC	ACF/VTAM	Virtual route pacing window size calculator specifies limits to data flow through a network to avoid congestion in nodes along a virtual route	*Advanced Communications Function for VTAM Version 2 Planning and Installation Reference*, SC27-0610
ISTRACON	ACF/VTAM	Module containing constants used to control functions not suitable for modification by operator command or start option	*Advanced Communications Function for VTAM Version 2 Planning Installation and Reference*, SC27-0610
ISTSCOS	ACF/VTAM	Class of service (COS) table provides an ordered list of routes for selection of a path for transmitting data through a network	*Advanced Communications Function for VTAM Version 2 Planning Installation and Reference*, SC27-0610
EODAD	VSAM	End of data exit routine	*MVS/XA Data Management Guide*, GC26-4013 (DFP Version 1), GC26-4140 (DFP Version 2)
SYNAD	ISAM	Synchronous Error routine	*MVS/XA Data Management Guide*, GC26-4013 (DFP Version 1), GC26-4140 (DFP Version 2)
EXLST	DFP	Specifies address of a list of exit routines	*MVS/XA Data Management Guide*, GC26-4013 (DFP Version 1), GC26-4140 (DFP Version 2)
Inactive Entry	DFP	Ignore the inactive entry	*MVS/XA Data Management Guide*, GC26-4013 (DFP Version 1), GC26-4140 (DFP Version 2)
Input Header Label	DFP	Process a user input header	*MVS/XA Data Management Guide*, GC26-4013 (DFP Version 1), GC26-4140 (DFP Version 2)
Output Header label	DFP	Create a user output header label	*MVS/XA Data Management Guide*, GC26-4013 (DFP Version 1), GC26-4140 (DFP Version 2)
Input Trailer label	DFP	Process a user input trailer label	*MVS/XA Data Management Guide*, GC26-4013 (DFP Version 1), GC26-4140 (DFP Version 2)
Output Trailer label	DFP	Create a user output trailer label	*MVS/XA Data Management Guide*, GC26-4013 (DFP Version 1), GC26-4140 (DFP Version 2)
Data Control Block	DFP	Address of routine that completes or modifies DCB	*MVS/XA Data Management Guide*, GC26-4013 (DFP Version 1), GC26-4140 (DFP Version 2)
End-of-Volume	DFP	Address of routine for end-of-volume	*MVS/XA Data Management Guide*, GC26-4013 (DFP Version 1), GC26-4140 (DFP Version 2)
Defer Label	DFP	Defer processing of a user input trailer label from end-of-data until closing	*MVS/XA Data Management Guide*, GC26-4013 (DFP Version 1), GC26-4140 (DFP Version 2)
Defer non-Standard label	DFP	Allows the user to specify a code in order to defer processing a non-standard input trailer label from end-of-data until closing	*MVS/XA Data Management Guide*, GC26-4013 (DFP Version 1), GC26-4140 (DFP Version 2)
DCB Abend	DFP	Provides system option for ABEND condition during processing or opening, closing or handling end-of-volume condition for associated DCB	*MVS/XA Data Management Guide*, GC26-4013 (DFP Version 1), GC26-4140 (DFP Version 2)
JFCBE	DFP	Address of routine to examine or modify JCL-specified setup for 3800 printer	*MVS/XA Data Management Guide*, GC26-4013 (DFP Version 1), GC26-4140 (DFP Version 2)

Exit	Component	Description	Source/Order No.
Format/DSCB Not-Found	DFP	Determines if a missing DSCB can be restored to the volume	*MVS/XA SPL: Data Administration*, GC26-4010 (DFP Version 1), GC26-4149 (DFP Version 2)
JRNAD	VSAM	Records contents of buffers for I/O errors/ record transactions	*MVS/XA VSAM Administration: Macro Instruction Reference*, GC26-4016 (DFP Version 1), GC26-4152 (DFP Version 2)
ERROR	IEBCOMPR	Routine for not equal comparison	*MVS/XA Utilities*, GC26-4018 (DFP Version 1), GC26-4150 (DFP Version 2)
INHDR	IEBCOMPR	Processes user input header labels	*MVS/XA Utilities*, GC26-4018 (DFP Version 1), GC26-4150 (DFP Version 2)
INTLR	IEBCOMPR	Processes user input trailer labels	*MVS/XA Utilities*, GC26-4018 (DFP Version 1), GC26-4150 (DFP Version 2)
PRECOMP	IEBCOMPR	Processes logical records >32K	*MVS/XA Utilities*, GC26-4018 (DFP Version 1), GC26-4150 (DFP Version 2)
DATA	IEBGENER	Modifies physical record	*MVS/XA Utilities*, GC26-4018 (DFP Version 1), GC26-4150 (DFP Version 2)
INHDR	IEBGENER	Modifies physical record	*MVS/XA Utilities*, GC26-4018 (DFP Version 1), GC26-4150 (DFP Version 2)
INTLR	IEBGENER	Modifies physical record	*MVS/XA Utilities*, GC26-4018 (DFP Version 1), GC26-4150 (DFP Version 2)
IOERROR	IEBGENER	Process permanent I/O conditions	*MVS/XA Utilities*, GC26-4018 (DFP Version 1), GC26-4150 (DFP Version 2)
KEY	IEBGENER	Creation of output record key	*MVS/XA Utilities*, GC26-4018 (DFP Version 1), GC26-4150 (DFP Version 2)
OUTHDR	IEBGENER	Creates user output header labels	*MVS/XA Utilities*, GC26-4018 (DFP Version 1), GC26-4150 (DFP Version 2)
OUTTLR	IEBGENER	Creates user output trailer labels	*MVS/XA Utilities*, GC26-4018 (DFP Version 1), GC26-4150 (DFP Version 2)
TOTAL	IEBGENER	Allows user routine to alter record prior writing output	*MVS/XA Utilities*, GC26-4018 (DFP Version 1), GC26-4150 (DFP Version 2)
INHDR	IEBPTPCH	Modifies physical record	*MVS/XA Utilities*, GC26-4018 (DFP Version 1), GC26-4150 (DFP Version 2
INREC	IEBPTPCH	Allows for manipulation of each logical record before it is processed	*MVS/XA Utilities*, GC26-4018 (DFP Version 1), GC26-4150 (DFP Version 2)
INTLR	IEBPTPCH	Modifies physical record	*MVS/XA Utilities*, GC26-4018 (DFP Version 1), GC26-4150 (DFP Version 2)
OUTREC	IEBPTPCH	Allows for manipulation of each logical record before it is printed or punched	*MVS/XA Utilities*, GC26-4018 (DFP Version 1), GC26-4150 (DFP Version 2)

Exit	Component	Description	Source/Order No.
ERROR	IEBTCRIN	Receives control before an error record is passed to the error output data set	*MVS/XA Utilities*, GC26-4018 (DFP Version 1), GC26-4150 (DFP Version 2)
OUTREC	IEBTCRIN	Allows for manipulation of each logical record before it is printed or punched	*MVS/XA Utilities*, GC26-4018 (DFP Version 1), GC26-4150 (DFP Version 2)
OUTHDR2	IEBTCRIN	Routine receives control during opening of SYSUT2 data set	*MVS/XA Utilities*, GC26-4018 (DFP Version 1), GC26-4150 (DFP Version 2)
OUTHDR3	IEBTCRIN	Routine receives control during opening of SYSUT3 data set	*MVS/XA Utilities*, GC26-4018 (DFP Version 1), GC26-4150 (DFP Version 2)
OUTTLR2	IEBTCRIN	Routine receives control during closing of SYSUT2 data set	*MVS/XA Utilities*, GC26-4018 (DFP Version 1), GC26-4150 (DFP Version 2)
OUTTLR3	IEBTCRIN	Routine receives control during closing of SYSUT3 data set	*MVS/XA Utilities*, GC26-4018 (DFP Version 1), GC26-4150 (DFP Version 2)
LERAD	VSAM	Allows for analysis of logical errors	*MVS/XA VSAM Administration Guide*, GC26-4015 (DFP Version 1), GC26-4151 (DFP Version 2)
SYNAD	VSAM	Allows for analysis of physical errors	*MVS/XA VSAM Administration Guide*, GC26-4015 (DFP Version 1), GC26-4151 (DFP Version 2)
EXCEPTION	VSAM	Routine monitors I/O errors associated with a data set	*MVS/XA VSAM Administration Guide*, GC26-4015 (DFP Version 1), GC26-4151 (DFP Version 2)
UPAP	VSAM	Routine for user processing	*MVS/XA VSAM Administration Guide*, GC26-4015 (DFP Version 1), GC26-4151 (DFP Version 2)
Datestamp	VSAM	Modules replacement to cause timestamp processing to be skipped or used with specified data sets only	*MVS/XA VSAM Administration Guide*, GC26-4015 (DFP Version 1), GC26-4151 (DFP Version 2)
USER1 (name)	SMF	User exit routine is given control after each record is read	*MVS/XA SPL: SMF*, GC28-1153
USER2 (name)	SMF	User exit routine is given control when the dump program selects a record to be written	*MVS/XA SPL: SMF*, GC28-1153
USER3 (name)	SMF	User exit routine is given control after the output data set is closed	*MVS/XA SPL: SMF*, GC28-1153

Exit	Component	Description	Source/Order No.
BEXIT= operand of STARTMH macro	ACF/TCAM	Bind exit. Allows user exit to modify bind parameters, to review the virtual list to disallow the session, and to request the route availability monitoring option	*Advanced Communications Function for TCAM Version 2 Base Installation Guide*, SC30-3132 *Advanced Communications Function for TCAM Version 2 Networking Installation Guide*, SC30-3153 *Advanced Communications Function for TCAM Version 2 Installation Reference*, SC30-3133 (BEXIT operand of STARTMH macro)
BEXIT= operand of STARTMH macro	ACF/TCAM	Unbind exit. Allows user to notify an application program or any other destination that a LU-LU or pseudo LU-LU session is terminating and to specify that a LU-LU or pseudo LU-LU session be automatically reinitiated	*Advanced Communications Function for TCAM Version 2 Base Installation Guide*, SC30-3132 *Advanced Communications Function for TCAM Version 2 Networking Installation Guide*, SC30-3153 *Advanced Communications Function for TCAM Version 2 Installation Reference*, SC30-3133 (BEXIT operand of STARTMH macro)
IEDUSWS or user-supplied name on WSZEXIT	ACF/TCAM	Window-size exit. Allows calculate minimum and maximum window sizes for virtual route paging	*Advanced Communications Function for TCAM Version 2 Networking Installation Guide*, SC30-3153 *Advanced Communications Function for TCAM Version 2 Installation Reference*, SC30-3133 (WSZEXIT operand of INTRO macro)
TCSUP or user-supplied name on GMMSG	ACF/TCAM	Good-morning message. Allows user to generate good morning messages	*Advanced Communications Function for TCAM Version 2 Base Installation Guide*, SC30-3132 *Advanced Communications Function for TCAM Version 2 Installation Sample Programs*, SC30-3134 *Advanced Communications Function for TCAM Version 2 Installation Reference*, SC30-3133 (GMMSG operand of READY macro *Advanced Communications Function for TCAM Version 2 Utilities*, SC30-3138
DKJHMX or user-supplied name on PRIEXIT	ACF/TCAM	Multiple destination message priority routine. Allows user to specify the priority for multiple-destination messages whenever a message is to be queued to any but the first entry in a cascade or distribution list or other than the first of multiple destinations	*Advanced Communications Function for TCAM Version 2 Base Installation Guide*, SC30-3132 *Advanced Communications Function for TCAM Version 2 Installation Sample Programs*, SC30-3134 *Advanced Communications Function for TCAM Version 2 Installation Reference*, SC30-3133 (PRIEXIT operand of READY macro)
DKJKAX or user-supplied name on PURGEXT	ACF/TCAM	Transfer/purge exit. In conjunction with the REDIRECT macro, allows user to transfer messages from one queue to another or to purge messages from a queue	*Advanced Communications Function for TCAM Version 2 Base Installation Guide*, SC30-3132 *Advanced Communications Function for TCAM Version 2 Installation Sample Programs*, SC30-3134 *Advanced Communications Function for VTAM Version 2 Installation Reference*, SC30-3133 (PURGEXT operand of READY macro)
TCSUP or user-supplied name on RSMSG	ACF/TCAM	Restart exit. Allows user to build and send "restart in progress" messages	*Advanced Communications Function for TCAM Version 2 Base Installation Guide*, SC30-3132 *Advanced Communications Function for TCAM Version 2 Installation Sample Programs*, SC30-3134 *Advanced communications Function for TCAM Version 2 Installation Reference*, SC30-3133 (RSMSG operand of READY macro) *Advanced Communications Function for TCAM Version 2 Utilities*, SC30-3138

Exit	Component	Description	Source/Order No.
User-supplied name on EXLST	ACF/TCAM	ABEND exit. Allows registers to be saved and restored when an OPEN macro has failed to execute properly	*Advanced Communications Function for TCAM Version 2 Base Installation Guide*, SC30-3132 *Advanced Communications Function for TCAM Version 2 Installation Reference*, SC30-3133 (EXLST operand of DCB macro and Return code section of OPEN macro)
User-supplied name on EXIT	ACF/TCAM	Allows user to correct the destination of a message, provide another destination for the message, or indicate the message is not to be processed for any destination	*Advanced Communications Function for TCAM Version 2 Base Installation Guide*, SC30-3132 *Advanced Communications Function for TCAM Version 2 Installation Reference*, SC30-3133 (EXIT operand of FORWARD macro)
User-supplied name on EXIT	ACF/TCAM	Allows user to supply the text of a message, to modify the text of the message, or to generate an FHP for the message	*Advanced Communications Function for TCAM Version 2 Base Installation Guide*, SC30-3133 *Advanced Communications Function for TCAM Version 2 Installation Reference*, SC30-3133 (EXIT operand of EXMSG macro)
User-supplied name on EXIT	ACF/TCAM	Allows user to alter error message processing. (can alter the text of the error message before incorporating the text into a header buffer)	*Advanced Communications Function for TCAM Version 2 Base Installation Guide*, SC30-3132 *Advanced Communications Function for TCAM Version 2 Installation Reference*, SC30-3133 (EXIT operand of ERRORMSG macro)
ACCTING or user-supplied name on EXIT	ACF/TCAM	Allows user to construct and generate a message or to exit to the special accounting routine	*Advanced Communications Function for TCAM Version 2 Base Installation Guide*, SC30-3132 *Advanced Communications Function for TCAM Version 2 Installation Reference*, SC30-3133 (EXIT operand of SENDMSG)
User-supplied name on EXIT	ACF/TCAM	Allows user to analyze the status information given by the concentrator and its attached stations	*Advanced Communications Function for TCAM Version 2 Installation Reference*, SC30-3133 (EXIT operand of QACTION macro)
User-supplied name on EXIT	ACF/TCAM	Allows user to redirect a message (must be used in conjunction with the READY macro PURGEXT routine to transfer or purge messages)	*Advanced Communications Function for TCAM Version 2 Installation Reference*, SC30-3133 (EXIT operand of REDIRECT macro)
User-supplied name on EXIT	ACF/TCAM	Allows user to specify an unchanged startup/restart condition or a different startup/restart condition number	*Advanced Communications Function for TCAM Version 2 Installation Reference*, SC30-3133 (EXIT operand of UPCONDTN macro)
IEAVTRML	BCP	Allows user to supply resource management routines	*MVS/XA SPL: System Modifications*, GC28-1152
ADYPSTD	DAE	Schedules a transaction on the DAE transaction processor queue whenever a SVC Dump or a SYSMDUMP completes or is suppressed	*MVS/XA SPL: User Exits*, GC28-1147
IEAVMXIT and user-specified names	BCP	General and User-Specified WTO/WTOR exit routines	*MVS/XA SPL: User Exits*, GC28-1147
IPCS user-defined verbs	IPCS	Format and print dump data sets created by standalone, SVC or SYSMDUMP dumping services	*MVS/XA: IPCS User's Guide and Reference*, GC28-1297
BLSUGWDM	IPCS	Command validation routine for IPCS	*MVS/XA IPCS User's Guide and Reference*, GC28-1297

Exit	Component	Description	Source/Order No.
INMRZ01	TSO	Receives initialization	*MVS/XA SPL: TSO*, GC28-1173 (Supp. with TSO/E - SD23-0267)
INMRZ02	TSO	Receives termination	*MVS/XA SPL: TSO*, GC28-1173 (Supp. with TSO/E - SD23-0267)
INMRZ11	TSO	Receives data set pre-processing	*MVS/XA SPL: TSO*, GC28-1173 (Supp. with TSO/E - SD23-0267)
INMRZ12	TSO	Receives data set post-processing	*MVS/XA SPL: TSO*, GC28-1173 (Supp. with TSO/E - SD23-0267)
INMRZ13	TSO	Receives data set decryption or notification	*MVS/XA SPL: TSO*, GC28-1173 (Supp. with TSO/E - SD23-0267)
INMXZ01	TSO	Transmission start-up	*MVS/XA SPL: TSO*, GC28-1173 (Supp. with TSO/E - SD23-0267)
INMXZ03	TSO	Transmission encryption or termination	*MVS/XA SPL: TSO*, GC28-1173 (Supp. with TSO/E - SD23-0267)
ICHRDX01	RACF	RACDEF pre-processing exit	*RACF Installation Reference Manual*, SC28-0734
ICHRDX02	RACF	RACDEF post-processing exit	*RACF Installation Reference Manual*, SC28-0734
ICHRCX01	RACF	RACHECK pre-processing exit	*RACF Installation Reference Manual*, SC28-0734
ICHRCX02	RACF	RACHECK post-processing exit	*RACF Installation Reference Manual*, SC28-0734
ICHRIX01	RACF	RACINIT pre-processing exit	*RACF Installation Reference Manual*, SC28-0734
ICHRIX02	RACF	RACINIT post-processing exit	*RACF Installation Reference Manual*, SC28-0734
ICHRLX01	RACF	RACLIST pre/post-processing exit	*RACF Installation Reference Manual*, SC28-0734
ICHRLX02	RACF	RACLIST selection exit	*RACF Installation Reference Manual*, SC28-0734
ICHCNX00	RACF	RACF password exit	*RACF Installation Reference Manual*, SC28-0734
ICHCCX00	RACF	RACF password exit	*RACF Installation Reference Manual*, SC28-0734
ICHFRX01	RACF	FRACHECK pre-processing exit	*RACF Installation Reference Manual*, SC28-0734
ICHPWX01	RACF	RACINIT SVC routine	*RACF Installation Reference Manual*, SC28-0734

Index

MVS/Extended Architecture System Programming Library: User Exits

GC28-1147-3 S370-36

Printed in U.S.A.

MVS/Extended Architecture
System Programming
Library: User Exits

READER'S
COMMENT
FORM

GC28-1147-3

This manual is part of a library that serves as a reference source for systems analysts, programmers, and operators of IBM systems. You may use this form to communicate your comments about this publication, its organization, or subject matter, with the understanding that IBM may use or distribute whatever information you supply in any way it believes appropriate without incurring any obligation to you.

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 comment are:

Clarity	Accuracy	Completeness	Organization	Coding	Retrieval	Legibility

If you wish a reply, give your name, company, mailing address, and date:

____

____

____

____

What is your occupation? ______________________________

How do you use this publication? ______________________________

Number of latest Newsletter associated with this publication: ______________________________

Thank you for your cooperation. No postage stamp 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.)

MVS/Extended Architecture System Programming Library: User Exits

GC28-1147-3 S370-36

Reader's Comment Form

Cut or Fold Along Line

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 D58, Building 921-2
PO Box 390
Poughkeepsie, New York 12602

Fold and tape Please Do Not Staple Fold and tape

Printed in U.S.A.

IBM®

MVS/Extended Architecture
System Programming
Library: User Exits

READER'S
COMMENT
FORM

GC28-1147-3

This manual is part of a library that serves as a reference source for systems analysts, programmers, and operators of IBM systems. You may use this form to communicate your comments about this publication, its organization, or subject matter, with the understanding that IBM may use or distribute whatever information you supply in any way it believes appropriate without incurring any obligation to you.

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 comment are:

Clarity   Accuracy   Completeness   Organization   Coding   Retrieval   Legibility

If you wish a reply, give your name, company, mailing address, and date:

____

____

____

____

What is your occupation? ______________________________

How do you use this publication? ______________________________

Number of latest Newsletter associated with this publication: ________________

Thank you for your cooperation. No postage stamp 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.)

MVS/Extended Architecture System Programming Library: User Exits

GC28-1147-3 S370-36

Reader's Comment Form

Cut or Fold Along Line

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 D58, Building 921-2
PO Box 390
Poughkeepsie, New York 12602

Fold and tape Please Do Not Staple Fold and tape

Printed in U.S.A.

IBM®

GC28-1147-03