IBM z/VSE
VSE Central Functions

Supervisor Calls and Internal Macros

z/VSE 4.1

xxxx-xxxx-xx

IBM z/VSE
VSE Central Functions

Supervisor Calls and Internal Macros

z/VSE 4.1

xxxx-xxxx-xx

Note!

Before using this information and the product it supports, be sure to read the general information under “Notices” on page ix.

First Edition 07/11/07

Contents

Notices

References in this publication to IBM products, programs, or services do not imply that IBM intends to make these available in all countries in which IBM operates. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any of the intellectual property rights of IBM may be used instead of the IBM product, program, or service. The evaluation and verification of operation in conjunction with other products, except those expressly designated by IBM, are the responsibility of the user.

IBM may have patents or pending patent applications covering subject matter in this document. The furnishing of this document does not give you any license to these patents. You can send license inquiries, in writing, to the IBM Director of Licensing, IBM Corporation, 500 Columbus Avenue, Thornwood, New York 10594, U.S.A.

Any pointers in this publication to non-IBM Web sites are provided for convenience only and do not in any manner serve as an endorsement. IBM accepts no responsibility for the content or use of non-IBM Web sites specifically mentioned in this publication or accessed through an IBM Web site that is mentioned in this publication.

Programming Interface Information

This book documents information that is NOT intended to be used as a Programming Interface of z/VSE.

Trademarks and Service Marks

The following terms, denoted by an asterisk (*) in this publication, are trademarks of the IBM Corporation in certain countries:

ACF/VTAM
AFP
AT
CICS
CICS/VSE
CUA
ECKD
Enterprise Systems Architecture/370
Enterprise Systems Architecture/390
ESA/370
ESA/390
ES/9000
ESCON400
IBM System z
z/OS
IBM
MVS/ESA
NetView
PR/SM
SQL/DS
System/370
System/390
VM/ESA re-branded to z/VM
VSE/ESA re-branded to z/VSE
VTAM
z/Architecture
OS/390 re-branded to z/OS
z/VSE
z/VM

Preface

This manual is intended primarily for use by IBM personnel responsible for program service. It describes the Supervisor Calls and Internal Macros of the VSE/Advanced Functions Supervisor. The manual supplements the program listings by providing text and charts as follows:

- Chapter 1: Internal Macros
- Chapter 2: VSE Supervisor Call Table
  shows the relationship between the Internal Macros and the Supervisor Calls
- Chapter 3: Supervisor Calls
  Contains a description of the various Supervisor functions

Macro Descriptions

Supervisor Interface Macros

The macros described on the following pages represent a symbolic interface between the VSE/AF supervisor and other SCP sub components, such as non-resident system tasks, IPL, EOT, etc., or IBM licensed programs, like VTAM, VSE/POWER, etc. They are not to be considered as new general purpose user interfaces and will not be included in the SRL documentation.

A specific authorization is required for most of the described functions. This is so because the related interfaces are only committed to restricted classes of users and because the integrity of the system may be affected by an inappropriate usage of some of the functions. In some cases, authorization is restricted to system tasks and to one or more other known components. An easy and fast identification of the requester is useful for this type of authorization checks. Components, which are not initialized by the VSE/AF supervisor itself, are therefore requested to identify themselves to the supervisor by means of a SUBSID NOTIFY macro during their initialization. Note, however, that authorization is related to the code being executed, and may therefore change dynamically. Note also that a protection key of 0 is in many cases neither necessary nor sufficient as authorization criterion. A list of authorized components (besides system tasks) is given whenever applicable. It may be extended in the future.

ALLOCATE

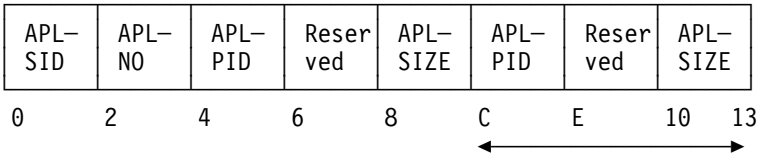
The ALLOCATE macro (see also SVC 83) allocates or reallocates real or virtual (static) partitions. The macro is used by the job control ALLOC(R) command processing, and the attention ALLOC(R) command processing. It may also be used by other system components if applicable.
Execution Mode is AMODE ANY and RMODE 24 (including the parameterlist).
The format is as follows:

```
[name]   ALLOCATE   [APL={name1|(1)}]
```

APL Defines the parameter list into which the specified operands have to be placed before issuing this macro. The address of the parameter list may be supplied either as an operand (name1) or in a register.

The format of the parameter list is as follows:

APL



Repetitive; depending on the number of elements specified in the command.

APLSID It contains the space identifier for which the partitions are to be allocated, with the following values:

```
'0 '     for the primary space
'n '     for the spaces 1 to B
'R '     for real (re)allocation
'S '     for the shared area
x'FFFF'  for the default space
```

If x'FFFF' has been specified, the default space id for the partition is taken and only one partition can be allocated in the space (APLNO=1). Any further reallocation request for that partition must also be done by using defaults and must not increase its initial allocation value. Exception:
If the BG was allocated by using defaults it may be increased up to PASIZE.
x'FFFF' must not be specified for reallocation, if the partition was allocated by specifying the space id explicitly.

APLNO It contains the number of operands which were specified in the ALLOC command, and which is equal to the number of elements in the parameter list.

APLPID It contains the partition id 'BG' or 'Fn', n=1,..,B.

APLSIZE It contains the partition size in KB

Note: For a description of dynamic partition allocation see z/VSE Supervisor Diagnosis Reference, Chapter "Storage Management".

To generate the layout of the parameter list, the following macro may be used:

```
[name]  APL   [DSECT=YES]
```

Output: Register 15 contains one of the following return codes:

0 (X'00')	The partitions are (re)allocated.
4 (X'04')	The partitions are (re)allocated. But the size of the program area in at least one partition has been decreased because it does not leave the minimum GETVIS area or the minimum GETVIS area below 16MB. Display the new partition characteristics by a MAP command.
8 (X'08')	The allocation was rejected. The requested (rounded) allocation exceeds the corresponding allocation pool (increase RSIZE for real or PASIZE/SPSIZE/VSIZE for virtual allocation).
12 (X'0C')	The allocation was rejected. At least one specified (rounded) virtual partition allocation value is below the minimum of 128K.
16 (X'10')	The allocation was rejected. At least one partition would have a real, but no virtual allocation.
20 (X'14')	The allocation was rejected. At least one of the affected partitions is active or stopped, and the new virtual allocation would not include the old virtual boundaries, or the lower virtual boundary of the current partition would have to be moved upwards.
24 (X'18')	The allocation was rejected. The allocation would affect another active or stopped partition and therefore was stopped. The new real allocation would not include the old real boundaries.
28 (X'1C')	The virtual allocation was rejected. At least one of the specified partitions is already allocated in another virtual space.
32 (X'20')	The virtual allocation was rejected. There is not enough system getvis space for virtual storage available to allocate the page manager tables.
24 (X'18')	The real partitions are not reallocated. At least one of the affected partitions, other than the current, is active or stopped and the new real allocation would not include the old real boundaries.
28 (X'1C')	The partition are not reallocated. At least one of the specified partitions is already allocated in another space.
32 (X'20')	The (virtual or real) partitions are not reallocated. There is not enough System GETVIS space or virtual storage available to allocate the Page Manager tables.
36 (X'24')	The real partitions are not reallocated. For at least one of the specified partitions a PFIX Limit (BELOW) has been set.
40 (X'28')	The virtual partitions are not reallocated. For at least one of the specified partitions the minimum partition GETVIS area of 48KB below 16MB cannot be preserved.
44 (X'2C')	The virtual partition is not reallocated. It was tried to increase the initial allocation value of a partition that was allocated by using the default space id.

48 (X'30') The virtual partitions are not reallocated. It was tried to reallocate a partition by using defaults, but the space was created by specifying the space id explicitly or vice versa.

52 (X'34') The (virtual or real) partitions are not reallocated.
There is not enough real storage available to allocate the Page Manager tables.

56 (X'38') The real partitions are not reallocated. In a system without PDS the size of a real partition must not exceed the size of the corresponding virtual partition.
This was true for at least one of the specified real partitions.

ASYSCOM

The ASYSCOM macro returns the address of the system communication region to the user. The macro has the following format:

```
[name]   ASYSCOM   [(1)]
```

The operand specifies the general register that is to be loaded with the address of the System Communication Region (SYSCOM).

CLOSEHCF

The macro CLOSEHCF must be issued to terminate accessing of the HCF started by the POINTHCF macro.

The macro has the following format:

```
[name]   CLOSEHCF [{(hcfreg)|(1)}]
```

HCFREG Is the general register containing the address of the HCFCB

control block returned by the corresponding POINTHCF macro.

Note: If no operand is specified, the WRITE HCFCB will be closed. Only the HCF system task is allowed to close the WRITE HCFCB.

Output: Register 15 contains one of the following return codes:

0 (X'00')	Normal processing successfully completed.
4 (X'04')	Inconsistent input.

Register Usage: The contents of general register 14 through 2 are destroyed by this macro.

CPCOM

The format is as follows:

```
[name]   CPCOM   ACMD={name1|(r1)|(1)}
                ,LCMD={n|(r0)|(0)}
```

The operands have the following meaning:

ACMD Address of command text

LCMD Length of command in bytes, must be between 1 and 240

Output: Register 15 contains one of the following return codes.

0 (X'00')	Command successfully completed
1 (X'01')	Is returned if the supervisor is not running under VM (corresponds to CP completion code for 'Invalid Command').
2 (X'02')	Is returned if any parameter is invalid (corresponds to CP completion code for 'Invalid Parameter').

In all other cases, the CP completion code is returned unchanged in Register 15.

Cancel Conditions: The requester is canceled with 'Invalid SVC' (Error 21) if he is not authorized.

Register Usage:

Reg. 0	Length of command
Reg. 1	Address of command text
Reg.15	Return code

DEVREL

DEVREL decrements the physical usage counter and in addition resets the ownership for the specified partition as soon as the decremented physical unit counter reaches zero.

GETFLD PU=...,FIELD=OWNER can be used to obtain the current ownership status.

The macro can only be used by IPL and VSE/POWER and has the following format:

ASSEMBLER:

```
[name]   DEVREL    PU={name1|(r1)|(0)},PART={name2|(r2)|(1)}
```

PU Name of a 2-byte field or register containing the physical unit number of the device (same as PUB index = PUB offset/8).

PART Name of a 2-byte field or register containing the identifier (PIK) of the applicable partition. A value of 0 is interpreted as a request for system ownership.

Output: Register 15 contains one of the following return codes.

0 (X'00')	Request complete
4 (X'04')	SIO-count for JA must be saved
8 (X'08')	Device not owned by specified partition

Register Usage:

Reg. 0	Physical unit number for input
Reg. 1	PIK for input
Reg.15	Function code for input, return code for output

DEVUSE

DEVUSE increments the physical usage counter and sets ownership for the specified partition.

GETFLD PU=...,FIELD=OWNER can be used to obtain the current ownership status.

The macro can only be used by IPL and VSE/POWER and has the following format:

```
ASSEMBLER:
 [name]    DEVUSE     PU={name1|(r1)|(0)},PART={name2|(r2)|(1)}
```

PU Name of a 2-byte field or register containing the physical unit number of the device (same as PUB index = PUB offset/8).

PART Name of a 2-byte field or register containing the identifier (PIK) of the applicable partition. A value of 0 is interpreted as a request for system ownership.

Output: Register 15 contains one of the following return codes.

0 (X'00')	Request complete
8 (X'08')	Non-DASD device owned by other partition
12 (X'0C')	Device is down
16 (X'10')	A specified dynamic partition is not allocated.

Register Usage:

Reg. 0	Physical unit number for input
Reg. 1	PIK for input
Reg.15	Function code for input, return code for output

DSPLOG

The DSPLOG macro provides layouts for the various records on the LOG-DATA-SET. It describes the header as well as the detail records.

The macro has the following format:

```
[name]   DSPLOG
```

The macro has no operands.

DSPLPAR

The DSPLPAR macro provides layouts for the communication areas used between the LOG system task and the components issuing LOG requests.

The macro has the following format:

```
[name]    DSPLPAR
```

The macro has no operands.

DTSAPL

The DTSAPL macro generates or describes the layout of the Authorization Parameter List (APL).

The macro has the following format:

```
[name]    DTSAPL    [DSECT=YES]
```

For detailed description of the various input and output fields see DTSAPL macro expansion.

DTSJPL

The DTSJPL macro provides the layout of the Job Control Parameter List (JPL) build by JCL and for which storage was reserved at IPL time to allow access control.

When a job control ID-statement is detected, the access control information is extracted from the access control resource table (DTSECTAB) and transferred to the JPL.

The macro has the following format:

```
[name]    DTSJPL
```

For detailed description of the various fields see DTSJPL macro expansion.

EXTENT

This macro provides a DASD file protect interface. The facility supports all DASD (FBA and CKD) devices and can perform the following functions.

- An extent can be ADDED to the Extent Information for a LUB.
- An extent can be DELETED if a matching extent is found for this LUB.
- ALL extents for a given LUB can be DELETED.
- Given a LUB, it can CHECK for an already existing matching extent.

The macro has the following format:

```
[name]   EXTENT  {(1)|name1|DSECT=YES}
```

name1 Address of a parameter list as described by the DSECT. In case register notation was used, the register must contain the address of the parameter list.

Output: Register 15 contains one of the following return codes:

0 (X'00')	Request successfully processed.
4 (X'04')	The LUB is invalid.
8 (X'08')	No matching extent found (DELETE or CHECK).
12 (X'0C')	No more extent entries available (ADD).
16 (X'10')	Parameter list contains invalid data.

EXTRACT

The EXTRACT macro (see also “SVC 98 (X'62' - EXTRACT/MODCTB)” on page 277) provides

- Partition related information like
  - Partition boundaries from the storage management control block (SMCB)
  - Partition GETVIS information (Partition GETVIS and dynamic space GETVIS)
- Unit information from the PUB, the PUB extension or the PUB2 table entries
- System related information like
  - Control registers
  - CPU identifier
  - a list of partitions belonging to one address space
  - SVA related information

The macro uses the following keywords:

ID Identifies the requested information. Valid parameters are:

ATLCUU	Returns free CUU no for automatic tape library handling.
BDY	Returns system or partition boundary information.
CPUID	Returns CPUID and the partition prefix for SYSLOG.
CR	Returns the contents of control registers.
DEVICE	Returns device specific information as retrieved by means of the SENSE-ID command, together with the MDR, OBR record ID.
DVTY	Returns the device type code of specified device.
GETVIS	Returns actual GETVIS information.
MAP	Returns a list of partitions allocated in the specified space.
PART	Returns partition related values.
PUB	Returns the contents of PUB of specified device.
PUB2	Returns the contents of PUB2 of specified device.
SENSE	Returns the last sense information available.
SAINFO	Returns information for the stand alone supervisor environment.
SVA	Returns SVA related information.

MODE Qualifies the type of information that the requester wants to be returned in addition to the ID parameter. For possible values see the detailed description below.

DISP Specifies the offset within the specified field where EXTRACT is to start. The default value is zero. DISP may either be specified as a number or as a register containing the displacement value. If DISP is not a number, nor a register, it is assumed to be the name of a field defined in the MAPPUB DSECT.

SEL Points to a halfword containing the logical unit information in the same format as the logical unit number in the CCB.

SEP Points to a halfword containing a device address (cuu).

PU Points to a halfword containing a PUB index.

PID Points to a two-byte field containing the PIK of the partition the information belongs to. The default is the identifier of the partition issuing the request.

SID Address of a two-byte symbolic space identifier. Supported values for static spaces are 'R ', 'n ' (n = 1..3) and 'S ' (NOTE the blank as second letter), whereas for dynamic spaces it is the two character SYSLOG id of the space. If SID is omitted '0 ' is taken as a default value.

AREA Address of the user area where the extracted information is to be stored.

LEN Length of user area in bytes.

MFG Points to a work area where the parameter list is to be generated by the EXTRACT macro (for re-entrant coding). If register notation is used, register 1 may point to this area.

Input in General:

R0 Is a work register if S-type operands are used (for self-relocating programs)

R1 Is used as a pointer to a parameter list (PARMLIST) built by the EXTRACT macro prior to calling SVC 98 to process the request. The list has the following format.

Bytes	Description
0	Identification Field. EXTRACT values are: x'00' illegal. Results in illegal SVC. x'01' EXTRACT ID=PUB2 x'02' EXTRACT ID=DVTY x'03' EXTRACT ID=BDY x'04' EXTRACT ID=CR x'05' EXTRACT ID=PUB x'06' EXTRACT ID=CPUID x'07' EXTRACT ID=MAP x'08' EXTRACT ID=DEVICE x'09' EXTRACT ID=SENSE x'0A' EXTRACT ID=GETVIS x'0B' EXTRACT ID=PART x'0C' EXTRACT ID=SVA x'0D' EXTRACT ID=SAINFO x'0E' EXTRACT ID=ATLCUU MODCTB values are: x'F0' MODCTB ID=PUB2 x'F1' MODCTB ID=MNTLIST x'F2' MODCTB ID=MNTNEXT x'F3' MODCTB ID=ATLSETUP x'F4' MODCTB ID=KEKL
1	Flag byte. Its contents either reflects the MODE the user took to further specify the service, or it informs about the kind of device specification the caller took for device related services.
2 – 3	Reserved
4 – 5	Length of communication area as specified by the user.
6 – 7	Displacement in PUB2 or PUB table entry; retrieval of information begins at this point.
8 – 11	Address of communication area as specified by the user.
12 – 15	Address of two-bytes identifying the device. They either identify a logical unit in the CCB format (if 'SEL' was specified) or contain the physical device address (if 'SEP' was specified) or the PUB index (if 'PU' was specified).
16 – 19	Address of PIK.

Figure 1. Format of the SVC 98 (X'62) Parameter List

Output in General: The user's area is cleared to binary zeros and the information requested by ID is moved to the user's area in the specified length if not specified otherwise in the description of the specific service.

Reg.15 Contains one of the following return codes.

0 (X'00')	The requested information has been returned.
4 (X'04')	The partition specified is not supported by the system or the specified SID is not supported or invalid for the current system mode. In case the service should have returned GETVIS related information there are two possibilities. If the service also returns a control area begin address and this is returned as zero, GETVIS was not initialized. If it was not zero, it was initialized, but the service could not obtain the GETVIS gate. The service never waits for the gate to be opened again in order to avoid deadlocks.
8 (X'08')	The logical unit specified exceeds the range of the logical units for the specified partition or the specified PUB index is not within the range of PUBs valid for this system or the specified external device type code is not supported by VSE.
12 (X'0C')	The LUB is not assigned or ignored.
16 (X'10')	The length parameter is found to be zero, or negative, or below the minimum or too small to accommodate all output that could have been provided if the output is not provided in different "parts" as described by the mappings (see meaning of dotted lines in output mappings below) or the DISP specification exceeds the length of the PUB or PUB2 entry.
20 (X'14')	No general meaning, see specific service for meaning
24 (X'18')	No general meaning, see specific service for meaning
28 (X'1C')	The requested version of the service is not yet supported.

Output Mappings: Since the number of mapping macros for the EXTRACT service increases with every release of VSE/ESA, this new macro shall give a proper mapping for EXTRACT services in one.

The existing mapping macros will still be supported, but they will not be expanded any more. Any new mapping will be done using the new MAPEXTR macro. The general idea is as follows: If there is a macro invocation of the following type:

```
[name]  EXTRACT    ID=name1,MODE=name2,...
```

the following macro invocation shall provide the proper mapping for it:

```
[name]  MAPEXTR    ID=name1,MODE=name2[,DSECT=YES][,PREFIX=xyz]
```

The DSECT parameter determines, if the whole mapping is given in the form of a DSECT or as inline code.

The PREFIX parameter provides a possibility of changing the first three characters of the labelnames. If not specified, it defaults to 'MAP' in most cases. The PREFIX value must not be longer than three bytes. Having different mappings with the default PREFIX value may lead to assembly errors. This is because some mappings actually contain the same values from system control blocks, and the mappings should reflect this in the names. There is no mapping for that kind of output, which is simply a copy of supervisor control blocks already available for the user or a single simple value or field. These services are: EXTRACT ID=PUB (use MAPPUB), ID=PUB2, ID=DVTY, ID=SENSE, ID=CR, and ID=MAP.

The following mappings are provided by the MAPEXTR macro:

ID=ATLCUU
: Free CUU of device in automatic tape library

ID=BDY,MODE=SYSP
: system boundaries (MAPSYSP of today)

ID=BDY,MODE=P
: gives MAPBDYVR of today

ID=BDY,MODE=PERM
: information about permanent partition boundaries.

ID=BDY,MODE=T
: gives MAPBDY of today

ID=BDY,MODE=TEMP
: information about temporary partition boundaries

ID=GETVIS,MODE=[PT|SP|SV]
: GETVIS information

ID=CPUID
: cpuid of machine and SYSLOG ID

ID=CPUID,MODE=REAL
: mapping equal to previous

ID=PART
: partition panel information

ID=SAINFO
: Stand alone supervisor information

ID=SVA
: SVA panel information

ID=DEVICE
: device information as in MAPDEVIN of today. The PL/S invocation of MAPEXTR for ID(DEVICE) will only support the DSECT version of MAPDEVIN and take RECID(YES) as default (in contrast to MAPDEVIN, where RECID=NO is the default).

There also is a PL/S version available for all of the mappings which is given in the examples below. The PL/S version only produces DSECTS.

All information by EXTRACT is provided in parts. If the user specified a user area long enough to accommodate the next part of information, it will be provided. Only if the first part of information does not fit, a return code of 16 indicates an invalid length of output area. The parts of information, that are provided by a service are described by a dotted line in the mapping. A part must fit entirely to be provided.

Some services return GETVIS related information and this can either be 24-bit or 31-bit related. (See EXTRACT ID=GETVIS,LOC=.. or ID=PART or ID=SVA). In all of these cases the amount of used GETVIS comprises the control area length, if the control area was contained in the area that the information belongs to. Also the control area length value is related to the area being looked at. If the control area is

entirely in "high" GETVIS and the information is for BELOW, the control area length value will be zero. If it is for ANY, it will be the complete length. All GETVIS information that is for 31-bit addressable GETVIS always includes the 24-bit areas as well. There is no specific "high" GETVIS information, that would only reflect the GETVIS area above the 16MB line. This does also mean, that the largest contiguous free space for ANY type GETVIS may partly or even completely be located below the 16MB line. If there is no GETVIS area above the 16MB line, the ANY type information will be similar to the BELOW type information.

Addressing Mode Considerations: The EXTRACT macro can be called from below and above the 16MB line. The parameter list and the area to be filled with the return information may be located below or above as well. If the values in the parameter list are interpreted as 24-bit or 31-bit pointers depends on the AMODE, that the caller was in when he issued the SVC generated by the EXTRACT macro. If he was in AMODE 24, all pointers to parameter list, outputarea, other values, will be taken as 24-bit pointers. If he was in AMODE 31 they will be taken as 31-bit addresses. The service does not support access register mode and must not be called in it (Caller will be cancelled).

Cancel Conditions: An illegal SVC is forced, if one or more of the following conditions are true:

- ID specification is invalid.
- An input parameter normally restricted to a key zero user was specified by a non-key zero user.

  An addressing mode violation will be forced by calling the service in access register mode.

Extracting Device-Related Information:

Invocation:

```
[name]    EXTRACT    ID={DEVICE|DVTY|PUB|PUB2|SENSE}
                     ,AREA={name1|(S,name1)|(r1)}
                     ,LEN={n|(r2)}
                     ,{SEL={name3|(S,name3)|(r3)}
                      [,PID={name6|(S,name6)|(r6)}]|
                       SEP={name4|(S,name4)|(r4)}|
                       PU={name5|(S,name5)|( r5)}}
                     [,DISP={0|n|name8|(r8)}]
                     [,MFG={name9|(r9)}]

[name]    EXTRACT    ID=SAINFO
                     ,AREA={name1|(S,name1)|(r1)}
                     ,LEN={n|(r2)}
                     [,MODE={PUBDEVTY|EXTCODE}]
                     [,MFG={name9|(r9)}]

[name]    EXTRACT    ID=ATLCUU
                     ,AREA={name1|(S,name1)|(r1)}
                     ,LEN={n|(r2)}
                     [,MFG={name9|(r9)}]
```

Parameters for different IDs:

ID	AREA	LEN	DISP	PID/SEL	SEP	PU	MFG
DEVICE	R	R	N/A	CX	CX	CX	O
DVTY	R	R	N/A	BX	BX	BX	O
PUB	R	R	O	BX	BX	BX	O
PUB2	R	R	O	AX	AX	AX	O
SENSE	R	R	N/A	CX	CX	CX	O

Explanation of abbreviations:

R Means "required". The parameter may not be omitted.

O The parameter is optional.

AX For all device related information a unique identification of the device in question must be given. Hence specification of SEP, PU or SEL (with or without PID) is mutually exclusive. Specification of a PIK of another partition than the callers partition or specification of a PU or a SEP parameter is only accepted from key zero users here.

BX In these cases a PID of a partition other than the callers partition may be issued without running in key zero state. PU and SEP still require key zero.

CX All kind of device identification (including 'foreign' PIK values) are accepted from a non-key-zero user.

N/A Not available or not accepted.

Output produced by these device related invocations:

ID=PUB The appropriate PUB bytes are copied to the user area. The layout of a PUB entry is described by the DSECT macro MAPPUB. If the logical unit specified by EXTRACT ID=PUB is unassigned or assigned ignore, a return code 12 is presented in Register 15 and, in addition, an indicator is stored in byte 0 of the area specified by the AREA operand. The indicator is X'FF' if the logical unit is unassigned, and X'FE' if it is assigned ignore. The minimum area length is 1 byte.

ID=PUB2 The PUB2 table entry is moved to the user area, either in the specified range, or in its complete length.

ID=DEVICE A length of at least 10 bytes is required. Device specific information is to be returned. The output is described by the mapping macro MAPDEVIN. The format is as follows:

```
[name]  MAPEXTR   ID=DEVICE[,DSECT=YES][,PREFIX=xyz][,RECID=NO|YES]
```

If no name is specified the default name xyzDVINF is generated. If no PREFIX value is specified it defaults to 'IJB' to give the same names as former MAPDEVIN macro did.

DEC	HEX	Label	Description
0 – 1	0 – 1	xyzDVCUU	CUU address
2	2	xyzDVDTC	VSE device type code
3 – 9	3 – 9	xyzDVSNS	Sense device type information
3	3	xyzDVSNV	Validity flag
		xyzDVVAL	X'FF' Entry is valid
4 – 5	4 – 5	xyzDVCUN	Control unit type number
6	6	xyzDVCUM	Control unit model number
7 – 8	7 – 8	xyzDVDTN	Device type number
9	9	xyzDVDTM	Device type model number
10 – 11	A – B	xyzDVOBR	OBR – OS device code
12	C	xyzDVMDR	MDR device code
10	A	xyzDVLEN	Length of mapping area with RECID=NO
13	D	xyzDVLEN	Length of mapping area with RECID=YES

Figure 2. Output as Described by MAPEXTR ID=DEVICE

ID=SENSE Stores as many sense bytes into the user area as is permitted by the LEN parameter. The rest of the AREA (if any) is filled up with binary zeroes.

ID=SAINFO For this service the area addressed by the AREA= parameter serves as *input and output* area. The service is intended to relate external device type codes to pub device type codes and vice versa. The area is mapped by the following DSECT which is named xyzSAINF with xyz being chosen by the PREFIX= parameter of the MAPEXTR macro.

DEC	HEX	Label	Description
0 – 3	0 – 3	xyzFNDNO	no of found entries
4 – 7	4 – 7	xyzFLAGS	flag word (currently all set to zero)
8 – 13	8 – D	xyzEXTCD	field for external device type code
14	E	xyzPDVTY	field for pub device type code
15	F	xyzPMODE	field for default mode of device
8	8	xyzENTLN	Length of one entry in list

Figure 3. Input and Output Area as Described by MAPEXTR ID=SAINFO

If MODE=PUBDEVTY was specified the service expects the xyzPDVTY field of the first entry to be set to some device type code. It will then scan the supervisor internal table of supported device types for all entries that share this specified PUB device type code and append eight byte entries to the list for each one it finds. Each entry will contain the EXTERNAL device type code of the device in readable character format. The field xyzPMODE of each entry will be set to the default mode setting used for this device. The field xyzFNDNO will be set to the number of entries found. If the number of entries does not fit into the area length the caller specified in the LEN= operand of the macro invocation, a return code of x10 (decimal 16) is set to indicate the list is incomplete.
If the caller specified MODE=EXTCODE, the service expects the xyzEXTCD field of the first entry to contain an external device type code in readable character format, left adjusted and padded with blanks to the end of the six byte field. It will then scan the table of supported devices and insert the devices pub device type code in the field xyzPDVTY of the entry if it finds it. If the external device type code could not be found, a return code of 8 is set.
The xyzFLAGS field must be set to zero when calling the service. It is intended for future expansions. The service is intended for use in the stand alone supervisor environment but does not check for any authorization of a caller.

ID=ATLCUU This service was installed to supply programs that handle automatic tape libraries with a CUU number of a device in such a library which is still free for mount operations. This service also uses the area addressed by the AREA= operand of the macro invocation as an input and output area. The area is mapped by the xyzATLCU DSECT generated by the MAPEXTR macro.

DEC	HEX	Label	Description
0 – 3	0 – 3	xyzATVER	version identification (currently zero)
4 – 5	4 – 5	xyzATCUU	field for CUU value (set by service)
6 – 11	6 – B	xyzATDVT	field for external device type code
12 – 19	C – 13	xyzATLNM	field for name of automat. tape library
16	10	xyzATLLN	Length of one entry in list

Figure 4. Input and Output Area as Described by MAPEXTR ID=ATLCUU

The service expects the caller to supply the external device type code of the kind of tape he requires to be set in the field xyzATDVT and the name of the library in field xyzATLNM. Both fields are in readable character format, left adjusted, padded with trailing blanks. The service then scans the system device tables for a device that matches the device type and library name and returns its CUU in the xyzATCUU field. In addition it issues a temporary mount reserve for the device. At the moment only one CUU entry is returned. The version identifi- cation of the service should be set to zero to allow for future extensions of the service. If it is not set to zero the service returns with a return code of x1c (decimal 28) to indicate unsup- ported function.
If no free device could be found a return code of x14 (decimal 20) is given. If the external device type code given by the caller is not supported for library devices, a return code of 8 is set. Currently the only supported values are "3490E " and "TPA " .

Extracting System-Related Information

Invocation:

```
[name]   EXTRACT    ID={CPUID|CR}
                    ,AREA={name1|(S,name1)|(r1)}
                    ,LEN={n|(r2)}
                    [,MFG={name9|(r9)}]

[name]   EXTRACT    ID=BDY,MODE=SYSP
                    ,AREA={name1|(S,name1)|(r1)}
                    ,LEN={n|(r2)}
                    [,PID={name6|(S,name6)|(r6)}]
                    [,MFG={name9|(r9)}]

[name]   EXTRACT    ID=MAP
                    ,AREA={name1|(S,name1)|(r1)}
                    ,LEN={n|(r2)}
                    [,SID={name7|(S,name7)|(r7)}]
                    [,MFG={name9|(r9)}]

[name]   EXTRACT    ID=SVA
                    ,AREA={name1|(S,name1)|(r1)}
                    ,LEN={n|(r2)}
                    [,MFG={name9|(r9)}]
```

Output produced by system related services:

ID=CR A length of 16*4 bytes is required for the content of the control registers. They are stored in the sequence 0 to 15.

ID=CPUID A length of 10 bytes is required. The area must start on a doubleword boundary. On return AREA will contain the CPUID as stored by the CPU, followed by the SYSLOG ID of the issuing partition. The SYSLOG ID is a two character identification of the partition (like: BG, F2 or LM) which is identical to the space id in case of dynamic partitions.

ID=BDY The MODE=SYSP version of the EXTRACT ID=BDY gives system boundary information. It was introduced as a substitute of the former MODE=S part. Its output is mapped by the following MAPEXTR macro invocation:

```
[name]  MAPEXTR   ID=BDY,MODE=SYSP[,DSECT=YES][,PREFIX=xyz]
```

Or in PL/S

```
?[name:] MAPEXTR  ID(BDY) MODE(SYSP) PREFIX(xyz);
```

The PREFIX parameter defaults to MAP, the optional labelname of the macro defaults to xyzSYSP. The mapping will look as follows:

DEC	HEX	Label	Description
0 – 3	0 – 3	xyzSUPBG	1ST BYTE OF SUPERVISOR
4 – 7	4 – 7	xyzSUPND	LAST BYTE OF SUPERVISOR
8 – 11	8 – b	xyzSSBEG	1ST BYTE OF SDAID AREA
12 – 15	c – f	xyzSSEND	LAST BYTE OF SDAID AREA
16 – 19	10 – 13	xyzSVA	1ST BYTE OF SVA
20 – 23	14 – 17	xyzESVA	LAST BYTE OF SVA
24 – 27	18 – 1b	xyzSPBEG	1ST BYTE OF SHARED PART. AREA
28 – 31	1c – 1f	xyzSPEND	LAST BYTE OF SHARED PART.AREA
32 – 35	20 – 23	xyzPRPBG	1ST BYTE OF PRIV. PART. AREA
36 – 39	24 – 27	xyzPRPND	LAST BYTE OF PRIV. PART. AREA
40 – 43	28 – 2b	xyzVPBEG	1ST BYTE OF VPOOL AREA
44 – 47	2c – 2f	xyzVPEND	LAST BYTE OF VPOOL AREA
48 – 51	30 – 33	xyzSVIS	1ST BYTE OF SYSTEM GETVIS AR.
52 – 55	34 – 37	xyzSVISE	LAST BYTE OF SYST. GETVIS AR.
56 – 59	38 – 3b	xyzPPBEG	1ST BYTE OF PROBL.PROGR. AREA
.....	.....	.....	.....
60 – 63	3c – 3f	xyzSVA3B	1ST BYTE OF 31 BIT SVA
64 – 67	40 – 43	xyzSVA3E	LAST BYTE OF 31 BIT SVA
68 – 71	44 – 47	xyzSGV3B	1ST BYTE OF 31 BIT SYST.GETVIS
72 – 75	48 – 4b	xyzSGV3E	LAST BYTE OF 31 BIT SYST.GETVIS
76 – 79	4c – 4f	xyzVLB3B	1ST BYTE OF 31 BIT VIRT.LIBRARY
80 – 83	50 – 53	xyzVLB3E	LAST BYTE OF 31 BIT VIRT.LIBRARY
84 – 87	54 – 57	xyzVSIZE	TOTAL VIRT.STOR.SIZE IN K–BYTE
88 – 91	58 – 5b	xyzSYSVS	TOTAL VIRTUAL STOR.SIZE OF SYSTEM USED ADDR. SPACES (K–BYTE)
92 – 95	5c – 5f	xyzSPFLL	SYSTEM PFIX LIMIT IN LOW AREA
96 – 99	60 – 63	xyzSPFCL	SYSTEM PFIX COUNT IN LOW AREA
100 –103	64 – 67	xyzSPFL3	SYSTEM PFIX LIMIT IN HIGH AREA
104 –107	68 – 6b	xyzSPFC3	SYSTEM PFIX COUNT IN HIGH AREA
108	6c	xyzSYLEN	Length of mapping (EQUate)

Figure 5. Output of service EXTRACT ID=BDY,MODE=SYSP

Boundaries for non existent areas show a 0 as begin and -1 as end address.

ID=MAP A list of two-byte PIK values of partitions allocated in the specified space will be returned. The list is ordered by increasing partition begin address. The ALLOCATE algorithm makes sure that all partitions are contiguous. The end of the PIK list is indicated by zero. A length of 32 bytes is required in case of a static space, 4 bytes (or more) are accepted for a dynamic space. For a dynamic space the list contains only one PIK entry.

ID=SVA A number of boundaries and GETVIS information about the shared virtual areas of the system are supplied. For Assembler Programs the invocation is as follows:

```
[name]  MAPEXTR   ID=SVA[,DSECT=YES][,PREFIX=xyz]
```

The invocation of MAPEXTR in PL/S programs:

```
?[name:] MAPEXTR  ID(SVA) PREFIX(xyz);
```

The PL/S invocation always generates a DSECT, the Assembler invocation either produces a DSECT or inline code, giving it either the user supplied name or the default name xyzSVA. For compatibility reasons, the PREFIX parameter will here default to 'XTR' instead of 'MAP'.

DEC	Label	Description
0–3	xyzSVAB	BEGIN ADDRESS OF SVA LOW
4–7	xyzSVAE	END OF SVA LOW
8–11	xyzVLIBB	1ST BYTE OF VIRT. LIBRARY IN LOW SVA
12–15	xyzVLIBF	AMOUNT OF FREE SPACE IN THIS VIRT.LIBRARY
16–19	xyzVLIBE	LST BYTE OF VIRTUAL LIBRARY
20–23	xyzSVISC	START OF SYSTEM GETVIS CONTROL
24–27	xyzSVIS	START OF ALLOCATABLE AREA OF GETVIS
28–31	xyzSVISE	END OF SYSTEM GETVIS AREA
		SYSTEM GETVIS LOC=BELOW INFORMATION
32–35	xyzSVGCL	LENGTH OF GETVIS CONTROL AREA
36–39	xyzSVGHW	LARGEST AMOUNT USED UP UNTIL NOW
40–43	xyzSVGFR	TOTAL AMOUNT OF FREE GETVIS SPACE
44–47	xyzSVGUS	TOTAL AMOUNT OF USED GETVIS SPACE
48–51	xyzSVGMF	LARGEST CONTIGUOUS FREE GETVIS SPACE
52–55	xyzSVGRS	NUMBER OF RESERVED PAGES IN GETVIS
56–59	xyzSLAB	START OF LABEL AREA
60–63	xyzSLAE	END OF LABEL AREA
64–67	xyzVPOLB	START OF VPOOL
68–71	xyzVPOLE	END OF VPOOL
72–75	xyzSVA3B	1ST BYTE IN SVA HIGH
76–79	xyzSVA3E	LST BYTE IN SVA HIGH
80–83	xyzSGV3B	1ST BYTE OF SYSTEM GETV. IN HIGH SVA
84–87	xyzSGV3E	LST BYTE OF SYSTEM GETV. IN HIGH SVA
88–91	xyzVLB3B	1ST BYTE OF VIRT. LIBRARY IN SVA HIGH
92–95	xyzVLB3E	LST BYTE OF THIS VIRTUAL LIBRARY
96–99	xyzVLB3F	AMOUNT OF FREE SPACE IN THIS VIRT.LIBRARY
		SYSTEM GETVIS LOC=ANY INFORMATION
100–103	xyzSAGCL	LENGTH OF GETVIS CONTROL AREA
104–107	xyzSAGHW	LARGEST AMOUNT USED UP UNTIL NOW
108–111	xyzSAGFR	TOTAL AMOUNT OF FREE GETVIS SPACE
112–115	xyzSAGUS	TOTAL AMOUNT OF USED GETVIS SPACE
116–119	xyzSAGMF	LARGEST CONTIGUOUS FREE GETVIS SPACE
120	xyzSVALN	Length of mapping (EQUate)

Figure 6. Output of service EXTRACT ID=SVA

For a possible return code of 4 see return code explanations above.

Extracting GETVIS-Related Information

Invocation:

```
[name]   EXTRACT    ID={GETVIS}
                    ,AREA={name1|(S,name1)|(r1)}
                    ,LEN={n|(r2)}
                    [,PID={name6|(S,name6)|(r6)}]
                    [,MODE={PT|SP|SV}]
                    [,LOC={RES|BELOW|ANY}]
                    [,MFG={name9|(r9)}]
```

ID=GETVIS is intended to extract information about the partition GETVIS area, the dynamic space GETVIS area or about the system GETVIS area in the SVA. Which GETVIS area is addressed is told either by specifying

MODE=SP (dynamic space GETVIS)

MODE=PT (partition GETVIS)

MODE=SV (SVA GETVIS)

or by the PID parameter. Omission of the parameter will result in partition GETVIS information for the issuing partition. This way applications running in ICCF interactive partitions may obtain information about their own partition GETVIS area. Setting it to binary zeroes will return the same information as MODE=SV. The LOC= parameter has the same meaning as in the GETVIS macro itself.

For the LOC parameter being

RES — the residency of the caller determines whether 31-bit addressable GETVIS or 24-bit addressable GETVIS will be taken for calculation of the GETVIS information. Like with the GETVIS macro itself, this is the default assumed, if no LOC= was specified.

BELOW — only 24-bit addressable GETVIS will be taken for the retrieval of the GETVIS information.

ANY — 31-bit addressable GETVIS will be taken for the retrieval of the GETVIS information.

The 31-bit GETVIS information always includes the values for 24-bit. For partition GETVIS for example, the maximum contiguous free space that is returned by EXTRACT ID=GETVIS,LOC=ANY may partly or totally be located below the 16MB line. For system GETVIS it may be in either the low or the high SVA, but not in both since the two areas are not adjacent. The amount of free ANY GETVIS is the sum of free GETVIS below and above the 16MB line. This logic is similar to the logic the GETVIS macro uses to satisfy a GETVIS LOC=ANY request. For a partition that has no GETVIS area above the 16MB line, the results of LOC=ANY will always be similar to the results of LOC=BELOW. For a partition with GETVIS above 16MB, the control area length as provided by EXTRACT LOC=BELOW will be zero, since the control area is above the 16MB line. The control area also counts as used part of GETVIS.

As described before, the MAPEXTR mapping macro will be provided for a mapping of the output.

```
[name]  MAPEXTR  ID=GETVIS,MODE=PT[,DSECT=YES][,PREFIX=xyz]
```

Or in PL/S

```
?[name:] MAPEXTR  ID(GETVIS) MODE(PT) PREFIX(xyz);
```

The PREFIX parameter defaults to MAP, the optional labelname of the macro defaults to xyzPTG for MODE=PT, xyzDYG for MODE=SP and xyzSVG for MODE=SV.

DEC	HEX	Label	Description
0 – 3	0 – 3	xyzPTGCL	length of GETVIS control area
4 – 7	4 – 7	xyzPTGHW	high water mark, max. ever used.
8 – 11	8 – b	xyzPTGFR	current free GETVIS space
12 – 15	c – f	xyzPTGUS	current used GETVIS space
16 – 19	10 – 13	xyzPTGMF	maximum contiguous free space
20	14		Length of mapping

Figure 7. Mapping given by MAPEXTR ID=GETVIS,MODE=PT

For MODE=SV the mapping will be the same, with the only exception that the names contain ...SVG.. instead. Likewise it is ...DYG.. for dynamic space GETVIS with MODE=SP.

Extracting Partition-Related Information

```
[name]   EXTRACT    ID={BDY}
                    ,AREA={name1|(S,name1)|(r1)}
                    ,LEN={n|(r2)}
                    [,PID={name6|(S,name6)|(r6)}]
                    [,MODE={P|PERM|TEMP|I}]
                    [,MFG={name9|(r9)}]

[name]   EXTRACT    ID={PART}
                    ,AREA={name1|(S,name1)|(r1)}
                    ,LEN={n|(r2)}
                    [,PID={name6|(S,name6)|(r6)}]
                    [,MFG={name9|(r9)}]
```

Output as produced by the boundary service:

MODE=P This service is obsolete with 1.3.0 and only supported for compatibility reasons. MODE=PERM should be used instead.

MODE=PERM This new service replaces the old EXTRACT ID=BDY,MODE=P which will remain supported for compatibility reasons. In contrast to the old MODE=P, it will no more accept a 0 as an indication to obtain values from the shared virtual area. Use MODE=SYSP for this instead. The output mapping is provided by the MAPEXTR invocation:

```
[name]  MAPEXTR    ID=BDY,MODE=PERM[,DSECT=YES][,PREFIX=xyz]
```

Or in PL/S

```
?[name:] MAPEXTR  ID(BDY) MODE(PERM) PREFIX(xyz);
```

The PREFIX parameter defaults to MAP, the optional labelname of the macro defaults to xyzBDYPM. The mapping will look as follows:

DEC	HEX	Label	Description
0 – 3	0 – 3	xyzVPBEG	1st addressable byte of partition
4 – 7	4 – 7	xyzVPPPE	last addr. byte of program area in part.
8 – 11	8 – B	xyzVPEND	last addr. byte of part., incl.GETVIS
12 – 15	C – F	xyzRPBEG	1st byte of real partition
16 – 19	10 – 13	xyzRPEND	last byte of real partition
.....	.....	.....	.....
20 – 23	14 – 17	xyzPFXLL	PFIX limit for storage below 16MB line
24 – 27	18 – 1b	xyzPFXL3	PFIX limit for storage above 16MB line
28	1c	xyzBDYPL	Length of MAPBDY area.

Figure 8. Output of Service EXTRACT ID=BDY,MODE=PERM

The values thus given are all permanent boundaries and need not be the actual boundaries of the partition. These are supplied by the MODE=TEMP invocation. The first five values are similar to the values supplied by the old MODE=P invocation (for compatibility reasons). The new two values are the newly supplied permanent PFIX limits for the partition.

MODE=T This service is obsolete with 1.3.0 and only supported for compatibility reasons. Use MODE=TEMP instead. However it is still the default if no mode operand was specified. The information it returns is identical to the first five fullwords of the information returned by the new service.

MODE=TEMP indicates that the temporary boundaries of the issuing partition are to be returned. This is also the default value. PID may not be specified in this case, since a snapshot of any other partition's temporary boundaries is unreliable.

```
[name]  MAPEXTR    ID=BDY,MODE=TEMP[,DSECT=YES][,PREFIX=xyz]
```

Or in PL/S

```
?[name:] MAPEXTR  ID(BDY) MODE(TEMP) PREFIX(xyz);
```

The PREFIX parameter defaults to MAP, the optional labelname of the macro defaults to xyzBDYTM. The mapping will look as follows:

DEC	HEX	Label	Description
0 – 3	0 – 3	xyzPTBEG	1st addressable byte of partition
4 – 7	4 – 7	xyzPTPPE	lst addr. byte of program area in part.
8 – 11	8 – B	xyzPTEND	lst addr. byte of partition, incl.GETVIS
12 – 15	C – F	xyzPFXLL	PFIX limit (KB or zero (real mode)
16 – 19	10 – 13	xyzPFXCL	PFIX count (number of PFIXed pages).
.....	.....	.....	.....
20 – 23	14 – 17	xyzPFXL3	PFIX limit for storage above 16MB (KB)
24 – 27	18 – 1b	xyzPFXC3	PFIX count for this storage (pages)
28	1c	xyzBDYTL	Length of MAPBDY area.

Figure 9. Output of Service EXTRACT ID=BDY,MODE=TEMP

The first five values are identical to the values provided by the old service ID=BDY,MODE=T. This is intended to avoid unnecessary recompiles for old programs. The data can be obtained by the new service ID=BDY,MODE=TEMP, but may still be interpreted by programs that use MAPBDY. The value xyzPFXL3 is the limit value provided by the new JOB CONTROL command SETPFIX for the storage above the 16MB line, xyzPFXLL is the same value below 16MB. The limit values are the current values.

If the partition is executing in real mode the boundaries of the real partition will be returned, that means:

xyzPTBEG defines the begin address of the real partition (Problem Program Save Area).

xyzPTPPE defines the logical end of the real partition which is identical with the allocated (or PFIXed) partition end address (if no SIZE was specified in the EXEC statement), and which is the real GETVIS area begin address (if SIZE was specified).

xyzPTEND is the allocated (or PFIXed) real partition end address (if no SIZE was specified in the EXEC statement), and it is the highest *used* GETVIS area address (rounded to the next page boundary) (if SIZE was specified). xyzPTPPE and xyzPTEND are equal if no SIZE was specified in the EXEC statement, or if no GETVIS request has been issued so far.

Output as produced by the PART service:

This service retrieves partition related data for display by the interactive interface II. Frequent use of the service may have a degrading impact on the performance of the partition being asked for, since the partition is not permitted to do any kind of GETVIS operation while this service is in execution.

The mapping is again provided by an invocation of MAPEXTR. Assembler programs code:

```
[name]   MAPEXTR  ID=PART[,DSECT=YES][,PREFIX=xyz]
```

The invocation of MAPEXTR in PL/S programs:

```
?[name:] MAPEXTR  ID(PART) PREFIX(xyz);
```

DEC	Label	Description
0–3	xyzPTGVC	START OF PART.GETVIS CONTR.AREA
4–7	xyzDYGVC	START OF DYN.SPACE GETVIS CONT.AREA
8–11	xyzPTBEG	CURRENT 1ST BYTE OF PARTITION
12–15	xyzPTHPH	HIGHEST PHASE LOAD ADDRESS
16–23	xyzPTNAM	NAME OF JOB IN PARTITION
24–27	xyzPTPPE	PROBLEM PROGRAM END ADDRESS
28–31	xyzPTEND	CURRENT LST BYTE OF PARTITION
32–35	xyzDYGVB	1ST BYTE OF DYN.SPACE GETVIS AREA
36–39	xyzDYGVE	LST BYTE OF DYN.SPACE GETVIS AREA
40–42		RESERVED
43	xyzPTPIB	PIBFLAG FROM PIB
		PARTITION GETVIS LOC=BELOW
44–47	xyzPTGCL	LENGTH OF PART.GETVIS CONTROL AREA
48–51	xyzPTGHW	LARGEST AMOUNT USED UP UNTIL NOW
52–55	xyzPTGFR	TOTAL AMOUNT OF FREE GETVIS SPACE
56–59	xyzPTGUS	TOTAL AMOUNT OF USED GETVIS SPACE
60–63	xyzPTGMF	LARGEST CONTIGUOUS FREE GETVIS SPACE
		DYNAMIC SPACE GETVIS INFORMATION
64–67	xyzDYGCL	LENGTH OF SPACE GETV. CONTROL AREA
68–71	xyzDYGHW	LARGEST AMOUNT USED UP UNTIL NOW
72–75	xyzDYGFR	TOTAL AMOUNT OF FREE GETVIS SPACE
76–79	xyzDYGUS	TOTAL AMOUNT OF USED GETVIS SPACE
80–83	xyzDYGMF	LARGEST CONTIGUOUS FREE GETVIS SPACE
.....	.....	.....
84–87	xyzVPBEG	1ST ADDRESSABLE BYTE OF VIRTUAL PARTITION
88–91	xyzVPEND	LST ADDR.BYTE OF VIRTUAL PARTITION
92–95	xyzRPBEG	1ST ADDR.BYTE OF REAL PARTITION
96–100	xyzRPEND	LST ADDR.BYTE OF REAL PARTITION
		PART. GETVIS WITH LOC=ANY
100–103	xyzPAGCL	LENGTH OF PARTITION GETVIS CONTROL AREA
104–107	xyzPAGHW	LARGEST AMOUNT USED UP UNTIL NOW
108–111	xyzPAGFR	TOTAL AMOUNT OF FREE GETVIS SPACE
112–115	xyzPAGUS	TOTAL AMOUNT OF USED GETVIS SPACE
116–119	xyzPAGMF	LARGEST CONTIGUOUS FREE GETVIS SPACE
120	xyzPARTL	LENGTH OF MAPPING (EQUATE)

Figure 10. Output of Service EXTRACT ID=PART

FREECBUF

The FREECBUF returns a Copy Buffer back to the supervisor. It is used by TAPE-ERP and has the following format.

```
[name]   FREECBUF  {name1|(1)}
```

name1 Address of copy buffer that is to be returned to the supervisor.

GETCBUF

The GETCBUF macro returns the address of a 72-byte area (one Copy Buffer) to the requester. It is used by the TAPE-ERP and has the following format.

```
[name]   GETCBUF
```

Output: The address of a Copy Buffer (72-byte area in supervisor) is returned in register 1. If none is available, zero is returned.

GETDADR

The GETDADR macro returns the virtual address of an I/O area pointed to by a CCW.

The macro is issued by system tasks, for example, ERP, RAS and it has the following format:

```
[name]   GETDADR  {ccwaddr|(8)},{offset|(0)}
```

ccwaddr Pointer to a CCW containing the address of the I/O area. If the CCW address is a real address, the issuing task will be canceled.

offset Offset within the I/O area, which is to be added to the I/O area before translation to virtual takes place.

Output: Register 15 contains the converted (virtual) address if the page frame indicated by the real address is connected to a virtual page. If the address of the CCW is invalid, zero is returned.

GETFLD

The GETFLD macro retrieves system, partition, task or device specific information. It can be called in any AMODE or RMODE. Access register mode will not have any effect on the processing of the request.

```
[name]   GETFLD FIELD=name1,
                [,PU={name2|(r2)|(0)}]
                [,LU={name3|(r3)|(0)}]
                [,PART={name4|(r4)|(0)}]
                [,TASK={name5|(r5)|(0)}]
                [,LOGID={name6|(r6)|(0)}]
                [,ADDR={name7|(r7)|(0)}]
                [,ALET={name8|(r8)|(0)}]
                [,CUU={name9|(r9)|(0)}]
                [,BRANCH={YES|NO}]
                [,RETURN={YES|NO}]
```

FIELD= Symbolic identification of the information to be retrieved. Valid symbols, and their interpretation see below.

PART= Name of a halfword or a register containing the PIK of the partition to which the specified information belongs.

TASK= Name of a halfword or a register containing the TID of the task to which the specified information belongs.

PU= Name of a halfword or register containing the physical unit number (PUB index) of a device, for which information shall be retrieved.

LU= Name of a halfword or a register containing a logical unit.

Note: Format as in CCB.

LOGID= Name of a halfword or a register containing the SYSLOG ID of the partition to which the required information belongs.

ADDR= Name of a fullword or a register containing the address used in combination with the service indicated in the field value.

ALET= Name of a fullword or a register containing the access list entry token used in combination with the service indicated in the field value.

BRANCH= Either set to YES or NO. Default is NO. If BRANCH=YES is specified, then instead of a SVC a branch into the supervisor service is performed. The branch interface is executed in parallel mode and is therefore preferrable for multi processor environments. BRANCH=YES can only be specified for a specific set of GETFLD functions and only if information is requested for the current task. The branch enabled functions have a comment in 'See notes:'. Register conventions for using the branch interface are as follows:

```
 R0: is used by GETFLD macro as in BRANCH=NO
 R1: is used by GETFLD macro as in BRANCH=NO
 R2: is used by GETFLD macro as in BRANCH=NO
R13: must point to a 76 byte (standard 18 word) save area, provided by
     the caller of the GETFLD macro
R15: is used to branch to the service, on output as in BRANCH=NO
```

RETURN= Either set to YES or NO. Default is NO. If RETURN=YES is specified false input specifications will not lead to a cancel but to a return code. The return code indicates, what input parameter was wrong. The following values are possible now (and there may be more in the future!!).

x'80000004' The specified GETFLD FIELD= value is not supported.

x'80000008' The caller is not authorized for this type of request.

x'8000000C' The value supplied by the PART= parameter was unusable.

x'80000010' The value supplied by the TASK= parameter was unusable.

x'80000014' The value supplied in the PU= parameter was unusable.

x'8000002C' A FIELD=LOGBAL invocation specified a record nr not equal to zero.

General Remarks on Register Usage: In all cases of an error being indicated by the value in R15, the content of the output register R0, R1 or both will be cleared to zero. Only in those cases that are explicitly mentioned in the description of the field values above the content of R0 and R1 is meaningful after the call.
The following can be selected by the FIELD parameter.

ABEXIT (ABEXIT1)

Pointer to standard AB exit routine and save area.
Supplied input: TASK ID of current task or 0.
Values returned:
R0 points to AB exit routine. If Bit zero in R0 is
on the exit routine is currently active.
R1 points to the save area. Both pointers are zero, if no exit is defined.
No interface is available to retrieve the addresses of the exit associated with OPTION=EARLY.
R2 is only modified if ABEXIT1 was specified. It will then contain the following bit settings.

BIT 31 is set if OPTION=NODUMP was specified for this exit.

BIT 30 is set if OPTION=EARLY was specified for this exit.

BIT 29 is set if OPTION=MSGONLY was specified for this exit.

BIT 28 is set if AMODE ANY was specified for this exit.

OPTION=DUMP was specified if bits 29 and 31 are both off.
See notes: NOFUTRC (for ABEXIT only), NOAUTHR, NOTID0, CURRTID, BRANCH.

ABINPR Abnormal exit in process flag
Supplied input: TASK ID.
Values returned: x'01' in R1 if the AB exit routine is active for the specified task, else 0 (No fast path service for TASK IDs of other task than current).
See notes: NORC, NOAUTHC, NOTID0.
Intended users: ICCF

ACEEPTR

Get accessor environment element
Supplied input: none.
Values returned: if R15 is:

0 R1 points to current ACEE, R0 is one of the following:

```
1   ACEE FOUND AT TASK
2   ACEE FOUND AT PARTITION
4   ACEE FOUND AT TASK JPL
8   ACEE FOUND AT PARTITION JPL
256 DTSECTAB UID AT PARTITION JPL FOUND
1
```

8 R1 and R0 are 0.

See notes: BRANCH.
Authorized: SYSTEM, SUBSYSTEMS, JOB CONTROL, Vendors.

ACLOSE VSAM automatic close
Supplied input: nothing.
Values returned: x'01' in R1 if VSAM Automatic Close is currently in process, else 0.
See notes: NORC

ALET Access list entry token
Supplied input: PART with PIK of partition that shall be accessed.
Values returned: if R15 is:

0 R1 contains ALET.
16 see note RCPIK below.

See notes: CANCPIK, NOPIK0.
Authorized: OCCF, VSE/POWER, VTAM, CICS.

AOTPTR Pointer to the AOT of the specified task
Supplied input: TASK.
Values returned: R1 is AOT pointer.
See notes: NORC, NOAUTHC, NOTID0, CANCTID.
Intended users: VTAM.

BALANCE

number of remaining physical bytes on track (See also description of balance and capacity value calculation by macro GETVCE and GETFLD FIELD=LOGBAL service)
Supplied input: No parameters of the macro are specified. Registers have to be loaded by the user.

R0 contains record descriptive information:
- Bytes 0-1: Length of one fixed data record
- Byte 2: Zero or length of key if keyed records
- Byte 3: Record number if no input balance is provided in R1 (see below) or zero.

R1 contains:
- Bytes 0-1: Logical unit identifier (as bytes 6-7 of CCB)
- Bytes 2-3: Input balance. This is the number of physical bytes left for the user on the track of the specified device.

Values returned: depending on return code in R0(!)

0 the record with the specified number fits on the track and R1 contains the new balance.
x'24' the record does not fit on the track any more, but the amount of user data of the longest record that would still fit on the track is contained in R1.
x'14' the input parameters in R0 or R1 contained invalid data.

See notes: NOAUTHC, NOCANC.
Intended users: BAM, RAS and ERP.

CELANCH
Provide ptr to CEL anchor table for current task.
Supplied input: nothing.
Values returned: R1 points to CEL anchor or is zero. R0 is set to 0.
See notes: NOAUTHR, NORC, BRANCH.

CLASSTAB
Dynamic class table
Supplied input: nothing.
Values returned: R1 points to class table.
See notes: NORC.
Authorized: key zero.

CNCLALL
Cancel all flag
Supplied input: nothing. Request is always for current task.
Values returned: x'01' in R1 if cancel has to be propagated to all tasks belonging to the same partition as the specified task, else 0.
See notes: NOFUTRC, NOCANC.
Authorized: SYSTEM.
Intended users: TERMINATOR

CNCLCODE
First cancel code
Supplied input: TASK ID.
Values returned: R1 contains first cancel code for specified TASK ID.
See notes: NOFUTRC, CANCTID, NOTID0, NOAUTHC, BRANCH.
Intended users: EOT, TERMINATOR, VSE/POWER, VTAM

CNCLCOD2
Last cancel code
Supplied input: TASK ID.
Values returned: R1 contains last cancel code for specified task.
See notes: NOFUTRC, NOTID0, CANCTID, NOAUTHC.
Intended users: EOT, TERMINATOR, VSE/POWER, VTAM.

COMRGPTR
Pointer to partition communication region
Supplied input: PIK in PART or SYSLOG ID in LOGID parameter.
Values returned: if R15 is:

0 R1 points to comreg of specified partition.
16 see note RCLOGPIK below.

For a static non-allocated partition return code is zero and pointer is provided.
See notes: NOPIK0, CANCPIK, NOAUTHC.
Intended users: VSE/POWER, ICCF, OCCF.

CPUTIME Amount of CPU time, in units of 16 micro seconds, charged to a partition since begin of job step. This function is available only if JA support was activated in the IPL SYS command. This counter overflows and is reset to zero each time it reaches a value of about 18 hours.

Supplied input: PIK of partition in PART parameter.
Values returned: if R15 is:

0 R1 contains value.
16 see note RCPIK below.

See notes: NOPIK0, CANCPIK, NOAUTHR.

DEVPROP
Device properties
Supplied input: either LU (logical unit) and PART (PIK of partition) or PU (PUB index) of device.
Values returned: if R15 is:

0 R1 contains 32 bit string, described by macro MAPPROP.
12 Logical unit specified is assigned ignore or unassigned.
16 see note RCPIK below.

See notes: NOPIK0, CANCPIK, CANCPUB, NOAUTHR.

DIBPTR Device information block pointer
Supplied input: logical unit in LU and either a PIK in PART or a SYSLOG ID in LOGID. LU may have the following values:

x'0000' for SYSRDR
x'0001' for SYSIPT
x'0002' for SYSPCH
x'0003' for SYSLST
x'0005' for SYSLNK

Values returned: if R15 is:

0 R1 points to device information block.
8 Specified partition is dynamic and LU value is not x'0005' (that means, SYSLNK).
16 LU value other then (0,1,2,3,5) or LOGID or PART belong to a dynamic non allocated partition.

See notes: NOPIK0, CANCPIK, NOAUTHR.

DSPACE Get data space information. Retrieve total amount of virtual storage which may be allocated by data spaces in a partition and the default size for creation of a single data space.
Supplied input: PIK in PART parameter or 0 for current partition
Values returned: R0 contains DFSIZE (form SYSDEF command), R1 contains PCBDSPAC, R15 return code
See notes: NOAUTHR, NOPIK0, CANCPIK

EOTACT End of task active
Supplied input: TASK ID.
Values returned: x'01' in R1 if flag EOTACT or EOTINPR are set to one, else 0.
See notes: NOFUTRC, NOTID0, CANCTID, NOAUTHC.
Intended users: ICCF.

EOTRTN Like EOTACT, but only flag EOTACT is tested.
See notes: NOAUTHR.

EXECMODE
Execution mode of partition.
Values returned: R1=0 means standard VSE partition, R1=4 means OS/390 partition
See notes: NOAUTHR, BRANCH.

FREELUB

Free logical unit
Supplied input: PIK in PART or SYSLOG ID in LOGID.
Values returned: if R15 is:

0 R0 is index of found free programmer logical unit, R1 points to LUB table entry.
4 No unassigned programmer lub found.
16 see note RCLOGPIK below.

See notes: NOPIK0, CANCPIK, NOAUTHR.

ICCFPP ICCF pseudo partition.
Supplied input: TASK ID.
Values returned: x'01' in R1 if specified task is assigned to an ICCF pseudo partition, else 0.
See notes: NOFUTRC, NOTID0, CANCTID, NOAUTHC, BRANCH.
Intended users: ICCF.

ICCFRO ICCF roll out
Supplied input: TASK ID.
Values returned: x'01' in R1 if the specified task is assigned to an ICCF pseudo partition and is eligible for for ICCF roll-out, else 0.
See notes: NOAUTHC, NOFUTRC, NOTID0, CANCTID.
Intended users: ICCF.

IPLTIME IPL time stamp.
Supplied input: nothing.
Values returned: R0 contains first part of IPL time stamp, R1 second part.
See notes: NOAUTHC, NORC.

ITACTIVE

Currently active timer exit
Supplied input: TASK ID.
Values returned: R0 points to IT exit routine. R1 points to the exit's save area. R2 is one of the following:

1 VSE timer is set

2 OS/390 STIMER is set

3 OS/390 STIMER REAL is set

4 OS/390 STIMERM is set

Zero values mean that no exit is defined.
See notes: NOAUTHR, NOTID0, CURRTID, BRANCH.

ITEXIT (ITEXIT1)

Interval timer exit
Supplied input: TASK ID.
Values returned: R0 points to IT exit routine. If bit 0 is on, the exit routine is currently active. R1 points to corresponding save area. Zero values in both registers indicate, that no exit is defined.
R2 is only modified if ITEXIT1 was specified. It will then contain the following bit settings.

BIT 28 is set if AMODE ANY was specified for this exit.

See notes: NOFUTRC (for ITEXIT only), NOAUTHR, NOTID0, CURRTID.

JCLPARM

Pointer to the PARM field of the job control EXEC statement
Supplied input: nothing.
Values returned: If R1=0 then there was no parameter specified.. Otherwise R1 points to a halfword containing the length of the parameter. The parameter itself (maximal 300 bytes) is located at displacement 2 relative to R1.
See notes: NOFUTRC, NOAUTHR

LOGBAL Number of remaining user data bytes on a track. (See description of balance and capacity value calculation in description of macro GETVCE and GETFLD FIELD=BALANCE) The user supplies a number of user data byte, that currently fit on the track and specifies the kind of record he wants to write. The service returns the number of user data byte, that will still fit on the track, after the write operation was carried out. The first input value can be obtained from the GETVCE macro, and is the 'maximum record size' for the specified key length. The necessary invocation of GETVCE is as follows:

```
[label]  GETVCE  REQUEST=TRKBAL,
                 OPTION=(REMOVE,MAXSIZE),
                 KEYLN=keylngth,
                 DATALEN=somelng,
                 LOGUNIT=logunit
```

The records may be of variable length, but the key length has to be equal for all records on the track. (keylngth is a one-byte field, logunit a two-byte field. Somelng is a two-byte field containing any value.)
Supplied input: No parameters of the macro are specified. Registers have to be loaded by the user.

R0 contains:
- Bytes 0-1: Number of user data byte in the record to be written.
- Byte 2: Zero or length of key if keyed records (must be equal for all records on the track).
- Byte 3: Must be zero.

R1 contains:
- Bytes 0-1: Logical unit identifier (as byte 6-7 of CCB)
- Bytes 2-3: Number of user data bytes presently fitting on the track.

Values returned: depends on return code in R0(!)

- 0 the record with the specified number fits on the track and R1 contains the new balance.
- x'24' the record does not fit on the track any more.
- x'14' the input parameters in R0 or R1 contained invalid data.

See notes: NOAUTHC
Special cancel condition: byte no. 3 of R0 was not zero on input.
Intended users: BAM, ASSEMBLER

LOGID SYSLOG ID of a partition
Supplied input: PIK in PART parameter
Values returned: if R15 is:

0 R1 byte 2-3 contain SYSLOG ID of partition.
16 dynamic partition for PIK not allocated.

See notes: NOAUTHR, NOPIK0, CANCPIK, BRANCH.

LTAACT Logical transient area active
Supplied input: TASK ID.
Values returned: x'01' in R1 if the LTA is active for the specified task, else 0.
See notes: NOFUTRC, CANCTID, NOTID0, NOAUTHC.
Intended users: ICCF.

LTAPTR Pointer to Logical Transient Area.
Supplied input: TASK ID.
Values returned: R1 contains the LTA address if currently owned by the specified task (TASK). (The LTA must not necessarily be active.) If not owned by the specified task, a value of zero is returned.
See notes: NOFUTRC, NOTID0, CANCTID, NOAUTHC.
Intended users: VSE/POWER, VTAM.

LUB Logical unit block entry
Supplied input: logical unit in LU and either PIK in PART or SYSLOG ID in LOGID parameter.
Values returned: if R15 is:

0 R1 points to normally assigned LUB table entry.
4 LU is assigned ignore.
8 LU is not assigned.
12 LU is invalid.
16 see note RCLOGPIK below.

See notes: NOAUTHR, NOPIK0, CANCPIK.

LUBTAB Logical unit block tables
Supplied input: PIK in PART or SYSLOG ID in LOGID parameter.
Values returned: if R15 is:

0 R1 points to first entry of table of system LUBs, R0 to first entry of table of programmer LUBs.
16 see note RCLOGPIK below.

See notes: NOAUTHR, NOPIK0, CANCPIK, BRANCH.

MAINTASK
TASK ID of partition maintask
Supplied input: TASK ID.
Values returned: R1 contains the TASK ID (IJBTIK) of the maintask of the partition to which the specified task belongs.
See notes: NOFUTRC, NOTID0, CANCTID, NOAUTHR.

MNTINFO
Returns the mount information for a device.
Supplied input: CUU to specify the device. Values returned: if R15 is:

0 The device is mount reserved for a partition. Register R0 contains the PIK of the partition in byte 2/3 and the bits x'0001' in byte 0/1 to indicate a temporary mount reservation or x'0002' to indicate a permanent one. Register R1 contains the value of the TLMECBSV field, which was set by the last MODFLD MNTINFO service.

4 indicates that the device is not mount reserved at all.
16 indicates an invalid input CUU= parameter.
20 indicates the specified CUU has invalid device type for this request.

See notes: NOAUTHR.

MODE Actual mode byte for tape devices, (that means, mode which is set by the system)
Supplied input: Either PART and LU (logical unit) or PU (PUB index) to identify the device. Values returned: if R15 is:

0 R1, byte 3, contains mode information
8 device is not a tape.
12 logical unit unassigned or assigned ignore.

See notes: NOAUTHR, NOPIK0, CANCPIK, CANCPUB.

MOUNTFLG
Mount flag
Supplied input: PUB index in PU.
Values returned: x'01' in R1 if the specified device has been reserved, to allow a volume change, else 0.
See notes: NOFUTRC, CANCPUB, NOAUTHR.

MSECS Current time slice for partition balancing in milliseconds.
Supplied input: nothing. Always for current partition.
Values returned: R1 contains value.
See notes: NOFUTRC, NOAUTHC.
Intended users: SYSTEM, JOB CONTROL

NPART Number of partitions
Supplied input: nothing.
Values returned:

R0, byte 0/1 number of static partitions
R0, byte 2/3 maximum number of partitions as defined by supervisor generation.
R1, byte 0/1 number of currently allocated dynamic partitions plus number of static partitions.
R1, byte 2/3 maximum number of partitions as defined in IPL SYS command.

See notes: NORC, NOAUTHR.

NSUB Number of attached subtasks
Supplied input: PIK in PART parameter.
Values returned: if R15 is:

0 R1 contains number of subtasks.
16 see note RCPIK below.

See notes: CANCPIK, NOPIK0, NOAUTHR, BRANCH.

NUMLUB Number of logical unit block entries
Supplied input: LOGID or PART to identify partition. Values returned: if R15 is:

0 R0 contains number of system LUBs, R1 number of programmer LUBs.
16 see note RCLOGPIK below.

See notes: NOAUTHR, CANCPIK, NOPIK0.

NXTOWNER

Give the next owner of a device starting from the partition identified by the input PIK value. As a start value specify a PIK of zero. System scans the owner table information and returns the PIK of the next owner it finds. (This PIK must be incremented by the caller (by x10) to not create a loop with the next invocation!)
Supplied input: PART to identify partition, PU to identify the device
Values returned: if R15 is:

0 R0 is set to 0, R1 contains the PIK of the next device owner.
4 There is no further owner from the specified PIK onwards.
8 input PIK is invalid. (End of loop, since PIK is too high)

See notes: NOAUTHR, CANCPUB, SYSPIK.

OCCFACT

OCCF active.
Supplied input: TASK ID.
Values returned: x'01' in R1 if at least one OCCF request is pending for the specified task, else 0.
See notes: NOFUTRC, CANCTID, NOTID0, NOAUTHC.
Intended users: OCCF.

OCEXIT (OCEXIT1)

Pointer to OC exit routine and save area.
Supplied input: PIK of current partition in PART parameter or zero.
Values returned: The pointers are returned in R0 and R1 respectively. Zero values are returned, if no exit is defined. Bit 0 in R0 is on, if the exit routine is currently active.
R2 is only modified if OCEXIT1 was specified. It will then contain the following bit settings.

BIT 29 is set if MSGDATA=YES was specified.

BIT 28 is set if AMODE ANY was specified for this exit.

See notes: NOFUTRC (for OCEXIT only), NOAUTHR, NOPIK0, CURRPIK.

OPENSVA

Open routine executing in SVA.
Supplied input: nothing. Request always for current task.
Values returned: x'01' in R1 if this is the case, else 0.
See notes: NOFUTRC, NOAUTHR.

OWNER Owner of specified device.
Supplied input: PUB index of device in PU.
Values returned: if R15 is:

0 No owner.
4 Unique device owner. 2-byte PIK of owner in byte 2/3 of R1. For a VTAM-owned device, the PIK of the VTAM partition is returned.
8 Multiple device owners (DASD).

See notes: NOAUTHR, CANCPUB.

PAGEIN Number of page-in I/Os.
Supplied input: nothing.
Values returned: Number of page-in I/Os in whole system since IPL or last wrap-around of the 3-byte counter. The function depends on JA support being active.
See notes: NOFUTRC, NOAUTHR.

PAGEOUT
Same function as PAGEIN, but page out I/O s.

PCBPTR Pointer to partition control block.
Supplied input: PIK in PART.
Values returned: if R15 is:

0 pointer in R1
16 see note RCPIK below.

See notes: SYSPIK, CANCPIK, NOAUTHC.
Intended users: SYSTEM

PCEATAB
Partition control block extension address table
Supplied input: nothing.
Values returned: R1 points to this table.
See notes: NORC, NOAUTHR.

PCEPTR Pointer to partition control block extension.
Supplied input: LOGID or PART to identify partition.
Values returned: if R15 is:

0 R1 points to PCE of specified partition.
16 dynamic partition for PIK not allocated.

See notes: NOAUTHR, NOPIK0, CANCPIK.

PCEXIT (PCEXIT1)
Pointer to PC exit routine and save area.
Supplied input: TASK ID of current task or 0.
Values returned: R0 points to PC exit routine. If Bit zero in R0 is on the exit routine is currently active.
R1 points to the save area. If no exit is defined, both pointers are zero.
R2 is only modified if PCEXIT1 was specified. It will then contain the following bit settings.

BIT 28 is set if AMODE ANY was specified for this exit.

See notes: NOFUTRC (for PCEXIT only), NOAUTHR, NOTID0, CURRTID.

PIB Partition information blocks
Supplied input: LOGID or PART to identify partition.
Values returned: if R15 is:

0 R0 points to PIB, R1 to PIB2 of partition.
16 see note RCLOGPIK below.

See notes: NOPIK0, CANCPIK, NOAUTHR, BRANCH.

PIK Partition identification key
Supplied input: SYSLOG ID of partition in LOGID parameter or TASK ID.
Values returned: if R15 is:

0 PIK in R1. Zero for system partition if provided TASK ID belonged to system task.
16 see note RCLOGPIK below.

See notes: CANCTID, NOTID0, NOAUTHR, BRANCH.

PMODE Get saved permanent mode set function byte for tape devices.
Supplied input: Either PART and LU or PU (pub index) to identify the device.
Values returned: if R15 is:

0 R1, byte 3 contains mode information
8 device is not a tape
12 logical unit unassigned or assigned ignore.

See notes: NOAUTHR, NOPIK0, CANCPIK, CANCPUB

POWCAT Provide ptr to VSE/POWER CAT table.
Supplied input: none
Values returned: if R15 is:

0 R1 is ptr to CAT table. R0 is set to 0.
4 ptr is not set in syscom.

See notes: NOAUTHC.
Intended users: POWER

POWJOB VSE/POWER job information
Supplied input: PART or LOGID of partition
Values returned: if R15 is:

0 R1 points to VSE/POWER job information for specified partition.
16 see note RCLOGPIK below.

See notes: NOPIK0, CANCPIK, NOAUTHC.
Intended users: CICS.

POWSYS Power system information
Supplied input: nothing.
Values returned: R0 contains the one character VSE/POWER SYSID in the low order byte or x'40' when no SYSID is available, or x'00' when POWER is not active. On return R1 points to the 8 byte VSE/POWER PNET node ID.
See notes: NORC, NOCANC, NOAUTHC.
Intended users: CICS.

PPSAVAR
Pointer to the problem program save area.
Supplied input: current TASK ID in TASK or zero.
Values returned: R1 points to the problem program save area. This is the partition save area for maintasks whereas for subtasks it is the save area specified in the ATTACH macro.
See notes: NOFUTRC, NOTID0, CURRTID, NOAUTHR.

PSTAT Status of a partition
Supplied input: SYSLOG ID in LOGID or PIK in PART parameter.
Values returned: if R15 is:

0 R1 contains bit flags, representing the following status:
X'08' Partition stopped
X'10' Partition dynamic

X'20' Partition active
X'40' Partition allocated

16 Supplied LOGID is invalid (not just not allocated) or 'AR'.

See notes: CANCPIK, NOPIK0, NOAUTHR, BRANCH.

PTRACE Check if interactive trace progam is initialized
Supplied input: PIK in PART parameter or 0 for current partition
Values returned: if R15 is:

0 indicating that no trace is active for this partition
16 indicating that trace is active for this partition

See notes: NOAUTHR

PU Index of physical unit block entry
Supplied input: CUU for which information is to be retrieved
Values returned: if R15 is:

0 Information retrieved. R1 contains PUB index.
16 No information for this cuu available.

See notes: NOAUTHR.

PU Index of physical unit block entry
Supplied input: PART (PIK of partition) and LU (logical unit) of device in question
Values returned: if R15 is:

0 LU normally assigned. R1 contains PUB index.
4 LU is assigned ignore. R1 contains x'FE' in byte 3.
8 LU is unassigned. R1 contains x'FF' in byte 3.
12 LU is invalid.
16 dynamic partition for PIK is not allocated.
20 LU spooled by power. R1 contains PUB index.

See notes: SYSPIK, CANCPIK, NOAUTHR.

PUB Physical unit block entry
Supplied input: LU (logical unit of device) and either PART or LOGID to identify the partition.
Values returned: if R15 is:

0 LU normally assigned. R0 contains PUB index, R1 points to PUB entry.
4 LU is assigned ignore.
8 LU is unassigned.
12 LU is invalid.
16 see note RCLOGPIK below.
20 LU spooled by power. R0 contains PUB index, R1 points to PUB entry.

See notes: NOPIK0, CANCPIK, NOAUTHC.
Intended users: VSE/POWER

PUB2 Physical unit block 2
Supplied input: pubindex of device in PU.
Values returned: R1 points to PUB2 entry of device.
See notes: NORC, CANCPUB, NOAUTHR.

PUBPTR Pointer to physical unit block table
Supplied input: nothing.
Values returned: R1 points to first entry in physical unit block table.
See notes: NORC, NOAUTHR, BRANCH.

PUBXPTR
Pointer to Physical Unit Block Extension entry.
Supplied input: PUB index in PU or CUU in CUU= parameter.
Values returned: R1 contains pointer.
See notes: NOFUTRC, CANCPUB, NOAUTHR.

PUSECNT
Number of physical users
Supplied input: PIK in PART parameter and PUB index of device in PU.
Values returned: R1 contains the number of physical users that access this device using "physical addressing". The returned value applies to the specified partition only. A negative value will be returned if

- the device is spooled by VSE/POWER.
- $$LST and $$PUN have DISP=N.
- the device is temporary reset to a partition.

See notes: CANCPUB, CANCPIK, SYSPIK, NOAUTHR, RCPIK.

REALKEY
retrieve the storage key of a storage location identified by the address and access list entry token supplied by the caller. If ALET is specified, the service enters access register mode, regardless of the processing mode of the caller. Callers processing mode is not changed when the caller is redispatched.
Supplied input: ADDR to specify the address and ALET to specify the access list entry token.
Values returned: depends on the returncode in register RF.

0 indicating that the storage key is in byte 3 of register R1. The key is in the format as stored by the ISKE instruction with the fetch, reference and change bit set to zero (i.e. one of the hexadecimal values: 00, 10, 20, 30, 40, 50, 60, 70, 80, 90, A0, B0, C0, D0, E0, or F0) The page may or may not be in storage but it is not paged in by this service.
4 indicating that the address was invalid.

RXEOJPTR
retrieve ptr for REXX interpreter for current task.
Supplied input: none
Values returned: R1 contains the required pointer from the current tasks task control block.
See notes: NOAUTHC, NORC.

SAVAR (SAVAR1)
Pointer to a tasks save area and in case of SAVAR1 to tasks AR save area.
Supplied input: TASK ID.
Values returned: R1 points to tasks save area and in case of SAVAR1 R2 points to AR save area.
If the provided TASK ID is not the ID of the current task, only 'system routines' are authorized. In this case service is no fast path SVC.
See notes: NORC, CANCTID, NOTID0, NOAUTHR, BRANCH.

SSFLAGS

Subsystem flags.
Supplied input: a PIK in PART parameter or zero for current partition.
Values returned: R1 contains a bit string identifying the subsystem (if any) active in the specified partition.

x'010000' Security Server
x'008000' User susystem
x'4000' IBM subsystem
x'002000' PSF print support facility
x'001000' DSNX
x'000800' ICKDSF
x'000400' Customization
x'000200' NETVIEW
x'000100' NPDA
x'000080' VSE/POWER
x'000040' VTAM
x'000020' ICCF
x'000010' CICS 2.X
x'000008' VM/VCNA
x'000004' OCCF
x'000002' SQL/DS
x'000001' SSX

See notes: NORC, NOPIK0, CANCPIK, NOAUTHC.
Intended users: SYSTEM.

STATUS Status of specified task.
Supplied input: TASK ID.
Values returned: Byte 3 of R1 indicates the status of the task. Currently committed values are:

X'81' Task is waiting for the LTA
X'82' Task is waiting for a CCB/IORB/ECB to be posted

See notes: NORC, NOTID0, CANCTID, NOAUTHC.
Intended users: ICCF.

STORKEY

Storage protection key
Supplied input: LOGID or PART
Values returned: if R15 is:

0 R1, byte 3, contains storage key of partition.
16 see note RCLOGPIK below.

See notes: NOAUTHR, NOPIK0, CANCPIK, BRANCH.

SUBCHN Subchannel number for a device
Supplied input: PU
Values returned: if R15 is 0 R1 contains the subchannel number in byte 2/3 of the register. See notes: NOAUTHR, CANCPUB.

SYSRESW

DASD file protection flag
Supplied input: nothing. Request is always for current task.
Values returned: x'01' in R1 if DASD file protection is being bypassed for this task, else 0.
See notes: NOFUTRC, NOCANC, NOAUTHR.

TCBPTR Task control block pointer
Supplied input: TASK ID.
Values returned: R1 points to TCB of task.
See notes: NOFUTRC, NOTID0, CANCTID, NOAUTHC, BRANCH.
Intended users: SYSTEM.

TERMACT
Terminator active flag.
Supplied input: TASK ID.
Values returned: R1 contains 1 if terminator is active for the specified task, else 0 (TASK).
See notes: NOFUTRC, CANCTID, NOTID0, NOAUTHC.
Intended users: ICCF.

TIDSTR Task id string pointer.
Supplied input: PIK in PART. Values returned: if R15 is:

0 pointer in R1
16 dynamic partition for PIK not allocated.

See notes: CANCPIK.

UCB Retrieve device information into an OS/390 UCB. The UCB data area has to be provided by the caller.
Supplied input: CUU in CUU parameter and UCB address in ADDR parameter
Values returned: R1 contains the ADDR parameter, if there is no attention address set for the specified CPU. R1 contains 0, if there is an attention address set for the specified CPU. See also SVC03 for setting an d resetting of attention exits. If R15 is:

0 the user supplied UCB contains device status and pub index.
16 no information available for the specified CUU.

See notes: NOAUTHC
Intended users: OSA/SF

USECNT Number of current or eventually stored assignments plus number of users that access this device using "Physical addressing". All other details like FIELD=PUSECNT service.

VSAMOPEN
VSAM open flag.
Supplied input: nothing. Request is always for current partition.
Values returned: x'01' in R1 if there is at least one open VSAM ACB for the partition, else 0.
See notes: NOFUTRC, NOCANC, NOAUTHC.
Intended users: EOT.

VTAMDISP
VTAM AP exit flag.
Supplied input: TASK ID.
Values returned: x'01' in R1 if the VTAM AP exit is scheduled for the specified task, else 0.
See notes: NOFUTRC, NOTID0, CANCTID, NOAUTHC.
Intended users: VTAM.

VTAMOPEN
VTAM ACB open flag.
Supplied input: TASK ID.

Values returned: x'01' in R1 if there is at least one open VTAM ACB for this task, else 0.
See notes: NOFUTRC, NOTID0, CANCTID, NOAUTHC.
Intended users: VTAM.

Notes mentioned in field description above:

NORC No return code is provided by this service. Still R15 is always cleared on return to the user, indicating that the service completed successfully. A number of return codes may be introduced for this service in future times. Still return code zero will always indicate 'no error encountered'.

NOFUTRC Although R15 is set to 0 on return to the user, this service will not have future return codes added for compatibility reasons.

NOAUTHR No special authorization is required for this service. It is open to all users.

NOAUTHC Although no authority checking is done in this service it should only be used by the mentioned intended users, since only these users will be notified about interface changes.

CANCPIK The caller of this service is canceled, if the provided pik is invalid. A PIK is invalid if it is not a multiple of 16 or not smaller than the maximum PIK mentioned in SYSCOM.

CURRPIK Here the user is canceled if the PIK he provided (if he did provide one) was not equal to the PIK of the current partition (that means, the partition being serviced).

NOPIK0 Providing a PART value of zero or omitting the PART parameter for this service will be handled as if the PIK of the current partition had been supplied by the caller.

SYSPIK Providing a zero with the PART parameter or omitting this parameter will be considered as identification of the SYSTEM partition.

CANCPUB The user is canceled if PUB index is invalid.

CANCUSE Unauthorized users are canceled.

CANCTID The user is canceled if the provided TASK ID was invalid, that means:

- if note NOTID0 does not apply and the TASK ID is zero.
- if TASK ID is bigger maximum possible TASK ID.
- if task information block for task does not exist.
- if task is an inactive subtask.

CURRTID The user is canceled if the provided TASK ID is not equal to the current TASK ID.

NOTID0 Providing a TASK ID value of zero or omitting the TASK parameter for this service will be handled as if the TASK ID of the current task had been provided.

SYSTID Providing a TASK ID of zero or omitting the TASK parameter will lead to a cancel of the callers task.

NOCANC There is no cancel condition for this service.

RCLOGPIK A return code of 16 is provided in case a dynamic partition was identified by the provided LOGID or PART parameter and the the partition was not allocated for some reason. It may also be that the LOGID was illegal or the class was not contained or not enabled in the current class table.
Information for a static partition will still be provided even if it is not allocated. Exceptions to this rule are mentioned in the service descriptions above.

RCPIK See note RCLOGPIK, just that you could specify only a PART parameter and no LOGID.

BRANCH This service can be called with BRANCH=YES, if the requested information is for the current task. The caller has to provide a save area if he wants to use this interface.

GETJA

The GETJA macro updates, clears or resets supervisor maintained information in the partition's job accounting tables. It is issued by job control at the end of a job step just before the user-written phase $JOBACCT is fetched. The format of the macro is as follows:

```
[name]   GETJA   [PART={name1|(r1)|1}
             [,ACTION={UPDATE|CLRTIME|RESET}]
```

PART Name of a 2-byte field or register containing the PIK of the partition, to which the request refers.
If PART is omitted the current partition is assumed.

ACTION Defines the function that is to be performed. This operand is required for JOB CONTROL and it is obsolete for other programs, since other programs usage is defaulted to ACTION=UPDATE.

UPDATE The time and SIO related accounting fields residing in supervisor storage are updated.

CLRTIME The time related accounting fields residing in supervisor storage are cleared.

The start I/O related accounting fields residing in supervisor storage are updated.

RESET The time and SIO related accounting fields residing in supervisor storage are reset and in addition, CPU time is moved to OVERHEAD time.

GETPRTY

The GETPRTY macro returns the dispatching and balancing sequence of the partitions and dynamic classes in the system. The macro has the following format:

```
ASSEMBLER

[name] GETPRTY {(1)|area},{(0)|length}
                    [,TYPE={STATIC|ALL}]
```

Figure 11. GETPRTY

area	Address of an area where the information is to be stored.
length	Length of area where the information is to be stored.
TYPE=STATIC	(default) Only the dispatching and balancing sequence of the static partitions is returned. The "length" has to be at least 3 times number of static partitions.
TYPE=ALL	The dispatching and balancing sequence of the static partition and dynamic classes is returned. The "length" has to be at least 3 times (number of static partitions + max. number of dynamic classes).

Input register:

Register 0	Contains the length for TYPE=STATIC and the pointer to a parameter list for TYPE=ALL.
Register 1	Contains the area address for TYPE=STATIC and the pointer to a parameter list for TYPE=ALL.

Work register:
Register 15 (TYPE=ALL only).

Output:

TYPE=STATIC	The 2-byte PIKs of static partition entries are placed into the user supplied area in ascending order of dispatching priorities.
TYPE=ALL	The 2-byte dynamic class entries are built by the 1-byte dynamic class character and X'FF'. The 2-byte PIKs of static partition and dynamic class entries are placed into the user supplied area in ascending order of dispatching priorities.

The entries are separated by a comma (X'6B'), if the partitions/classes do not participate in balancing whereas the members of a balancing group are separated by an equal-sign (X'7E'). The output will be truncated if the specified length is too small. When the returned string is shorter than the output area, the string will end with X'40'.

Cancel Conditions:

Illegal SVC	Specified length <= 0.

GETVCE

The GETVCE macro returns to the user a specific volume characteristic entry, retrieved from the Volume Characteristics Table (VCT). It also returns information about the track capacity or track balance. For the Automatic Volume Recognition refer also to z/VSE Supervisor Diagnosis Reference, Chapter "Physical Input/Output Control System".

The macro has the following format:

```
[name]  GETVCE  AREA={name1|(S,name1)|(r1)},
                {DEVICE={SYSxxx|X'cuu'|DASD}
                   |VOLID={name2|(S,name2|(r2)}[,DEVTYPE={name3|(S,name3)|(r3)}]
                   |LOGUNIT={name4|(S,name4)|(r4)}
                   |CHNUNIT={name5|(S,name5)|(r5)}}
                [,CLASS={TAPE|DASD|ANY}]
                [,LENGTH=n]
                [,MFG={name6|(S,name6)|(r6)}]
                [,REQUEST={TRKBAL|TRKCAP}]
                [,DATALEN={name7|(S,name7)|(r7)}]
                [,KEYLEN={name8|(S,name8)|(r8)}]
                [,RECNO={name9|(S,name9)|(r9)}]
                [,BALANCE={name10|(S,name10)|(r10)}]
                [,OPTION ={(REMOVE{,MAXSIZE{,LAST}})})]
```

AREA Address of an area where the user wants the specified volume characteristic entry to be stored.

CLASS Indicates the device class for which the request has to be done. Default is DASD (that means, disk or diskette), ANY includes TAPE and DASD.

DEVICE Identifies the device whose volume characteristics entry the user wants to be returned. The specification may refer to a logical unit as well as to a physical one.

DASD To retrieve (cuu,VOLID) for all DASDs.

VOLID Address of the VOLID. The volume characteristic entry that the user wants to be returned is identified by its volume serial number. This is a 6-byte field and when specified with register notation also points to a 6-byte field.

If duplicate VOLIDs are present in the system, the first one found by GETVCE is saved and returned to the user if the associated device is not DOWN or, if all the associated devices are DOWN. In all other cases the first device which is not DOWN will be returned to the user.

The VOLID can, however, be further qualified by the DEVTYPE parameter which limits the search to all 3330s or all 3340s, etc.

DEVTYPE Address of device type (1-byte field). If VOLID is given , it can be qualified by the PUB device type code. With this parameter, GETVCE will return volume and device information for the volume with matching VOLID and device type (for example 3340 or 3330).

LOGUNIT Address of device description in logical unit format. The volume characteristic entry that the user wants to be returned is identified by its logical assignment. This operand points to a halfword with the same format as a logical unit number in a CCB.

CHNUNIT Address of device description in physical unit (channel, unit) format. The volume characteristic entry that the user wants to be returned is identified by its physical device address. This operand points to a halfword with the physical device address.

LENGTH Length of data to be placed in AREA by GETVCE. AREA is cleared for this length and the data is moved in. Default is the maximum defined length.

MFG Address of a dynamic storage area that is to be used for parameter list construction for re-entrant programs.

REQUEST

TRKBAL The user requests the track balance (number of remaining data bytes on a track) of the specified DASD device to be returned.

TRKCAP The user requests the number of whole records to be returned that will fit into the given or calculated track balance.

DATALEN Pointer to a two-byte field containing the length of one fixed length data record. This field is processed as a unsigned binary value.

KEYLEN Pointer to a one-byte field containing the key length of one fixed-length data record. This field is processed as an unsigned binary value. If non-keyed records are processed, this byte must either be set to zero or the keyword must be omitted.

RECNO Pointer to a one-byte field containing the record number. A record number of zero results in the maximum track balance being returned to the user or being used for track capacity calculations.

BALANCE Pointer to a two-byte field containing the track balance that is to be used for calculation. This balance field is processed as an unsigned binary value. If the balance is not known, this parameter must be omitted.

OPTION

REMOVE The given record with the specified DATALEN and KEYLEN is assumed as having been removed from the track by the user. GETVCE processing will calculate and/or increment the track balance or track capacity. If the user also provided a balance, it must always equal the number of bytes available on the track before the record was removed.

MAXSIZE The maximum data record length is to be returned to the user (keylength has already been taken into account).

LAST The caller requests that the actual physical record size of the last record is to be used as the decrement/increment to obtain the output balance.

Output:

AREA Is initially cleared, before the requested information is moved in the specified length.
The 'AVRLIST DSECT=YES,DEVICE=YES' macro describes the layout of the max. information returned to the user.

R0 Feedback information varies depending on the function (TRKCAP or TRKBAL) and the results.

TRKCAP Set to the number of whole records that will fit on the remainder of a track (input track balance).

TRKBAL If a new record fits or an old record is removed, R0 contains the updated track balance. If a whole record would not fit (R15=X'24') and MAXSIZE was specified, R0 is set to the number of maximum writable DATA bytes. (KEY bytes have already been taken care off). If a whole record would not fit and MAXSIZE was not requested, R0 is set to zero.

R15 Contains one of the following return codes:

0 (X'00')	Successful completion.
4 (X'04')	Successful completion, but some data is not valid (described by AVRFLG)
8 (X'08')	The volume specified is not mounted, or the logical unit specified is not assigned, or the specified unit is not included in the system.
12 (X'0C')	The logical unit specified is assigned 'IGNORE'.
16 (X'10')	The device is not operational.
20 (X'14')	The parameter list is invalid (for example, logical unit number too high).
24 (X'18')	The given logical unit or device belongs not to the class specified by the CLASS parameter.
28 (X'1C')	The device is not ready.
36 (X'24')	For REQUEST=TRKBAL or TRKCAP only: The input balance is not sufficient to accommodate a record of the specified key and data length. MAXSIZE was specified and at least one byte of data could be written. R0 is set to the maximum number of DATA bytes that will fit.

AVRLIST and DCTENTRY

The contents of the VCT entry is described by the AVRLIST and the contents of the DCT entry is described by the DCTENTRY macro:

```
[name]   AVRLIST  [DSECT={YES|NO}][,DEVICE={NO|YES}]

[name]   DCTENTRY [DSECT={YES|NO}]
```

DSECT Determines whether a DSECT statement is to be generated or not,

DEVICE Determines whether DCTENTRY is called within AVRLIST thus describing all the output within one DSECT.

Defaults are: DSECT=YES, DEVICE=NO.

INVPART

The INVPART macro invalidates the partition and initializes the partition for the execution of the (next) job step (see also SVC 58). This includes GETREAL for real execution. The function is restricted to programs with a PSW key of zero.

The macro has the following format:

```
[name]  INVPART  [REAL={YES|NO}]
```

REAL

- YES: Switch to 'real' mode required.
- NO: No switch to 'real' mode required.

General register 2 will be destroyed.

MCSOPER Macro

The MCSOPER macro requests the system to activate or deactivate a VSE console session. This macro supports a compatible subset of the form provided on MVS. Following parameters or options of the MVS version are not supported on VSE:

ABTERM, ALERTPCT, MIGID, QLIMIT, QRESUME
REQUEST=RELEASE

MCSOPER must be invoked in supervisor state (any PSW-key), primary ASC-mode and 31-bit addressing mode, and is not supported from ICCF interactive partitions.

Depending on the options specified for REQUEST and MSGDLVRY, and on the macro format used (MF parameter), other parameters may be required, optional or invalid. The applicability for REQUEST and MSGDLVRY options is summarized in Figure 12 on page 58 and Figure 13 on page 59. For the dependency on the macro format, refer to the description of the MF parameter.

Registers 0, 1, 14 and 15 are used by the macro and must be saved, if necessary, prior to macro invocation. Except where noted otherwise, registers 2 to 12 may be used for register notation.

```
[name]  MCSOPER  [REQUEST={ACTIVATE|DEACTIVATE}
                 [,NAME={address|(gr)}]
                 [,CONSID={address|(gr)}]
                 [,TERMNAME={address|(gr)}]
                 [,MCSCSA={address|(gr)}]
                 [,MCSCSAA={address|(gr)}]
                 [,OPERPARM={address|(gr)}]
                 [,MSGDLVRY={FIFO|SEARCH|NONE}]
                 [,MSGECB={address|(gr)}]
                 [,ALERTECB={address|(gr)}]
                 [,RTNCODE={address|(gr)}]
                 [,RSNCODE={address|(gr)}]
                 [,MF={S|(L,name[,{string|0D}])|
                 ({M|E},{address|(gr)}[,{
                 COMPLETE|NOCHECK}])}]
```

REQUEST={**ACTIVATE**|**DEACTIVATE**}

Specifies the MCSOPER function requested:

ACTIVATE	A new console session is to be established.
DEACTIVATE	An active console session is to be terminated.

NAME={*address*|(*gr*)}

Specifies the address of an 8-byte field containing the name of the console being activated or deactivated. All active consoles must have a unique name. The name must be a 2- to 8-byte string of characters with values higher than x'40' and lower than x'FF', left justified and padded with blanks.

VSE recognizes a predefined set of names, each identifying one of the following standard consoles:

IC	Integrated Console
SYS	3270/3215 System Console (CRT/Line Mode)

HCF HCF task dummy console for replying to WTOR's with HRDCPY option
JCL Job Control dummy console for issuing AR commands
NET NetView Console
S30 Dummy console used for SVC 30 processing
VMC VMCF Console (used for all CMS users)

Other names must be at least 4 characters long and are interpreted as verified userids, and also used for authorization and auditing purposes. Since VSE userids are at least 4 characters long, overlaps with above predefined names are excluded.

Console names are also matched with userids specified on the ECHO or ECHOU parameter of the POWER *$$ JOB statement, for the purpose of routing job-related messages.

CONSID={*address*|(*gr*)}

Specifies the address of a 4-byte area for returning (ACTIVATE) or retrieving (DEACTIVATE) a unique, system defined console ID for this console. This ID is used for all subsequent console requests.

TERMNAME={*address*|(*gr*)}

Specifies the address of an 8-byte field containing an identification of the terminal originating the request. This parameter is only provided for compatibility, its value is ignored on VSE.

MCSCSA={*address*|(*gr*)}

This parameter specifies the address of a 4-byte area, where the address of a system defined Console Status Area (CSA) is returned. This area is used to indicate special conditions pertaining to this console, together with ALERTECB when used (see below).

The CSA-address of the console used to IPL the system (IC or CRT) changes during IPL and is updated automatically, when the transition to IPL stage III occurs.

MCSCSAA={*address*|(*gr*)}

This parameter specifies the address of a 4-byte area, where the ALET of the space containing the console status area is returned. VSE currently returns 0 in this field.

OPERPARM={*address*|(*gr*)}

Specifies the address of a profile area containing console attributes such as authority (master or user), routing codes, DOM options, etc.. This area is mapped by the MVS-compatible mapping macro IEZVG111. Its use, and the default attributes assumed if OPERPARM is omitted, are explained in detail in “Operator Parameter Area” on page 59

MSGDLVRY={**FIFO**|**SEARCH**|**NONE**}

Specifies how messages are delivered to this console:

FIFO Messages are delivered in first-in first-out sequence.
SEARCH Messages may be delivered based on search arguments specified on the MCSOPMSG macro.
NONE No messages are delivered, the console is used for input only.

MSGECB={*address*|(*gr*)}

Specifies the address of an ECB that is posted whenever a message is queued for delivery to this console.

ALERTECB={*address*|(*gr*)}

Specifies the address of an ECB that is posted whenever a special condition has occurred for this console. Additional information about such conditions is then provided in the CSA.

RTNCODE={*address*|(*gr*)}]

Specifies the address of a 4-byte area where the return code is to be stored. When omitted, the return code is placed in register 15.

RSNCODE={*address*|(*gr*)}]

Specifies the address of a 4-byte area where the reason code is to be stored. When omitted, the reason code is placed in register 0.

MF=**S** Specifies the standard form of the macro. It generates an in-line parameter list, with the specified parameters and with defaults for omitted parameters, and code to invoke the requested service. Full checking for valid parameter combinations is included.

MF=(**L**,*name*[,{*string*|0D}])

Specifies the list form of the macro. It generates a parameter list in a storage area defined by *name*. The optional *string* parameter specifies a string of 1 to 60 characters, to be used as the argument of the generated DS instruction. No other macro parameters may be specified with this option.

MF=(**M**,{*address*|(*gr*)}[,{ **COMPLETE**|**NOCHECK**}])

Specifies the modify form of the macro. It generates code to store specified parameters in a parameter list at the specified address. Checking for required parameters and supply of defaults is included (COMPLETE) or omitted (NOCHECK). When COMPLETE is specified, the parameter list is cleared before filling. Otherwise, only the specified parameters are updated.

MF=(**E**,{*address*|(*gr*)}[,{ **COMPLETE**|**NOCHECK**}])

Specifies the execute form of the macro. It generates code to store specified parameters in a parameter list at the specified address and to invoke the requested service. Checking for required parameters and supply of defaults is included (COMPLETE) or omitted (NOCHECK). When COMPLETE is specified, the parameter list is cleared before filling. Otherwise, only the specified parameters are updated.

Figure 12. Applicability of MCSOPER Parameters for REQUEST Options

Parameters	REQUEST=ACTIVATE	REQUEST=DEACTIVATE
NAME	required	invalid
CONSID	required	required
TERMNAME	required	invalid
MCSCSA	required	invalid
MCSCSAA	required	invalid
OPERPARM	optional	invalid
MSGDLVRY	optional	invalid
RTNCODE	optional	optional
RSNCODE	optional	optional

Figure 13. Applicability of MCSOPER Parameters for MSGDLVRY Options

Parameters	MSGDLVRY=FIFO	MSGDLVRY=SEARCH	MSGDLVRY=NONE
MSGECB	required	required	invalid
ALERTECB	optional	optional	invalid

Return codes and reason codes:

00 00 Successful completion.

04 00 Console with specified name is already active (ACTIVATE) or not active (DEACTIVATE).

10 00 Invalid input: The address of the parameter list or of an input parameter is invalid.

10 02 Invalid input: The specified console was not activated by this task (DEACTIVATE).

10 04 Invalid input: The requested function is invalid (not ACTIVATE nor DEACTIVATE).

10 08 Invalid input: The specified name contains invalid characters, or is none of the predefined values nor a valid VSE userid (ACTIVATE).

10 10 Invalid input: The specified MSGDLVRY option is invalid (ACTIVATE).

10 18 Invalid input: The specified authority level (OPERPARM area) is invalid.

10 20 Invalid input: The specified message level (OPERPARM area) is invalid.

10 2C Invalid input: The macro acronym or version indicator in the parameter list is invalid.

14 00 Service routine failure.

18 00 The caller is not in supervisor state or not in primary ASC mode or not in 31-bit addressing mode.

Cancel codes: None.

After a console has been successfully activated, the system attempts to deliver all 'hold' messages that are eligible for this console. The MSGECB is immediately posted after activation, and the application should issue the MCSOPMSG macro with the default option CMDRESP=NO until return code 8 is presented (no message available). If MCSOPMSG is issued with CMDRESP=YES before receiving all 'hold' messages, the delivery of such messages is terminated and only current messages can be received from that point on.

Operator Parameter Area

The Operator Parameter Area is used to specify console attributes passed to the MCSOPER macro via the OPERPARM parameter. Its lay-out is described by the macro IEZVG111, that is taken over from MVS. Fields supported on VSE are listed below, the contents of unsupported fields is ignored.

MCSOAUTH Specifies the requested authority level for commands entered at this console. The authority levels supported on VSE are:

MASTER All commands are allowed. This authority level is used for all VSE system and master consoles.

INFO Only a subset of commands is allowed. The subset is defined by command processors (AR, POWER, ICCF, OCCF, VTAM and OC exits). This authority level also restricts the scope of messages, that can be replied to by this console.

The default option is INFO. Intermediate levels of authorization, like SYS, IO or CONS, are not supported on VSE.

MCSOMLVL Specifies the levels of messages that can be routed to this console. Message levels are defined as follows:

- **R** Messages issued via WTOR or SYSLOG I/O with a READ CCW.
- **A** Messages with descriptor codes 1 or 2 and requiring immediate master console action other than a reply.
- **CE** Messages with descriptor code 11 (critical eventual action).
- **E** Messages with descriptor code 3 (critical eventual action).
- **NB** Broadcast messages (not supported on VSE).
- **I** Information messages.
- **ALL** Includes all above levels (default).

MCSORCDT Specifies, by a 128-bit string, the set of routing codes of messages that can be routed to this console. Alternatively, following options can be specified:

- **ALL** All routing codes are received. This option is used for all VSE system and master consoles, except for the Integrated Console.
- **NONE** Routing to this console is not based on routing codes (default value).

This specification is ignored for consoles with authority level of INFO, and the default value NONE is assumed.

MCSODOM Specifies what DOM requests are to be sent to this console, with following options:

- **NORMAL** DOM requests are sent for all messages that could have been routed to this console (default value).
- **ALL** All DOM requests are sent to this console.
- **NONE** No DOM requests are sent.

MCSOMISC Specifies miscellaneous routing information. The default settings for all options is 'off'. Only the following options are supported on VSE:

- **UD** This console receives messages that cannot be routed to any other console. On VSE, this option is restricted to consoles with MASTER authority and covers all messages that need to be displayed on a master console. It is used by the Integrated Console.
- **AUTO** This console receives messages that are flagged as eligible for automation. This option also requires MASTER authority. When OCCF is active, it is used for routing messages to NetView.

These options are ignored when specified for a console with authority level of INFO.

All message and DOM delivery options are ignored, when the console is activated with MSGDLVRY=NONE.

MCSOPMSG Macro

MCSOPMSG is used to retrieve messages queued to a console. This macro is new for VSE and is compatible with the form provided on MVS. All parameters and options of the MVS version are supported on VSE.

MCSOPMSG must be invoked in supervisor state (any PSW-key), primary ASC-mode and 31-bit addressing mode, and is not supported from ICCF interactive partitions. Contrary to the MVS version, it cannot be invoked in AR mode nor by any task other than the one that activated the console (via MCSOPER).

Depending on the option specified for REQUEST, and on the macro format used (MF parameter), other parameters may be required, optional or invalid. The applicability for REQUEST options is summarized in Figure 14 on page 62. For the dependency on the macro format, refer to the description of the MF parameter.

Registers 0, 1, 14 and 15 are used by the macro and must be saved, if necessary, prior to macro invocation. Except where noted otherwise, registers 2 to 12 may be used for register notation.

```
[name]  MCSOPMSG  [REQUEST=GETMSG|RESUME}]
                  [,CONSID={address|(gr)}]
                  [,CMDRESP={NO|YES}]
                  [,CART={address|(gr)}]
                  [,MASK={address|(gr)}]
                  [,RTNCODE={address|(gr)}]
                  [,RSNCODE={address|(gr)}]
                  [,MF={S|(L,name[,{string|0D}])|
                  ({M|E},{address|(gr)}[,{
                  COMPLETE|NOCHECK}])}]
```

REQUEST={**GETMSG**|**RESUME**}

Specifies the MCSOPMSG function requested:

GETMSG Requests delivery of a queued message or command response.

RESUME Requests to resume message queuing to this console.

CONSID={*address*|(*gr*)}

Specifies the console ID that was returned by the MCSOPER macro.

CMDRESP={**NO**|**YES**}

Specifies whether the first message queued for this console is to be delivered (NO), or the first command response matching the criteria indicated by the CART and MASK parameters (YES). A message is considered a command response when it was issued via WTO or WTOR with the option MCSFLAG=RESP, or with a descriptor code 5. The specification CMDRESP=YES is only valid for REQUEST=GETMSG and when MSGDLVRY=SEARCH was specified on MCSOPER.

CART={*address*|(*gr*)}

Specifies the address of an 8-byte area containing a command and response token to be used for searching the next message to be delivered. This parameter is only valid in connection with CMDRESP=YES.

MASK={*address*|(*gr*)}

This parameter is only valid in connection with CART. It specifies the address of an 8-byte area containing a mask to be applied to the specified CART value. Only the bit positions that are set in the mask are taken into account, when comparing the specified CART with the CART associated with a message.

MF=**S** Specifies the standard form of the macro. It generates an in-line parameter list, with the specified parameters and with defaults for omitted parameters, and code to invoke the requested service. Full checking for valid parameter combinations is included.

MF=(**L**,*name*[,{*string*|0D}])

Specifies the list form of the macro. It generates a parameter list in a storage area defined by *name*. The optional *string* parameter specifies a string of 1 to 60 characters, to be used as the argument of the generated DS instruction. No other macro parameters may be specified with this option.

MF=(**M**,{*address*|(*gr*)}[,{ **COMPLETE**|**NOCHECK**}])

Specifies the modify form of the macro. It generates code to store specified parameters in a parameter list at the specified address. Checking for required parameters and supply of defaults is included (COMPLETE) or omitted (NOCHECK). When COMPLETE is specified, the parameter list is cleared before filling. Otherwise, only the specified parameters are updated.

MF=(**E**,{*address*|(*gr*)}[,{ **COMPLETE**|**NOCHECK**}])

Specifies the execute form of the macro. It generates code to store specified parameters in a parameter list at the specified address and to invoke the requested service. Checking for required parameters and supply of defaults is included (COMPLETE) or omitted (NOCHECK). When COMPLETE is specified, the parameter list is cleared before filling. Otherwise, only the specified parameters are updated.

Figure 14. Applicability of MCSOPMSG Parameters for REQUEST Options

Parameters	REQUEST=GETMSG	REQUEST=RESUME
CONSID	required	required
CMDRESP	optional	invalid
CART	optional	invalid
MASK	optional	invalid
RTNCODE	optional	optional
RSNCODE	optional	optional

A successful GETMSG request returns the 31-bit address of a Message Data Block (MDB) in register 1, qualified by an ALET in AR register 1 (currently always 0). The MDB structure is described in “Message Data Block” on page 63.

Return and reason codes:

00 00 Successful completion. For REQUEST=GETMSG, reason code 00 also indicates that no more messages nor DOMs are currently queued for this console.

00 01 REQUEST=GETMSG completed successfully, and at least one more message is queued for this console.

00 02 REQUEST=GETMSG completed successfully, and at least one DOM is queued for this console.

00 03 REQUEST=GETMSG completed successfully, and at least one message and one DOM are queued for this console.

04 00 Console was not suspended (only applicable for REQUEST=RESUME).

08 00 No message available for the specified REQUEST=GETMSG search criteria (if any), and no more messages nor DOMs are currently queued for this console.

08 01 No message available for the specified REQUEST=GETMSG search criteria, but there are is at least one other message queued for this console.

08 02 No message available for the specified REQUEST=GETMSG search criteria, but there are is at least one DOM queued for this console.

08 03 No message available for the specified REQUEST=GETMSG search criteria, but there are is at least one message and one DOM queued for this console.

0C 00 Console is suspended (applicable only for REQUEST=GETMSG). REQUEST=RESUME must be issued before messages can be retrieved again for this console.

10 00 Invalid input: The requested function is invalid (not GETMSG or RESUME).

10 01 Invalid console ID: The console is not active.

10 02 Invalid console ID: The console was not activated by this task.

14 00 The address of the parameter list or of an input parameter is invalid.

14 01 The parameter list contains an incorrect macro acronym or version indicator.

14 04 The console was activated with MSGDLVRY=NONE, or with MSGDLVRY=FIFO but CMDRESP=YES was specified.

14 05 The caller is not in supervisor state or not in primary ASC mode or not in 31-bit addressing mode.

18 00 Service routine failure.

Cancel codes: None.

Message Data Block

The MDB lay-out is described by the mapping macro IEAVM105, which is ported from MVS and adapted for use under VSE.

Embedded in the MDB are three types of objects:

- A **general object**, containing general information about the message. Its lay-out is the same as for MVS.

  MDBGLEN Length of general object.

  MDBGTYPE Type for general object = X'0001'.

  MDBGMID Message ID, consisting of a 1-byte system ID (currently 00) and a 3-byte sequence number assigned in wrap-around mode. Connected messages have the same message ID. For a DOM request, this is the message ID of the message(s) to be deleted. For a message returned by a redisplay command, this is the message ID of the original message, if the message was retrieved from the Router queue, and all 0's otherwise.

MDBGTIMH 8-byte time stamp in EBCDIC format 'HH:MM:SS'. For a redisplay response, this is the time stamp of the original message.

MDBGTIMT 3-byte extension of the time stamp in EBCDIC format '.TH', where TH are hundredths of one second.

MDBGDSTP 7-byte date stamp in EBCDIC format 'YYYYDDD'. For a redisplay response, this is the date stamp of the original message.

MDBGMFLG Message flags:

MDBGDOM All messages with matching message ID are to be deleted.

MDBGALRM Sound warning alarm for this message.

MDBGHOLD Message is to be held until DOMed or explicitly deleted by the operator.

MDBGFGPA Foreground presentation attributes, matching 3270 extended data stream definitions.

MDBGBGPA Background presentation attributes.

MDBGOSNM Originating system name. This is the same as the node name used by POWER, when defined.

MDBGJBNM POWER job name of the originating job. This field is blank if the message is not job related (the message prefix contains 'AR').

- A **control object**, containing operating system dependent information about the message. Following fields are used by VSE:

  MDBCLEN Length of control object.

  MDBCTYPE Type for control object = X'0002'.

  MDBCVER VSE version level (currently 1).

  MDBCPNAM Control program name = 'VSE '.

  MDBCCNID 4-byte ID of the console to which this message is directed, as specified by WTO/WTOR (CONSID parameter) or implicitly set for SYSLOG I/O (AR responses only). May be zero if CONSID was not specified nor implicitly set, and differ otherwise from the ID of the console that is receiving the MDB, since the message may also have been routed by other criteria.

  For a redisplay response, this is the console that issued the redisplay command.

  MDBCCNNM 8-byte name of the console to which this message is directed. May be blank if CONSNAME was not specified nor implicitly set, and differ otherwise from the name of the console that is receiving the MDB, since the message may also have been routed by other criteria.

  For a redisplay response, this field identifies the name of the destination console of the original message, if any, or of the origin of a console input.

MDBCERC 16-byte field with 128 bit positions, each corresponding to a routing code. Bits are set for the routing codes specified by WTO/WTOR or implicitly set for SYSLOG I/O. For redisplay responses, these are the routing codes of the original message.

MDBCCART CART as specified by WTO/WTOR or implicitly set for SYSLOG I/O (AR responses only). For redisplay responses, this is the CART specified with the redisplay command.

MDBCDESC 2-byte field with 16 bit positions, each corresponding to a descriptor code. Bits are set for the routing codes specified by WTO/WTOR or implicitly set for SYSLOG I/O. For redisplay responses, these are the descriptor codes of the original message.

MDBCMLVL Message level (see description of MCSOPER macro). For redisplay responses, this is the message level of the original message.

MDBCATTR Message attribute flags:

- **MDBCCMDR** Message is a command response. This flag is set when MCSFLAG=RESP or descriptor code 5 was specified by WTO/WTOR, and for all AR messages issued via SYSLOG I/O. For redisplay responses, this flag refers to the original message.
- **MDBCRDIS** Message is a redisplay response.
- **MDBCCRCR** Message is a Console Router command response.
- **MDBCREND** Message is the last message of a redisplay response.
- **MDBCMCON** Incomplete message (no end of data present).

MDBDOMFL DOM options, as specified on a DOM macro or by a system generated DOM request:

- **MDBDMSGI** DOM by message ID. This is the only option currently supported by VSE and is therefore always on for DOM requests.

MDBCTOFF Offset in the line text of the beginning of the original message line.

MDBCLCNT Number of lines (i.e. of text objects) in this MDB.

MDBCRET Feed-back information from command processors:

- **MDBCRC** Return code of redisplay command. When non-zero, this message is an error response.
- **MDBCRS** Reason code for given return code.

The meaning of these codes for the various command processors is show in Figure 15.

Figure 15. Return and Reason Codes from Command Processors

RC	RS	Console Router	Attention Routine	HCF Processor
0	0	OK	OK	OK
0	1	not used	not used	OK, response is caused by RED E,xL
4	0	not used	last message of command response	last message of command response
4	1	not used	not used	last message of command response, caused by RED E,xL
4	2	not used	not used	last message of command response, top of HC File
4	3	not used	not used	last message of command response, bottom of HC File
8	0	not used	end of command response (no message), command was processed by AR	not used
8	1	not used	end of command response (no message), command was passed to a sub-system	not used
8	>8	MDB contains an error message (English version). The reason code is the index of the message in $IJBxDEF.	not used	MDB contains an error message (English version). The reason code is the index of the message in $IJBxDEF.

MDBCREDI Unique redisplay information:

MDBCERRO Offset of error field in redisplay command, when applicable for given return code.

MDBCFILT Effective filter.

MDBCSFIL Effective subfilter.

- Zero or more **text objects**, containing control information and text of each message line. Text objects have the same layout as for MVS:

  MDBTLEN Length of text object.

  MDBTTYPE Type for text object = X'0004'.

  MDBTLNT1 Line type flags, byte 1. Line type as specified by WTO/WTOR or implicitly set.

  MDBTLNT2 Line type flags, byte 2. Indicates if the following presentation attributes override those given in the general object.

  MDBTMTPA Overriding line presentation attributes (correspond to 3270 extended data stream definitions).

MDBTMSGT Line text, including the system generated prefix (if any). The length of the line text can be calculated from MDBTLEN minus the constant length of the preceding control information.

Some messages are delivered in more than one MDB, each requiring a separate GETMSG request. In such a case, only the last text object of the last MDB contains a 'last message line' indicator.

MGCRE Macro

The MGCRE macro is used to enter a system command or a reply to a pending message. from a console session. This macro is new for VSE and supports a compatible subset of the form provided on MVS, except for the VSE-unique CMDFLAG=EXPLAIN option. Following parameters or options of the MVS version are not supported on VSE:

TOKEN
CMDFLAG=COMMTASK, NOHCPY, SUBSYS

MGCRE must be invoked in supervisor state (any PSW-key), primary ASC-mode and 24- or 31-bit addressing mode, and is not supported from ICCF interactive partitions.

Registers 0, 1, 14 and 15 are used by the macro and must be saved, if necessary, prior to macro invocation. Except where noted otherwise, registers 2 to 12 may be used for register notation.

```
[name]  MGCRE   TEXT={address|(gr)}
                [,CONSID={address|(gr)}]
                [,CONSNAME={address|(gr)}]
                [,CMDFLAG=(keyword[,...])]
                [,CART={address|(gr)}]
                [,UTOKEN={address|(gr)}]
                ,MF={L|(E,{address|(gr)})}
```

TEXT={*address*|(*gr*)}
: Specifies the address of an area containing the command or reply. The first 2 bytes contain the length, immediately followed by up to 126 bytes of command/reply text. The input is rejected if the length value is 0 (except for CMDFLAG=EXPLAIN, see below) or larger than 126, or if the text consists of all blanks.

  Any input starting with a numeric character is interpreted as a reply. In this case, there must be a leading token of 1 to 4 numeric characters that can be interpreted as a reply ID.

CONSID={*address*|(*gr*)}
: Specifies the 4-byte console ID that was returned by the MCSOPER macro. CONSID is mutually exclusive with CONSNAME (see below) and can be used only when console input is to be related to an active console session, that is to receive response messages, if any.

CONSNAME={*address*|(*gr*)}
: Specifies the address of a 8-byte field containing the console name to be associated with the input. The name must be a 4- to 8-byte string of characters with values higher than x'40' and lower than x'FF', left justified and padded with blanks. CONSNAME is mutually exclusive with CONSID and does not depend on a session for the named console to be active.

  If no session was established and VSE security is active, the authority level associated with the input is determined by the UTOKEN parameter (see below). If VSE security is not active, MASTER authority is assumed.

MGCRE with CONSNAME is the recommended replacement for the current SVC 30 interface, if the additional function provided for active consoles is not required.

CMDFLAG=(*keyword*[,...])
Specifies one or more keywords requesting some special handling for this command. The following specifications are supported by VSE:

EXPLAIN The input text is interpreted as the key of a record of the Online Explanation File, to be loaded into virtual storage in response to this command. If the text length is 0, the only effect of the command is that any previously allocated explanation area is freed. This option is only valid in connection with an active console session.

CART={*address*|(*gr*)}
Specifies the address of an 8-byte field containing a command/response correlation token. This token is passed, along with the command text and console identification, to the command processor. When specified on the response WTO, it allows the issuing program to correlate the response with the original command.

UTOKEN={*address*|(*gr*)}
Specifies the address of an area containing Access Control data in the format described by the DTSJPL macro. When register notation is used, the register is assumed to contain the address of a pointer to the area. This parameter must be used whenever input authorization and logging are to be associated with a userid other than the console name. When VSE security is active, this parameter is required for all input submitted without a console session and requiring MASTER authority.

MF={**L**|(**E**,{*address*|(*gr*)})}
Requests a macro expansion in the list (L) or execute (E) format (this macro has no standard format). For the execute format also specifies the address of a MGCRE parameter list generated by the list format.

Return and reason codes in registers 15 and 0:

00 00 Processing completed successfully, input is accepted.
00 01 Input is accepted, but was recognized as sensitive, like a Job Control // ID statement possibly containing a password. The input text is logged with an overlay '(PARAMETERS SUPPRESSED)' and the modified text is returned in the CSA, allowing consoles to echo it instead of the original input text.
04 00 Console with specified name is already active.
08 01 Command not accepted because a previous command from the same console and for the same command processor is not yet completed.
08 02 Invalid reply ID. Either no message is pending for the specified reply ID or the console is not authorized to reply to the pending message.
08 03 The console is not authorized for the specified command.
08 04 The Attention command processor is not active.
08 05 The Redisplay command processor is not active.
08 06 Input from system console is inhibited due to REMOTE operating mode established via an OPERATE command.
08 07 Redisplay mode is already active for another user. This condition is only possible for consoles that operate on behalf of multiple users by means of the UTOKEN parameter.

08 08 The input was rejected by an exit routine.
08 09 REDISPLAY C or E is rejected because redisplay mode is not active.
08 0A Console input (e.g. REDISPLAY command) rejected due to shortage of 24-bit system GETVIS storage.
08 10 Command not accepted because the specified console is suspended.
08 11 The specified command (e.g. REDISPLAY or EXPLAIN) is not supported for an inactive console (only possible when CONSNAME was specified).
08 12 No dummy console is available to process input for an inactive console (only possible when CONSNAME was specified).
0C 00 The input text is all blanks.
0C 01 The input length is 0 or larger than 126 (not EXPLAIN), or different from 0 and 12 for EXPLAIN requests.
0C 02 The input starts with a numeric character, but there is no leading token of 1 to 4 numeric characters that can be interpreted as a reply ID.
10 01 Invalid console ID: The console is not active.
10 02 Invalid console ID: The console was not activated by this task.
10 08 Invalid console name: The name is shorter than 4 characters or contains invalid characters.
14 00 Service routine failure.

For return codes 08 and 0C with reason code 00-0F, the address of a related error message is also provided in fields MCSCEMSG (English version) and MCSCNMSG (NLS version) of the CSA. These addresses are unpredictable for other return/reason code combinations.

Cancel codes:

21 Caller is not in supervisor state, or one or more input parameters, other than the special cases covered by return codes, are invalid or not supported on VSE,
25 One or more of the specified addresses are invalid.

EXPLAIN Interface

Contrary to other system commands submitted via MGCRE, the response to an EXPLAIN request is not returned in an MDB, but rather as an address and length made available in CSA-fields MCSCEXPT and MCSCEXLN. Also, request completion is indicated by posting the ALERTECB, rather than the MSGECB, along with CSA-flag MCSCAFL2.MCSCA2EX and one of the following return/reason codes in CSA-fields MCSCEXRC/MCSCEXRS:

00 00 Successful, with a perfect match between input and output keys.
04 00 Successful, with a partial match between input and output keys.
08 00 No match found for input key.
08 04 No GETVIS space obtained for output area.
0C xx EXPLAIN File access failure, with VSAM error code xx.
10 00 EXPLAIN support is not active.

On successful completion (MCSCEXRC is 0 or 4), MCSCEXPT points to a virtual storage area containing explanation text for the input key. If no exact match is found for the requested key, a closest match based on knowledge about the key contents (e.g. a message code) may be returned. The application has to decide, based on the actual key of the data, whether or not to use the returned data.

The key to be supplied as input must be 12 bytes long and is structured as follows:

- Language character (1 byte): 'E' for English, 'G' for German, 'J' for Japanese (KANJI) or 'S' for Spanish. In VSE explanation data is provided only for one of above languages, and the corresponding language character can be obtained from SYSCOM field IJBNLSIC. To allow for a future multilingual extension, the language character is nevertheless treated as application input.
- A 10-byte mixed-case character string, left aligned and padded by blanks, that uniquely identifies the item (e.g. a message number) for which explanation is requested.
- A 1-byte binary sequence number, that identifies a single record of explanation data. The sequence number is x'00' for the first record and is incremented by x'01' for each continuation record, whenever applicable.

Explanation data consist of the key field followed by a sequence of variable length lines of formatted explanation text in a compressed format, as shown in Figure 16. For the first (or only) record, the 10-byte item string in the key is not necessarily identical with the input, since the EXPLAIN service also looks for a closest match. In such cases, the output key of the first record must be used to build the input key for continuation records, when applicable, since a full match is required when the sequence number is larger than x'00'.

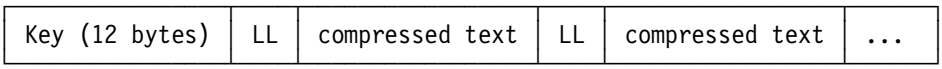


Figure 16. Structure of Explanation Data

The length of explanation lines may vary between 0 (blank line) and a maximum value close to 80 decimal, whereas special values are used to indicate the logical end of the record: x'FF' when continuation records are present, otherwise x'FE' for the last (or only) record. The console application has to check for one these values, to find the end of the record, since the length returned in MCSCEXLN may be larger than the actual record length. To retrieve the next continuation record, when applicable, the application has to build a new input key from the current output key, by incrementing the continuation suffix by x'01'.

For presenting the data, console applications must invoke the expansion service described in VSE/ESA System Macro Reference. The dictionary used to compress and expand explanation text in the appropriate national language is shipped in load module $IJBxDCT, where 'x' stands for an NLS identification character provided in SYSCOM.IJBNLSIC. A symbol size of 9 bits is used. The module must reside in a page-aligned area, and for this reason it cannot be loaded in the SVA. Applications must load it in the partition, e.g. via CDLOAD with PAGE=YES. By convention, the load point and entry points of IJBxDCT correspond respectively to the addresses of the compression and expansion dictionaries, and may be obtained via the LOAD or CDLOAD services.

Character Sets: Two single-byte character sets are defined, one for English, German and Spanish, and one for Japanese. The two character sets are chosen to contain a common set of language-neutral special characters, that are typically not translated.

The E/G/S character set is defined as 01114 - {¤, ´, .., °, §, ¦}, which is a subset of character set 697 common to these and several other languages.

The Japanese SBCS is defined as 1172. Together with DBCS 370 (Kanji), it builds the complete character set for Japanese.

Code Pages: Displaying EXPLAIN text on terminals and PC's configured for the installed VSE language requires a match between the code page of the display device and the code page used for the EXPLAIN File. These code pages are:

- English: 37
- German: 273
- Spanish: 284
- Japanese: 290 (SB) + 300 (DB)

Highlighting Control: A single special code x'79' is used within explanation text to mark the beginning and end of strings, that are to appear highlighted when displayed. This code is contained in all supported code pages and corresponds in all of them to character . It is assumed, that this character is never used within message explanation text.

Highlighting control applies separately to each single line. Column 1 is reserved for initial highlight setting and may only contain x'79' (highlighting on) or blank (highlighting off). Any further occurrence of x'79' changes highlighting status in a flip/flop manner. End of line implies highlighting off.

Whenever the control code appears, it replaces a blank character in the original text and may be used for the attribute byte, appearing as a blank, in the presented text. Therefore, no further editing of the input text (e.g. for blank suppression) is required.

For Japanese, SI/SO brackets may not span multiple lines, and keyword identification and highlighting control are only recognized outside SI/SO brackets.

Load Utility: Explanations are loaded into the EXPLAIN file by means of utility IESMSGS. This utility is shipped with VSE and may be used to add explanations for messages generated by products or applications, that are not covered in the original file. The EXPLAIN file must not be in use (EXPLAIN must be OFF) while IESMSGS is executing. After IESMSGS completes, EXPLAIN may be switched ON and added explanations are then immediately available.

IESMSGS is invoked as shown in the following example.

```
// JOB LOADEXPL
// OPTION PARTDUMP,NOSYSDMP
// DLBL NEWMSGS,'VSE.MESSAGES.ONLINE',,VSAM,CAT=IJSYSCT
// EXEC IESMSGS,SIZE=60K,PARM='EU'
 (input data)
/*
/&
```

The IESMSGS input parameter PARM specifies a language character (E for English, G for German, S for Spanish, J for Japanese) and a processing option (I for Initialize, U for Update, C for Check). When omitted, the language of the running system (from SYSCOM field IJBNLSIC) and the processing option U are assumed. Therefore, PARM may be omitted when adding explanations in the customer environment.

The explanations are supplied as SYSIPT data in the format shown above.

Keywords are identified by control character in column 1, followed by a non-blank character other than , They may be between 4 and 10 characters long and may only contain alphabetic (a-z, A-Z) and numeric (0-9) characters. Keywords are used to build the VSAM key described in the previous section. Lower case alphabetic characters are translated to upper case, assuming single byte characters only for Japanese, and the result is padded with blanks up to the maximum length of 10 characters. Keywords shorter than 4 characters or containing invalid characters are skipped. For keywords longer than 10 characters, only the first 10 characters are used for the key. Keywords are also considered part of the explanation text, and are included as such without translation or truncation.

Explanations for the same keyword, with respect to the first 10 characters, are chained together, when immediately following each other. Otherwise, the last explanation is taken. When an explanation for the same keyword already exists, it is replaced (this is only applicable when the processing option is U). The providers of additional explanations are responsible for avoiding accidental duplications with existing data, e.g. by means of unique provider prefixes.

Other explanation text may be in any format, except that

- column 1 may only contain blank or ,
- in column 1 must be followed by blank or .

Explanation text is displayed in the supplied input format, except that strings delimited by are highlighted, whereas itself is shown as blank.

IESMSGS assumes that supplied explanation keywords and texts are coded according to the code page for the processed language, and that SO/SI pairs do not span multiple lines. Warnings are issued when invalid characters or SO/SI pairing errors are detected. No transformations other than those mentioned above are performed.

Adherence to IBM-supplied explanations, mostly based on message definition tags, is recommended for consistency and for compression efficiency.

MODCTB

The MODCTB macro (see also SVC 98) modifies a PUB2 table entry for a 3800 printer device, provides selective services for tape libraries and acts as interface for encryption parameter exchange.

```
[name]   MODCTB    ID=PUB2
                  ,AREA={name1|(r1)}
                  ,LEN={name2|(r2)}
                  ,DISP={name3|(r3)}
                  ,{SEL={name4|(r4)}|SEP={name4|(r4)}}
                  [,PID={name5|(r5)}]
                  [,MFG={name6|(r6)}]

[name]   MODCTB    ID={MNTLIST|MNTNEXT}
                  ,AREA={name1|(r1)}
                  ,LEN={name2|(r2)}
                  ,{SEL={name3|(r3)}|SEP={name3|(r3)}}
                  [,PID={name4|(r4)}]
                  [,MFG={name5|(r5)}]

[name]   MODCTB    ID=ATLSETUP
                  ,AREA={name1|(r1)}
                  ,LEN={name2|(r2)}
                  [,MFG={name3|(r3)}]

[name]   MODCTB    ID=KEKL
                  ,AREA={name1|(r1)}
                  ,LEN={name2|(r2)}
                  ,FUNC={SET|CLEAR|QUERY}
                  [,MFG={name3|(r3)}]
```

Explanation of Parameters

The parameters of the macro invocation have the following overall meaning.

ID Defines the kind of service requested from the MODCTB macro.

AREA Address of the user area where the user supplied information for the requested service is stored. May serve as input and output area of the service.

LEN Length of the user area.

DISP Used only in combination with the PUB2 service. Specifies the Offset from which on the PUB2 Control block is modified with the user information. Defaults to 0 if omitted.

FUNC Used only in combination with the KEKL service. Specifies the kind of KEKL service requested.

- SET associate KEKLs with the device specified.
- CLEAR clear KEKL information related to the device. This does not imply the clearing of KEKL information in the control unit or from a mounted cartridge.

- QUERY returns *in the following order* either the KEKL information from a previous -still active- SET function or from the currently mounted cartridge IF encrypted.

SEL Address of a halfword containing the logical unit (same as in CCB) assigned to the physical device for the partition specified by PID.

SEP Address of a halfword containing the physical unit (cuu). Either the SEL or the SEP operand must be specified.

PID Address of a halfword containing the PIK of the partition the logical unit belongs to. Default is the PIK of the partition of the issuing program. If SEP is specified, PID is ignored.

MFG Address of a 16-byte work area where the parameter list is to be generated by the MODCTB macro (refer to the MFG description for the EXTRACT macro).

Note: The register usage and the layout of the parameter list (see “SVC 98 (X'62' - EXTRACT/MODCTB)” on page 277) are the same as for the EXTRACT macro.

Output in General Output information can be stored in the user area provided by the Area operand and in a return code given in register R15***:*** Register 15 contains one of the following return codes.

Possible Cancel Exits An illegal SVC is forced when:

- ID specification is invalid.
- PIK is specified and the user does not have key 0.
- SEP is specified and the user does not have key 0.

Description of PUB2 service The PUB2 information of the specified device is modified with the user information supplied in the area addressed by the AREA= parameter. It is inserted into the PUB2 from the displacement onward that was specified using the DISP= parameter in the length specified by the LEN= parameter. The return code in register R15 can have one of the following values.

0 (X'00')	The service processed successfully.
4 (X'04')	The specified PIK is invalid for this supervisor.
8 (X'08')	The specified logical or physical unit does not exist.
12 (X'0C')	The logical unit specified in SEL is not assigned or it is assigned IGNore.
16 (X'10')	Something is wrong with the point at which the user data shall be inserted in the PUB2. The length specified is zero, or the DISP specification exceeds the length of the PUB2 table entry for the specified device, or the range defined by DISP and LEN exceeds the range of the PUB2 table entry for the specified device.

Description of MNTLIST service This service was written for the support of automatic tape libraries. It is intended to attach a list of volume serial numbers to the PUB extension of the device which is kept in system GETVIS. The device must be mount reserved for the caller or the caller must have the task ID of the attention routine task. The list of volume serial numbers is taken from the user supplied AREA, which is mapped by the following DSECT:

DEC	HEX	Label	Description
0 – 7	0 – 7	MNILIBNM	name of library
8 – 15	8 – F		first entry consisting of
8 – 13	8 – D	MNIVOLNM	volume serial number
14 – 14	E – E	MNIFLAG	flag byte (reserved, must be 0)
15 – 15	F – F	MNIVOLRW	read write information.

Figure 17. Mapping of input area for MODCTB ID=MNTLIST service

The first 8 bytes contain the library name of the library that the device is contained in. This name is returned any time the MNTNEXT service is invoked to retrieve the next volume serial number from the list.Then follows any number of eight byte entries, up to the end of the area, which is specified by the LEN= operand. Each eigth byte entry contains a volume serial number, a flag which must be set to zero now to allow for future extensions and a read write information byte, which can either be set to "R" or "W".
The service is intended as an interface between Job Control (which sets up the list of volume serial numbers to be mounted) and Basic Access Method, which does the actual tape handling. It does not do any authority checking, but unintended users of the service may not be informed about changes in the processing.
The service gives one of the following return codes in register R15:

0 (X'00')	The service processed successfully. New list is appended to devices PUB extension.
8 (X'08')	The flag byte of an entry is not set to zero or the read/write information contained a value other than "R" or "W".
12 (X'0C')	The service can not obtain sufficient GETVIS for the list
16 (X'10')	The specified length is smaller than 8, indicating that not even the header information with the library name was supplied or the length is not divisible by 8.
20 (X'14')	The specified device can not be found.
24 (X'18')	The device is not a tape in a library.
28 (X'1C')	The device is not mount reserved for the callers partition.

Description of MNTNEXT service This service is intended to retrieve the next volume serial number from the list of volume serial numbers attached to a specified device. The caller specifies a 16 byte area and the service will insert the 8 byte library name in the first 8 bytes and the entry of the next volume serial number in the list in the second 8 bytes. The service then increments a pointer in the devices library information to keep track of this operation. The following return codes may be provided in register R15:

0 (X'00')	The service processed successfully, the next volume serial number is returned in the user area.
4 (X'04')	The device has no volid list attached.
16 (X'10')	The specified length was smaller than 16.
24 (X'18')	The device is not a tape in a library.
28 (X'1C')	The device is not mount reserved for the callers partition.

Description of ATLSETUP service Using this service a setup program can ascribe a logical library name to the devices of a physical library. Since the logical library name is something that cannot be sensed from the device itself, the system has to be notified by this service. The user input is given in an area which is mapped by the following DSECT:

DEC	HEX	Label	Description
0 – 3	0 – 3	ATLSVERS	version identification
4 – 7	4 – 7	ATLSCUUN	number of CUUs in appended list
8 – 15	8 – F	ATLSLIBN	logical name of library
16 – 19	10 – 13	ATLSCUUV	first CUU in appended list
20	14	ATLSMINL	minimum length of user area

Figure 18. Mapping of input area for MODCTB ID=ATLSETUP service

In the field ATLSLIBN the caller specifies a logical library name (left adjusted, character format, padded with trailing blanks if any). The field ATLSCUUN contains the number of CUUs that are appended behind the logical library name, starting with the field ATLSCUUV. The version indication should be set to zero for future extensions of the service. The library name is entered in the library information of the devices mentioned in the CUU list. The following return codes may be given by the service.

0 (X'00')	The service processed successfully.
4 (X'04')	One of the devices had no volid list attached.
8 (X'08')	One of the CUUs in the list does not exist in the PUB.
12 (X'0C')	Device cannot be a tape in a tape library
16 (X'10')	The specified length was smaller than minimum.
20 (X'14')	Either the library runs VM controlled and the device is found to be operational and should have been ascribed a logical library name before it was attached, or the library is native VSE controlled and the device is not operational.
24 (X'18')	The library is VSE controlled and the CUUs in the list belong to different physical libraries.
28 (X'1C')	The specified value for ATLSVERS is not supported.

Description of KEKL service With this service Key Encryption Key Labels (KEKL) can be set, queried or cleared. KEKL services can only be requested against encryption capable tape devices. The user input is given in an area which is mapped by the following DSECT:

DEC	HEX	Label	Description
0 – 1	0 – 1	ENCRCUU	CUU to be processed
2 – 2	2 – 2	ENCRKEKN	number of KEKL to process
3 – 66	3 – 42	ENCRKEK1	KEKL1
67 – 67	43 – 43	ENCRKMH1	encoding mechanism for kekl1
68 – 131	44 – 83	ENCRKEK2	KEKL2
132 – 132	84 – 84	ENCRKMH2	encoding mechanism for kekl2

Figure 19. Mapping of input area for MODCTB ID=KEKL service

The CUU addresses the tape device for which the KEKLS are to be set, reset or queried. Even if for the "SET" function the number of KEKLs to be processed can be one, z/VSE will always set two KEKLs. Which in case of only one KEKL specified will result in KEKL2 equals KEKL1. For the "CLEAR" function only CUU has to be specified. The following return codes may be provided in register R15 by this service:

0 (X'00')	The service processed successfully.

4 (X'04')	No encryption information allocated on a clear request.
12 (X'0C')	The CUU is not encryption capable.
16 (X'10')	Invalid parameter. Either the parameter length is incorrect, the number of KEKLs is incorrect, the encoding mechanism is incorect or there are no KEKLs to be returned on a query request.
20 (X'14')	Service was requested against a device which is not and/or not uniquely assigned to the requestor.
24 (X'18')	Service was requested against a device for which the encryption mode is not active.

MODESET — Change System Status (SVC Generation)

The MODESET macro is used to change system status by altering the PSW key and/or PSW problem state indicator.

The requirements for the caller are:

Minimum authorization:	Problem state with any PSW key; for MODE=SUP the program must be identified as Job Control, a subsystem or a vendor-exit or must run with PSWkey zero.
AMODE:	24- or 31-bit
ASC mode:	Primary
Control Parameters:	Must be in primary address space

Input Register Information

Before issuing the MODESET macro, the caller does not have to place any information into any register unless using it in register notation for a particular parameter, or using it as a base register.

Output Register Information

When control returns to the caller, the GPRs contain:

Register	Contents
0	Reason code
1	Used as a work register by the system
2 - 13	Unchanged
14	Used as a work register by the system
15	Return code

When control returns to the caller, the ARs are unchanged.

Some callers depend on register contents remaining the same before and after issuing a service. If the system changes the contents of registers on which the caller depends, the caller must save them before issuing the service, and restore them after the system returns control.

MODESET-Standard Form

The standard form of the MODESET macro that generates an SVC is written as follows:

```
[name]  MODESET
                [KEY={ZERO|NZERO|USER}]
                [,MODE={PROB|SUP}]
                [,RELATED=value]
```

Note: KEY is required if MODE is not specified.

Note: MODE is required if KEY is not specified.

The parameters are explained as follows:

KEY=ZERO
KEY=NZERO
KEY=USER
Specifies that the PSW key (bits 8-11) is to be either set to zero (ZERO) or set to the storage key of the SVC requestor (NZERO, USER).

Note: The argument USER is introduced to support the current VSE MODESET.

,MODE=PROB
,MODE=SUP
Specifies that the PSW problem state indicator (bit 15) is to be either turned on (PROB) or turned off (SUP). If the MODESET operation completes with a problem state PSW, only the key specified by the problem state PSW will be authorized (if PKM in CR 3 is supported).

,RELATED=*value*
Specifies information used to self-document macros by "relating" functions or services to corresponding functions or services. The format and contents of the information specified are at the discretion of the user, and may be any valid coding values.

MODESET—List Form

The list form of the MODESET macro that generates an SVC is written as follows:

```
[name]  MODESET
                [KEY={ZERO|NZERO|USER}]
                [,MODE={PROB|SUP}]
                [,RELATED=value]
                ,MF=L
```

Note: KEY is required if MODE is not specified.

Note: MODE is required if KEY is not specified.

The parameters are explained under the standard form of the MODESET macro that generates an SVC, with the following exception:

,MF=L
Specifies the list form of the MODESET macro.

MODESET—Execute Form

The execute form of the MODESET macro that generates an SVC is written as follows:

```
[name]  MODESET
                [KEY={ZERO|NZERO|USER}]
                [,MODE={PROB|SUP}]
                [,RELATED=value]
                ,MF=(E,list addr)
```

Note: KEY is required if MODE is not specified.

Note: MODE is required if KEY is not specified.

The parameters are explained under the standard form of the MODESET macro that generates an SVC, with the following exception:

,MF=(E,*list addr)*

Specifies the execute form of the MODESET macro.

list addr specifies the area that the system used to store the parameters.

Messages and Codes:

The MODESET macro abnormally terminates with cancel code X'21', message 0S04I ILLEGAL SVC, in case of an unauthorized requestor.

When the MODESET macro returns control to your program, GPR 15 contains a hexadecimal return code and GPR 0 contains a hexadecimal reason code.

Figure 20. Return and Reason Codes for the MODESET Macro

Return Code	Reason Code	Meaning and Action
00	00	**Meaning**: The operation was successful. **Action**: None.
04	04	**Meaning**: NOP was requested. **Action**: None.
04	08	**Meaning**: Invalid action codes in register 1, probably caused by using a private macro expansion. **Action**: Use the system macro interface.
04	12	**Meaning**: Invalid mode request in register 1, probably caused by using a private macro expansion. **Action**: Use the system macro interface.
04	16	**Meaning**: Invalid key request in register 1, probably caused by using a private macro expansion. **Action**: Use the system macro interface.

Example:

Change to supervisor mode and key zero (authorized request).

```
MODESET KEY=ZERO,MODE=SUP
```

Migration And Coexistence:

Notes:

1. The expansion of the new MODESET macro is not supported in previous VSE releases. However, the generation of the old macro expansion can be forced via the SPLEVEL macro, as shown in the following example:

```
SPLEVEL SET=3
MODESET KEY=ZERO
```

2. SPLEVEL SET=n, n: any value smaller than 4.
3. MODE keyword is not supported prior to VSE/ESA 2.1.0. Valid arguments for KEY are ZERO and USER.

MODFLD

The MODFLD macro must be used whenever a field or parameter of the system has to be modified or updated.

The MODFLD macro has the following format:

```
[name]    MODFLD   FIELD=name1
                  ,NEWVAL={name2|(r2)|(1)}
                  [,PART={name3|(r3)|(0)}]
                  [,TASK={name4|(r4)|(0)}]
                  [,PU={name5|(r5)|(0)}]
                  [,LU={name6|(r6)|(0)}]
                  [,CUU={name7|(r7)|(0)}]
                  [,RETURN={YES|NO}]
```

FIELD= Identification of the field to be modified. Valid symbols and their interpretation are given below. Valid specification of other parameters and required authorizations are also listed with the field value descriptions.

NEWVAL= The name of a 4-byte field or a register containing the new value to be stored in the specified field. Only the right adjusted significant portion of this argument is used. Register 0 must not be used for register notation.

PART= Name of a 2-byte field or register containing the PIK of the partition to which the specified field belongs. The default value is the requester's PIK.

TASK= Name of a 2-byte field or register containing the TIK of the task to which the specified field belongs. The default value is the requester's TIK (IJBTIK).

PU= Name of a 2-byte field or register containing the physical unit number of the device to be identified for the service.

LU= Name of a 2-byte field or register containing the logical unit number of the device to be identified. The logical unit number must be in the same format as in the CCB, byte 6/7.

CUU= Name of a 2-byte field or register containing the CUU number of the device to be identified.

RETURN= Either set to YES or NO. Default is NO. If RETURN=YES is specified false input specifications will not lead to a cancel but to a return code. The return code indicates, what input parameter was wrong. The following values are possible now (and there may be more in the future!!).

x'80000004' The specified GETFLD FIELD= value is not supported.

x'80000008' The caller is not authorized for this type of request.

x'8000000C' The value supplied by the PART= parameter was unusable.

x'80000010' The value supplied by the TASK= parameter was unusable.

x'80000014' The value supplied in the PU= parameter was unusable.

x'8000001C' A space id given with NEWVAL was unusable for requested service.

x'80000020' An illegal sequence of addressing scope changes was requested.

x'80000024' Illegal combination of partition hold and addressing scope change.

Valid FIELD Symbols and their Interpretation:

ACEEPTR Modify accessor environment element address field
Input R1=0 means that the ACEE of the current task has to be removed, R1<>0 means that the ACEE of the current task has to be connected to this ACEE (R1 contains ACEE pointer)
Values returned if R15 is:

0 R1 points to a removed ACEE (if R1=0 on input), R0 is one of the following:

```
X'01'  ACEE CONNECTED TO/CLEARED FOR TASK
X'02'  ACEE CONNECTED TO/CLEARED FOR PARTITION
x'04'  ACEE CONNECTED TO/CLEARED FOR TASK JPL
x'08'  ACEE CONNECTED TO/CLEARED FOR PARTITON JPL
x'10'  ACEE CONNECTED TO/CLEARED FOR MVS TCB
x'20'  ACEE CONNECTED TO/CLEARED FOR MVS ASXB
```

8 R1 and R0 are 0.

Authorized: SYSTEM, SUBSYSTEMS, JOB CONTROL, Vendors.

ACLOSE VSAM automatic close.
Input: A NEWVAL value of 1 if VSAM automatic close is being started for current task, a value of 0 if VSAM automatic close processing has completed.
Authorized: SYSTEM
Intended users: EOJ
See notes: NOFUTRC, CANCUSE

CELANCH Set pointer to CEL anchor table for current task.
Input NEWVAL parameter containing the new pointer value.
Values returned none. Pointer is changed.
See notes: NOAUTHC, NORC.
Intended users: LE/370

CNCLALL Cancel all tasks flag.
Input: A NEWVAL value of 1, if cancel has to be propagated to all tasks belonging to the same partition as the current task, else 0.
Authorized: SYSTEM.
Intended users: TERMINATOR.
See notes: NOFUTRC, CANCUSE

CNCLCODE Cancel code.
Input: NEWVAL and TASK parameter.
Low order byte of NEWVAL parameter is set as cancel code for specified task.
Authorized: SYSTEM, VTAM and VSE/POWER.
Intended users: EOJ, VTAM and VSE/POWER.
See notes: NOFUTRC, NOTID0, CANCTID, CANCUSE

COPYOWN Copy ownership indication for SYSLOG assignments.
Input NEWVAL and PU, both containing a PUB index value.
Copy all ownership indications from the device identified by PU to the device identified by NEWVAL and delete the ownership indications for the device identified by PU.
Return codes in Register R15:

0 copy carried out successfully
16 A PUB index specified by the caller was illegal.
Authorized: VSE/POWER and JOB CONTROL.
See notes: CANCUSE

DB2 Establish end-of-task appendage for DB2 maintask
Input R0(entry point of DB2 appendage), R1(address of DB2 control block). On entry to the DB2 appendage routine, RF contains the entry point, RA the address of the DB2 control block.

ICCFPP ICCF pseudo partition.
Input: NEWVAL and TASK parameter. A NEWVAL value of 1 if the specified task is assigned to an ICCF interactive partition, else 0.
Authorized: ICCF.
See notes: NORC, CANCUSE, CANCTID, NOTID0

ICCFRO ICCF roll-out
Input: NEWVAL and TASK parameter. A NEWVAL value of 1 if the specified task is assigned to an ICCF interactive partition and is eligible for ICCF roll-out, else 0.
Authorized ICCF.
See notes: NORC, CANCUSE, CANCTID, NOTID0

ICCFSVC ICCF intercept of SVCs
Input: NEWVAL and TASK parameter. A NEWVAL value of 1 if SVCs issued by the specified task are to be intercepted by ICCF, else 0. For tasks other than the current task this service is not a fast SVC.
Authorized: ICCF.
See notes: NORC, CANCUSE, CANCTID, NOTID0

IPWSEGM Power segmentation request.
Input None.
Indicate the segmentation request TIBSEGM in the current task's TIB.
Intended users: POWER

LIBRSERV LIBRARIAN cleanup required at EOT.
Input: NEWVAL and TASK parameter. This service is provided for the LIBRARIAN routines to indicate that the LIBRARIAN clean up routines need to be called at EOT. The flag is set (NEWVAL¬=0) for the current task and the corresponding maintask but reset (NEWVAL=0) for the current task only.
Intended users: LIBRARIAN
See notes: NORC, NOAUTHC, NOTID0, CANCTID

MODE Set current mode set function byte for tape devices.
Input Either PART and LU or PU (pub index) to identify the device and NEWVAL parameter.
Values returned if R15 is:
0 Successful, mode changed
8 device is not a tape
12 logical unit unassigned or assigned ignore.
See notes: NOAUTHR, NOPIK0, CANCPIK, CANCPUB
Intended users: JOB CONTROL

MSECS Milliseconds
Input: Time slice for partition balancing in milliseconds. The specified value in NEWVAL must be within 100 and 10000.
Return codes in R15:
0 Modification completed
8 Specified value not within supported range.
Authorized: SYSTEM and JOB CONTROL.
See notes: CANCUSE

MNTINFO this service is intended to give access to the TLMECBSV pointer defined in the TLMADR DSECT and pointed to by the PBXLBINF pointer in the PUBX. The function is not performed, if the pointer value in NEWVAL is not zero and TLMECBSV is already set, or if the pointer value in NEWVAL is zero and TLMECBSV is not set.
Input It is given a CUU= parameter to identify the device and a new pointer value in the NEWVAL= parameter. Values returned if R15 is:

- 0 The NEWVAL= parameter value has been stored. The device may be in a library and the caller has either been the attention routine task or the device was owned by the caller's partition.
- 4 indicates NEWVAL not zero and TLMECBSV already set.
- 8 indicates NEWVAL zero and TLMECBSV not set.
- 12 indicates device not owned by issuing partition and caller is not attention routine task either.
- 16 indicates an invalid input CUU= parameter.
- 20 indicates the specified CUU has invalid device type for this request.

MNTRLPRA notify system of a release of all permanent mount reservations for the calling partition. The service is identical to the MNTRLTPA service, but the permanent mount reservations are reset together with the temporary mount reservations. Intended users: JOB CONTROL
See notes: CANCUSE.

MNTRLPRM notify system of a release of a permanent and temporary mount reservation for a specified device and for the callers partition. If the caller is the attention routine task, the mount reservation is released unconditionally, otherwise the calling partition must be the owner of the device. Input The device is specified by the CUU parameter. Values returned if R15 is:

- 0 The service processed ok, the temporary mount indication was removed.
- 4 The specified device was in a not mounted state already.
- 8 Service processed ok, but a temporary mount reservation had to be released in addition.
- 12 The specified device was not owned by the callers partition and caller was not AR task either.
- 16 The specified device is not defined.
- 20 The specified device has an invalid device type for this request.

Intended users: JOB CONTROL, AR
See notes: CANCUSE, CANCPIK

MNTRLTMP notify system of a release of a temporary mount reservation for a specified device and for the callers partition. If the caller is the attention routine task, the mount reservation is released unconditionally, otherwise the calling partition must be the owner of the device. Input The device is specified by the CUU parameter. Values returned if R15 is:

- 0 The service processed ok, the temporary mount indication was removed.
- 4 The specified device was in a not mounted state already.
- 8 Temporary mount condition was reset, but the device still is in a permanent mount condition.
- 12 The specified device was not owned by the callers partition and caller was not AR task either.
- 16 The specified device is not defined.

20 The specified device has an invalid device type for this request.

Intended users: JOB CONTROL, AR

See notes: CANCUSE, CANCPIK

MNTRLTPA notify system of a release of all temporary mount reservations for the calling partition. This service is intended to be called in a loop starting with a negative pub index and returning to this service as long as the return code given by the service is not 0. Any return code other than zero indicates that there is a device that something may have to be done for. The device is indicated in the return registers. The same device indication should be returned to the service with the next invocation to complete the cleanup operation before proceeding to the next device.

Input PU parameter specifies the device to start the scan with.

Values returned if R15 is:

0 The system has scanned all devices starting with the specified device up to the end of the PUB. It found no device, for which a mount had been pending or which had a mount reservation for this partition and was operational. For all the devices that would stay reserved permanently after removing the temporary mount reservation, or that were not operational, the temporary mount indication was just reset and scanning continued. There is nothing left to do, the caller should not reinvoke the service now.

4 Register R0 contains the CUU, R1 the pubindex of a device, which has a mount reservation indicated for the current partition, no mount is pending and the device would not stay reserved after removing the temporary reserved indication. The attached list of volume serial numbers has been freed by the service now, but the mount reservation is not yet reset and it is up to the caller to take all action he deems necessary. The pubindex should be returned with the next invocation of the service, which will reset the mount reservation.

16 Similar to return code of 4, but the device had a mount pending in addition. Again the pubindex should be returned with the next invocation of the service to reset the mount reservation.

Intended users: JOB CONTROL

See notes: CANCUSE.

MNTRSFRC notify system of a permanent mount reserve for a partition regardless of an already existing mount reservation for another partition. Input The device is specified by the CUU parameter and the PIK by the PART parameter. Values returned if R15 is:

0 The service processed ok, the device is mount reserved for the specified partition. System has updated device ownership information for the device.

4 The specified device was owned by another partition for some other reason but mount reservation. The old reservation was not changed, service did not complete successfully.

12 There had not been a list of volume serial numbers attached to the device. Service has not completed successfully.

16 The specified device is not defined.

20 The specified device has an invalid device type for this request.

Intended users: AR
See notes: CANCUSE, CANCPIK

MNTRSPRM notify system of a permanent mount reserve. Processing and input similar to MNTRSTMP service. Input The device is specified by the CUU parameter and the PIK by the PART parameter. Values returned if R15 is:

- 0 The service processed ok, the device is mount reserved for the specified or callers partition. System has updated device ownership information for the device.
- 4 The specified device was already reserved temporarily for the partition. It is now permanently reserved in addition.
- 8 The specified device is already reserved permanent for this partition. It may be temporarily reserved in addition.
- 12 The specified device is already reserved for another partition.
- 16 The specified device is not defined.
- 20 The specified device has an invalid device type for this request.

Intended users: JOB CONTROL, AR, RID 4 callers.
See notes: CANCUSE, CANCPIK

MNTRSTMP serves to notify system of a temporary mount. The caller has to specify a device for the mount reserve operation. Reservation is done for the callers PIK. Only the attention routine is permitted to supply a different PIK in the PART parameter. The specified device must have a valid device type (i.e. a tape device), but need not be operational. Input The device is specified by the CUU parameter and the PIK by the PART parameter. Values returned if R15 is:

- 0 The service processed ok, the device is mount reserved for the specified or callers partition. System has updated device ownership information for the device.
- 4 The specified device was already reserved temporarily for the partition. It may be permanently reserved in addition.
- 8 The specified device is already reserved permanent for this partition.
- 12 The specified device is already reserved for another partition.
- 16 The specified device is not defined.
- 20 The specified device has an invalid device type for this request.

Intended users: JOB CONTROL, AR, RID 4 callers.
See notes: CANCUSE, CANCPIK

MOUNTFLG Reserved for volume change.
Input: NEWVAL and PU parameter. A value of 1 if the specified device is to be reserved to allow a volume change, else 0 to indicate that the specified device is to be released.
Return codes in R15:

- 0 No owner of device, flag set/reset.
- 8 Multiple device owner or not owned by requester, flag not set/reset.

Authorized: SYSTEM and JOB CONTROL.
Intended users: JOB CONTROL.
See notes: CANCUSE, CANCPUB

OPENSVA Open routine in SVA.
Input: NEWVAL (and TASK) parameter. A value of 1 if OPEN routines are executing in the SVA for the current task, else 0.
See notes: NOAUTHR, NOTID0, CANCTID, CURRTID

PASCOPE Addressability scope for partition.
Input: NEWVAL parameter with the PIK of a partition, that shall be made addressable for the current task. The new scope is shared/private, if the specified partition is shared/private. If the specified partition is executing real, the new scope is for space R.
This service does not ensure that the partition which is to be made addressable will not disappear while the caller is processing in its storage (unlike MODFLD FIELD=PASCOPE1). On deallocation of the space for a partition all the tasks still working in it will be placed into their original address space again *without further notification*.

Return codes in R15:
0 Successful. Address scope changed.
8 Not successful, the specified partition is inactive
Special cancel condition: Any attempt to leave a scope by this service, which was aquired by the PASCOPE1 service will result in a cancel.
Authorized: SYSTEM, VSE/POWER, OCCF, VTAM.
See notes: CANCUSE, SYSPIK

PASCOPE1 Addressability scope and *HOLD* for partition
Input: NEWVAL parameter with PIK of partition.
This service is meant to permit access to a partition and ensure its presence as long as this access is required. Static and dynamic partitions may be accessed alike. Whenever a task changes its addressability scope to the partition it specifies in NEWVAL, a flag is set for this partition, that it may not be deallocated. The flag is reset, when the caller changes his addressability scope to another partition, or back to his own original partition. The flag is never set for the callers original partition.

Note: It is important, that a scope is only maintained as long as it is required. Having completed work in a partition the scope must be changed back to the issuing partition or to shared space (see SASCOPE1 service).

Return codes in R15:
0 Scope change was successful.
8 Partition to be accessed is inactive.
16 Partition to be accessed is not allocated or the attempt to issue a hold request for the partition failed (partition in cleanup).

Whenever the return code is not zero, the callers scope is changed back to his own original partition and the hold of the partition he wanted to leave is reset.
Special cancel condition: If MODFLD cannot issue an implicit hold for a caller, since the caller holds another partition already *AND* this hold had not been issued by MODFLD *AND* it is not the partition the caller wanted to change scope to, the caller is canceled.
Authorized: See PASCOPE service.
See notes: CANCUSE, SYSPIK, CANCPIK

PERBIT PER bit for partition.
Input: NEWVAL and PART parameter.
Modify PER active indication in the partition control block (PCB) and the save area PSW of the specified partition to on or off as specified by NEWVAL-operand.
Intended users: SDAID
See notes: NOAUTHC, SYSPIK, CANCPIK

PMODE Set saved permanent mode set function byte for tape devices.
Input Either PART and LU or PU (pub index) to identify the device and NEWVAL parameter.
Values returned if R15 is:
0 Successful, mode changed
8 device is not a tape
12 logical unit unassigned or assigned ignore.
See notes: NOAUTHR, NOPIK0, CANCPIK, CANCPUB
Intended users: JOB CONTROL

RESETOWN Reset ownership indication for partition and device.
Input: PART and PU parameter. The indication, that the device specified by the PUB index in PU is owned by the partition specified in PART is reset.
Return codes in R15:
0 Ownership indication reset.
16 PUB index in PU was invalid or PIK in PART was invalid.
Authorized: VSE/POWER and JOB CONTROL
See notes: CANCUSE, SYSPIK

RID Change current routine ID
This service is provided for VTAM, when VTAM wants to continue with a new RID. The requested RID must be passed in register 1. If current RID is 4 the new RID must be 16, if current RID is 8 or 16 the new RID must be 4. The task is canceled if it supplies any other input in register 1.
Authorized: VTAM.
See notes: CANCUSE

RIPLAUTH Give authorization for REIPL.
This service will authorize the requester to issue a system REIPL.
Authorized: KEY 0 callers.
See notes: CANCUSE

RUNMODE Change virtual or real execution mode.
Input: NEWVAL parameter. This service is provided for the INVPART routine to switch the current partition from virtual to real (NEWVAL=0) and vice versa (NEWVAL¬=0), before and after an EXEC REAL job step. The counter of active virtual partitions (IJBAPNO) and the run mode flag in the PIB (TRAM bit) are updated. If the partition was deactivated, it is reactivated before switching to real. The addressability scope of the partition is switched to the R space or back to the virtual allocation space.
This service is supported only for the current partition.
Authorized: SYSTEM
See notes: CANCUSE

RXEOJPTR Set the REXX information pointer in the task control block of the current task. Input: NEWVAL parameter containing new pointer value.
Intended users: REXX interpreter
See notes: NOAUTHC

SASCOPE Dropped with VSE/ESA 2.4.0. Caller is cancelled with illegal SVC. (see SASCOPE1, PASCOPE, PASCOPE1)

SASCOPE1 Set addressability scope and ensure its presence.
Input: NEWVAL parameter.
The service can be used to access static or dynamic spaces alike. It forbids deallocation for a partition in the space if it would otherwise be possible that the space is deallocated while the caller is still working in it. This 'hold' is revoked, as soon as the caller changes his address-

ability scope to another space or partition or his own original scope again.
It is important, that any sequence of changes, done by SASCOPE1 or PASCOPE1 requests (or both), finally ends in a return of the caller to his own original addressability scope or in shared space!
Return codes in R15 are:

0 Successful scope change.
8 Not successful, the specified space is not allocated
12 Not successful, the specified spaceid is invalid (for example, dynamic class not known, impossible second character etc.)

Like with PASCOPE1, any non zero return code indicates that the users scope has now changed back to his own original scope, and all 'hold' requests for other partitions, that had previously been issued by MODFLD are now reset.
Authorized: SYSTEM.
See notes: CANCUSE

SAVAR Set save area pointer of task.
Input: Set the pointer contained in the NEWVAL parameter as new save area pointer of current task.
Authorized: IPL.
See notes: CANCUSE

SYSAL Giv task system access list. NEWVAL value:

0 Give task system access list and return ALET for specified partition.
1 Give task original access list.

Authorized: Terminator

SYSRESW DASD file protection bypass.
Input: NEWVAL (and TASK) parameter. A value of 1 if DASD file protection is to be bypassed for the current task, else 0.
Authorized: KEY 0 callers.
See notes: NORC, CANCUSE, NOTID0, CURRTID

VSAMOPEN Open VSAM ACB flag.
Input: A NEWVAL value of 1 indicates, that there is at least one open VSAM ACB for the current partition, 0 indicates, there is none.
Intended users: OPEN, CLOSE only.
See notes: NORC

VTAMDISP VTAM AP exit scheduled.
Input: A NEWVAL value of 1 if the VTAM AP exit is scheduled for the current task, else 0.
Authorized: SYSTEM and VTAM only.
See notes: CANCUSE

VTAMOPEN Indicate open VTAM ACB.
Input: A NEWVAL value of 1 if there is at least one open VTAM ACB for the task specified by TASK, else 0.
Authorized: SYSTEM and VTAM only.
See notes: CANCUSE, NOTID0, CANCTID

Notes mentioned in field description above:

NORC No return code is provided by this service. A number of return codes may be introduced for this service in future times, all intended users will be notified.

NOFUTRC This service will not have future return codes added for compatibility reasons.

NOAUTHR No special authorization is required for this service. It is open to all users.

NOAUTHC Although no authority checking is done in this service it should only be used by the mentioned intended users, since only these users will be notified about interface changes.

CANCPIK The caller of this service is canceled, if the provided pik is invalid. A PIK is invalid if it is not a multiple of 16 or not smaller than the maximum PIK mentioned in SYSCOM.

CURRPIK Here the user is canceled if the PIK he provided (if he did provide one) was not equal to the PIK of the current partition (that means, the partition being serviced).

NOPIK0 Providing a PART value of zero or omitting the PART parameter for this service will be handled as if the PIK of the current partition had been supplied by the caller.

SYSPIK Providing a zero with the PART parameter or omitting this parameter will be considered as identification of the SYSTEM partition.

CANCPUB The user is canceled if the provided PUB index is invalid, that means, if it is negative or bigger than the number of added devices.

CANCTID The user is canceled if the provided TASK ID was invalid, that means,

- if note NOTID0 does not apply and the TASK ID is zero.
- if TASK ID is bigger maximum possible TASK ID.
- if task information block for task does not exist.
- if task is an inactive subtask.

CURRTID The user is canceled if the provided TASK ID is not equal to the current TASK ID. In combination with NOTID0 a zero is also accepted.

NOTID0 Providing a TASK ID value of zero or omitting the TASK parameter for this service will be handled as if the TASK ID of the current task had been provided.

SYSTID Providing a TASK ID of zero or omitting the TASK parameter will lead to a cancel of the callers task.

NOCANC There is no cancel condition for this service.

Register Usage: In general register R0, R1 and R15 should be considered changed after the macro invocation. *Never* rely upon a specific value still being preserved in a returned register (R0,R1 and R15), although it will be true with many services.

Internally the use of the registers is as follows:

R0 Is usually used to pass the PIK, TIK, PU or LU value. A value of 0 is passed if the PART or TASK operand is omitted.

R1 Is usually used to pass the new value for the specified field. The value must be right adjusted.

R15 Is used to pass the function code and return code.

MODHCF

The macro MODHCF provides the ability to change the direction in which the HCF is to be retrieved.

The macro has the following format:

```
[name]   MODHCF   {(hcfreg)|(1)},{BW|FW|UNC|ORMSG|REDISP}
```

hcfreg Is the general register containing the address of the HCFCB returned by the corresponding POINTHCF macro.

BW Starting from the current position, changes the direction of the next READ to backward.

FW Starting from the current position, changes the direction of the next READ to forward.

UNC Starting from the current position, it reverses the READ direction unconditionally.

Note: If the MODHCF results in an actual change of direction, then the next subsequent READHCF returns the record already provided by a preceding READHCF.

Output: Register 15 contains one of the following return codes.

0 (X'00')	Normal processing successfully completed.
4 (X'04')	Inconsistent input.
8 (X'08')	No record found, incorrect length.
12 (X'0C')	Unrecoverable I/O error.
16 (X'10')	HCF device is not ready.

Register Usage: The contents of general register 14 through 2 are destroyed by this macro.

MODVCE

This macro indicates to the supervisor the changing of a volume serial number of a DASD device. The supervisor reads the new volume serial number and updates the appropriate entry in the Volume Characteristics Table (VCT).

The macro has the following format:

```
[name]   MODVCE   {LOGUNIT={name1|(r1)}|CHNUNIT={name2|r2}}
                  [,CLASS={TAPE|DASD|ANY}]
                  [,RESERVE={YES,NO}]
                  [,SHARE={YES,NO}]
```

The operands have the following meaning:

LOGUNIT
: Address of device description in logical unit format.

  The volume characteristics entry that the user wants to be updated is identified by its logical assignment. This operand points to a halfword with the same format as a logical unit number in a CCB.

CHNUNIT
: Address of device description in physical unit (channel, unit) format (as in the PUB).

  The volume characteristic entry that the user wants to be updated is identified by its physical device address. This operand points to a halfword with the physical device address.

CLASS
: Indicates the device class for which the request has to be done. Default is DASD (that means, disk or diskette), ANY includes TAPE and DASD.

RESERVE
: YES: Do not allow the specified device to be assigned until a volume is mounted. RESERVE is not allowed for CLASS=TAPE|ANY.

SHARE
: YES: The device is defined shareable among different CPUs. SHARE is not allowed for CLASS=TAPE|ANY.

Output: Register 15 contains one of the following return codes.

0 (X'00')	Request successfully processed.
4 (X'04')	The logical unit specified is not assigned.
8 (X'08')	The physical unit specified is not in the system or the device does not belong to the class specified by the CLASS parameter.
12 (X'0C')	The device is not ready.
16 (X'10')	The VOL1 label has not been found or is not valid.
20 (X'14')	Some other unrecoverable I/O error occurred.
24 (X'18')	The device is not operational.

MSAT

The MSAT macro is used to manipulate stored assignment information.

It has the following format:

```
[name] MSAT  ID={ALT|ALP|CKU|DEL|INQ|NXT|PER|RSA|RSU|RTL1|RTP|
                                        DRL|DVR|DVU|NPM|NTM|PSP|PST}
             [,LOGUNIT={name1|(S,name1)|(r1)}]
             [,CHNUNIT={name2|(S,name2)|(r2)}]
             [,PHYUNIT={name3|(S,name3)|(r3)}]
             [,AREA={name4|(S,name4)|(r4)}]
             [,LEN={name5|(S,name5)|(r5)}]
             [,PID={name6|(S,name6)|(r6)}]
             [,MFG={name7|(r7)}]
             [,MODE={YES|name8|(S,name8)}]
```

The operands have the following meaning:

ID Specifies the function required

PER The current LUB value is saved as permanent and changed to UA, provided it is the first one to be saved. If there is already a permanent assignment saved, a return will be provided and the saved assignment will not be overwritten.

INQ An indicator byte is returned as leftmost byte of register 15 indicating the types of assignments that have been stored for the specified logical unit. The bits of the indicator byte have the following meaning:

```
X'80' Permanent alternate assignment stored
X'40' Temporary alternate assignment stored
X'20' Permanent assignment stored
X'10' Reserved
X'08' Reserved
X'04' Reserved
X'02' Reserved
X'01' Reserved
```

If no assignment is stored, return code 0 is given.

DEL All assignments, if any, of the specified logical unit in the specified partition are deleted, device ownership information is updated and the current assignment is set to UA.

ALP The physical device specified in CHNUNIT is noted as permanent, alternate assignment for the logical device specified by LOGUNIT and the device ownership information is updated, but only if the current assignment is also a permanent one. Only one chain of alternates is maintained for SYSPCH and SYSLST, if both are combined to SYSOUT.

ALT The physical device specified in CHNUNIT is noted as temporary, alternate assignment for the logical device specified by LOGUNIT and the device ownership information is updated, but only if the current assignment is also temporary.

NXT The current assignment (value in LUB) is saved as the last alternate one (in the pertinent chain that means, either temporary or permanent). The first alternate assignment is made the current one provided the associ-

ated device is not down. The mode byte is moved from the original current one to the new current one. If the device associated with the new assignment is down, the process is repeated until all alternates have been tried. If all alternate devices are down, return code 4 is returned. If the logical unit specified is SYSPCH or SYSLST and both are combined to SYSOUT then both LUBs are updated with the new alternate assignment found for the logical unit just being processed.

RSU The LUB of the logical unit specified is reset to the saved permanent assignment or unassign. Any temporary, alternate assignments are deleted. For each deleted assignment, the device ownership information is updated.

RSA Starting with the logical unit specified, the LUBs of the higher system logical units or programmer logical units of the specified partition are reset to the saved permanent assign or unassign. Any temporary alternate assignments are deleted. For each deleted assignment, the device ownership information is updated.

CKU Starting with the logical unit specified, all higher system logical units or programmer logical units of the specified partition are checked if they are assigned to the physical device specified in CHNUNIT. If at least one logical unit is assigned to the device, return code 0 is given. Return code 4 indicates that no logical unit in the range is assigned to the device.

NPM (JOB CONTROL, VSE/POWER)
All assignments of the specified logical unit are deleted and the specified PHYUNIT is noted as the current permanent assignment. Device ownership is updated for any deleted and new assignment.

NTM (JOB CONTROL, VSE/POWER, VTAM only)
All temporary assignments of the specified logical unit are deleted, all permanent assignments are saved and the specified PHYUNIT is noted as the current temporary assignment. Device ownership information is updated for any deleted and new assignment.

PST (VSE/POWER only)
Indicates that the specified device will be used in the specific partition as a dummy unit record device for spooling by VSE/POWER. All assignments are left unchanged, but device ownership for the specified partition is reset.

PSP (VSE/POWER only)
Indicates that the specified device is no longer being used as a VSE/POWER dummy device in the specified partition. All assignments to this device are reset to UA.

DVU (LIBRARIAN, VTAM only)
Indicates that the specified device is to be accessed by physical addressing in the specified partition. Device ownership information is updated and a 2-byte physical unit number is returned in the specified area.

DVR (LIBRARIAN, VTAM only)
Indicates that a physical addressing access to the device with the specified physical unit number in the specified partition is released. Device ownership information is updated and the field specified by the PHYUNIT parameter is changed to X'FFFF'. Each such request has to be paired with a previous ID=DVU request.

The following functions ID=RTL1|RTP|DRL require the additional parameters AREA and LEN because they return retrieved information in a user- defined area. These functions are used by LISTIO and DVCDN command processors.

RTL1 For the specified logical unit, the current assignment and all stored assignments are returned together with an indication of their type. Output for each assignment is of the form:

```
Byte 0   Flag byte

         X'80' Permanent alternate assignment
         X'40' Temporary alternate assignment
         X'20' Permanent assignment
         X'10' Temporary assignment
         X'08' - X'01' Reserved

Byte 1   High byte of PUB index for IODEV>254 or X'00'
Byte 2   PUB index
```

If all retrieved assignments fit into the user specified area, return code 0 is given. Return code 32 indicates that the area is too small. In both cases the number of existing assignments for the logical unit is returned in the first two-bytes of the user area.

RTP The LUB index of all higher system logical units or programmer units of the specified partition being assigned to the physical device specified in CHNUNIT is returned together with an indication of the type of assignment. Output for each logical unit in the range is of the form flag byte/00/LUBindex. The bits in the flag byte have the same meaning as for RTL1. Several bits may be on if different types of assignment of the logical unit exist for the specified physical device.

If all retrieved assignments fit into the user specified area, return code 0 is given. Return code 32 indicates that the area is too small. In both cases the number of existing assignments for the specified physical device is returned in the first two-bytes of the user area.

DRL (JOB CONTROL only)
Any permanent or temporary assignment of the specified logical unit in the specified partition to the specified device is changed to UA and its alternate assignments are deleted. Any alternate assignment to the specified device is deleted. For all changed or deleted assignments, device ownership information is updated. All changed or deleted assignments are returned together with an indication of their type. Output for each assignment is of the format flagbyte/00/PUBindex. The bits in the flagbyte have the following format:

```
X'80' Permanent alternate assignment
X'40' Temporary alternate assignment
X'20' Permanent assignment
X'10' Temporary assignment
X'08' Reserved
X'04' Reserved
X'02' Reserved
X'01' Reserved
```

The number of affected assignments is returned in the first two bytes of the user area. If all returned assignments fit into the specified area, return

code 0 is given. Return code 32 indicates that the area is too small. In this case, all assignments remain unchanged.

LOGUNIT Address of device description in logical unit format. This operand points to a halfword with the same format as a logical unit number in a CCB.

CHNUNIT Address of device description in physical unit format (channel, unit). This operand points to a halfword containing the physical device address.

PHYUNIT Address of a halfword containing a physical unit number (PUB index) to be used as new current assignment (NPM, NTM) or to be released (DVR). X'FFFF' is interpreted as UA, X'FEFF' as IGN.

AREA Address of the area where the retrieved information is to be returned. Only valid for ID=RTL1|RTP.

LEN Length of the area where retrieved information is to be returned. The length specification must be at least two bytes. Only valid for ID=RTL1|RTP.

PID Points to a two-byte field containing the PIK of the partition the logical unit belongs to. The default is the PIK of the issuing partition.

MFG Address of dynamic storage area that is to be used for construction of parameter list for reentrant programs.

MODE MODE=name|(S,name)
Address of a byte containing the mode to be set for the device provided by the PHYUNIT parameter. This form of the MODE specification is only valid for ID=NPM|NTM, if it is specified for a non-tape device it will be ignored at execution time. No checking of the mode byte is done.
MODE=YES
Indicates that mode handling should be done. This form of the MODE specification is only valid for ID=DEL and triggers the resetting of the temporary mode to the standard mode for all tape devices for which the assignment is reset. The specification will be ignored at execution time for non-tape devices.

Output: Register 15 contains one of the following return codes.

0 (X'00')	Requested function complete.
4 (X'04')	No permanent assignment stored (RSU). No assignment to device found (CKU). No alternates present or all devices down or all devices assigned to same physical unit (NXT).
8 (X'08')	No more space available (ID=ALT|ALP|PER).
12 (X'0C')	Status of alternate assignment incompatible with status of current assignment (ID=ALT|ALP). Permanent assignment already saved (ID=PER). Device is already spooled (PST) or not spooled (PSP). Device not in use by specified partition (DVR).
16 (X'10')	Logical unit specified exceeds the range of logical units
20 (X'14')	Physical unit specified not supported in the system. Physical unit specified is not a unit record device (PST,PSP).
24 (X'18')	Partition specified not supported by the system.
28 (X'1C')	Function requested not supported.
32 (X'20')	User area too small.
36 (X'24')	Specified device is already owned by or reserved for another partition (ALP|ALT|NPM|NTM|DVU).

40 (X'28')	Specified device is down (ALP|ALT|NPM|NTM|DVU).

The following list shows which operands are required with the different IDs:

ID	LOGUNIT	CHNUNIT	PHYUNIT	AREA	LEN	PID	MFG	MODE
PER	R	N/A	N/A	N/A	N/A	O	O	N/A
INQ	R	N/A	N/A	N/A	N/A	O	O	N/A
DEL	R	N/A	N/A	N/A	N/A	O	O	O
ALP	R	R	N/A	N/A	N/A	O	O	N/A
ALT	R	R	N/A	N/A	N/A	O	O	N/A
NXT	R	N/A	N/A	N/A	N/A	O	O	N/A
RSU	R	N/A	N/A	N/A	N/A	O	O	N/A
RSA	R	N/A	N/A	N/A	N/A	O	O	N/A
CKU	R	R	N/A	N/A	N/A	O	O	N/A
RTL1	R	N/A	N/A	R	R	O	O	N/A
RTP	R	R	N/A	R	R	O	O	N/A
NPM	R	N/A	R	N/A	N/A	O	O	O
NTM	R	N/A	R	N/A	N/A	O	O	O
DRL	R	R	N/A	R	R	O	O	N/A
PST	N/A	R	N/A	R	R	O	O	N/A
PSP	N/A	R	N/A	R	R	O	O	N/A
DVU	N/A	R	N/A	R	N/A	O	O	N/A
DVR	N/A	N/A	R	N/A	N/A	O	O	N/A
R = Parameter is required O = Parameter is optional N/A = Parameter not applicable								

NPGR

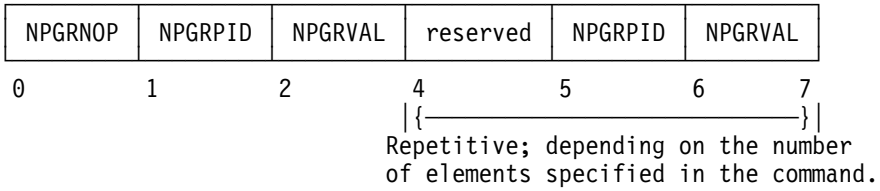
The NPGR macro causes the number of programmer LUBs to be set to the specified value. The macro is used by the job control NPGR command processing routine.
The service is available for static partitions only.
The format of the NPGR macro is as follows:

```
[name]  NPGR   [NPGRLST={name1|(r1)}]
```

NPGRLST Is the address of the parameter list into which the specified operands have to be placed before issuing the macro. The address of the parameter list may be supplied either as an operand or in a register. If the operand is omitted, register 1 is used.

The format of the parameter list is as follows:

NPGRLST



NPGRNOP It contains the number of operands which were specified in the NPGR command and which is equal to the number of elements in the parameter list.

NPGRPID It specifies the partition number from 0 to n for which the following number of programmer LUBs is to be allocated, where 0 is the background, and 1 to n are the foreground partitions F1 to Fn.

NPGRVAL It contains the number of programmer LUBs.

Output: Register 15 contains one of the following return codes:

0 (X'00')	The specified partition programmer LUB values are accepted.
8 (X'08')	The NPGR command is rejected. The sum of all partition programmer LUBs is larger than the supervisor generated NPGR Value.
12 (X'0C')	The NPGR command is rejected. At least one of the specified NPGR value is either below the minimum of 20 or above the maximum of 255.
16 (X'10')	The NPGR command is rejected. At least one of the specified partition has been started before (may be unbatched now).
20 (X'14')	The NPGR command is rejected. NPGR for BG was specified but another partition was already started before (may be unbatched now).
24 (X'18')	The NPGR command is rejected. Reallocation of BG LUBs is below highest assigned BG LUB.
28 (X'1C')	The NPGR command is rejected. A partition was specified, which is not supported.

NPGRLST

The macro NPGR generates a NPGR parameter list. Its format is as follows.

```
[name]  NPGRLST   [DSECT=YES]
```

DSECT YES: Forces the layout of the parameter list to be generated.

NUCLKUP — Nucleus Map Lookup

The NUCLKUP (Nucleus Map Lookup) macro can be used to retrieve the address and AMODE of a VSE/ESA system CSECT or ENTRY.

This macro runs in the key and state of the caller. On entry to this macro, register 13 must point to a 72-byte register save area. Users must include the CVT mapping macro.

Notes:

1. Only the BYNAME function is supported and will be described.
2. NUCLKUP is an internal macro.

The requirements for the caller are:

Authorization:	Problem or supervisor state, any PSW key
AMODE:	24- or 31-bit
ASC mode:	Primary
Control parameters:	Must be in the caller's primary address space
RID of requestor:	X'04', X'08' or X'40' - X'FF' for Callable Cell Pool services

```
[name]  NUCLKUP    BYNAME
                 ,NAME=name
                 ,ADDR=addr
```

BYNAME The user supplies the name of a CSECT or ENTRY and receives the address and AMODE of that CSECT or ENTRY.

NAME= Specifies the name of the CSECT.

name contains the 8 byte character name to be searched for or the address of that name (8 byte literal enclosed in apostrophes, or the address of the 8 byte literal which can be either an RX-type address, or register (1) - (12)).

Consider the following conventions:

- Callable cell pool services are referenced via names starting with C'CSR'. The following entry names are supported:
  - CSRPACT
  - CSRPBLD
  - CSRPCON
  - CSRPDAC
  - CSRPDIS
  - CSRPEXP
  - CSRPFRE
  - CSRPGET
  - CSRPGFR
  - CSRPQCL
  - CSRPQEX
  - CSRPQPL
  - CSRPRGT

ADDR= Contains the address of the CSECT or ENTRY that is returned.

The NUCLKUP service routine sets bit 0 of the word containing the address returned on a BYNAME request to indicate the AMODE (bit 0 on = CSECT / ENTRY has AMODE 31, AMODE 24 otherwise).

When control is returned, the registers contain the following information:

0 The address and AMODE of the CSECT or ENTRY.

1 In VSE/ESA register 1 is always 0.

2-14 Unchanged

15 Return code

The return codes in register 15 are as follows:

0 The request is satisfied.

4 The request is not satisfied.
For a BYNAME request, the name was not found and the location containing the address is set to zero.

8 The request was not satisfied because the type of request was not specified correctly. The location containing the address is set to zero.

12 The request is not satisfied because the requestor's RID was invalid.
The location containing the address is set to zero.

Example:

Example to place the address and AMODE of the entry point CSRPBLD in register 0.

```
NUCLKUP BYNAME,NAME='CSRPBLD',ADDR=(0)
```

PAGESTAT

The PAGESTAT macro returns the status of an area. Each page of the specified area is checked for validity and whether the page is used or not.

The macro has the following format:

PAGESTAT macro

```
[name] PAGESTAT  {name1|(0)},{ name2|(1)}[,ALET={name3|(r1)}]
[,MFG={name4|(r2)}]
```

name1 Address within the first page to be handled.

name2 Address within the last page to be handled.

name3 Address of a 4-byte area containing the ALET of a data space. This specification is required only if name1 and name2 designate pages within a data space.

name4 Address of a 12-byte dynamic storage area that is to be used for parameter list construction for re-entrant programs.

Output: Register 15 contains an identification about the status of the first page of the specified area, as described below.

Register 0 contains the address of the first page of the area that has a different status. If register 0 contains zeros all pages of the area have the same status.

Return information in register 15:

Byte 3=0 Address is valid and page is used.
Byte 3=4 Address is valid and page is not used.
Byte 3=8 Address is invalid which means that the appropriate task is canceled whenever it attempts to reference this page or address. This may be due to one of the following reasons:
- Address beyond virtual storage.
- PTEBIT and PTEINVAD are on in corresponding page table entry.

Byte 3=12 Address connected to failing storage.
Byte 3=16 Address belongs to VPOOL.

Execution of this macro with name1 higher than name2, with an invalid ALET, or with an ALET that does not identify a data space results in 'cancel due to invalid address' (ERR25).

Execution of this macro with ALET specified and AMODE 24 results in 'cancel due to mode violation' (ERR45).

The macro SPLEVEL SET= can be used to generate downward compatible code of the PAGESTAT macro, but output on ESA 1.3.0 and upward is always as described above.

PFIXCHPT

The macro ensures that during checkpointing the parameter list for PFIXREST is built (see also “SVC 74 (X'4A' - PFIXCHPT/PFIXREST)” on page 268). The function is restricted to programs with a PSW key of zero.

The macro has the following format:

```
[name]   PFIXCHPT   {name1|(1)},{length|(0)}
```

name1 The symbolic name of an area where the entries have to be inserted. For the layout of the entries refer to the description of the “SVC 74 (X'4A' - PFIXCHPT/PFIXREST)” on page 268.

length Length of the area provided by the user.

Note: A length of 0 is not allowed.

Output: Register 2 contains zero if no additional area is needed; register 2 contains 4 if an additional area is needed. A non-zero byte is placed right after the last generated entry in each area.

PFIXREST

The macro ensures that during RESTART the pages which were permanently fixed at checkpoint time are PFIXed again with the same value of the PFIX counter (see also “SVC 74 (X'4A' - PFIXCHPT/PFIXREST)” on page 268). The function is restricted to programs with a PSW key of zero.

The macro has the following format:

```
[name]  PFIXREST {name1|(1)}
```

name1 Is the symbolic name for a list of consecutive 10-byte entries built during checkpointing (end of list indication is X'FF' following the last entry). For the layout refer to the description of the “SVC 74 (X'4A' - PFIXCHPT/PFIXREST)” on page 268.

POINTHCF

The macro POINTHCF opens the HCF and provides the interface for accessing the HCF. According to the parameters passed, the logical record pointer is initialized and the specific logical file limits are determined.

The macro has the following format:

```
[name]   POINTHCF {WRITE[,{CONTINUE|CREATE}]|
                   READ,{REDISPL|PRINTLOG[,{ALL|NEW}]
                        [,{NOSELECT|SELECT}]}}
                  ,SAVE={name|13}
                  [,MFG={name1|(r)}]
```

WRITE — Opens the HCF for the output.

CONTINUE — This option ensures writing onto the HCF behind the last previously written record.

CREATE — Initializes the HCF (as a result of the SET RF=CREATE command).

READ — Opens the HCF for the input.

REDISPL — Opens the HCF for the redisplay function of CRT.

PRINTLOG — Opens the HCF for the PRINTLOG utility program.

- ALL — All available data is to be retrieved.
- NEW — Only the new data (since last IPL) is to be retrieved.
- NOSELECT — No special records are to be selected.
- SELECT — Special record selection is to take place.

SAVE — Specifies a 68-byte register save area in which the caller's registers are saved.

MFG — Name of a work area where the parameter list has to be built (for re-entrant programming). The length of this work area must be 20 bytes. Register notation may be used.

Output: Register 1 points to the control block (HCFCB), and this address must be provided in subsequent READHCF, MODHCF, SKIPHCF or CLOSEHCF requests. If POINTHCF returns in error, register 1 will be set to zero.
For POINTHCF with WRITE, the HCFCB address is not returned in register 1.

Register 15 contains one of the following return codes.

0 (X'00')	Normal processing successfully completed.
4 (X'04')	Inconsistent input / inconsistent state.
8 (X'08')	No record found, incorrect length.
12 (X'0C')	Unrecoverable I/O error.
16 (X'10')	HCF device is not ready.
20 (X'14')	READ HCF not yet opened. WRITE Incorrect HCF format.
24 (X'18')	Unauthorized POINTHCF request.
28 (X'1C')	HCF not accessible. Byte 0 of register 15 contains the original return code of the GETVCE/GETVIS/EXTENT request. READ No JOB statement for partition, or EXTENT failed. Byte 0 of register 15 is zero if no JOB statement was detected for the partition. WRITE Open unsuccessful, or GETVCE failed. Byte 0 of register 15 is zero if OPEN was unsuccessful.
32 (X'20')	HCF too small.
36 (X'24')	GETVIS failed. Byte 0 of register 15 contains the original return code of the GETVCE/GETVIS/EXTENT request.
40 (X'28')	HCF does not meet extent specifications.

Register Usage: The contents of general register 14 through 2 are destroyed by this macro.

PRODEXIT

The objective of this service is to allow vendor programs to interface with the system at predefined points in an orderly manner rather than every OEM program hooking into the system. The use of such an interface by OEM products should result also in less problems for the customer.

General

Exit Specification: A set of vendor interfaces will be provided via exit points which must adhere to the following specifications.

1. To use the vendor exits the application must be authorized. The valid token will be obtained via the PRODID macro, the user must be authorized (DTSECTAB) to access IJBVEND and the products exit routine.
2. An exit can be dynamically enabled/disabled without affecting other exits for the same event type.
3. The different exit specifications will be stacked per event and invoked as needed for the specific exit.

Customer Value: Vendors are enabled to attach their applications to VSE via officially supported interfaces. Therefore, less problems should occur that are caused by "miscommunication" between the vendor application and VSE/ESA. The result for the customer is a more reliable VSE system (effect: less effort for test of vendor application, less effort for problem determination).

Vendors can use officially supported and documented interfaces for writing their applications which should result in increased development productivity.

IBM Internal Value: Flexibility for future development will be increased by the fact that vendor applications no longer rely on "unofficial" interfaces. Giving the vendors official interfaces allows IBM development organizations to change everything "below" these interfaces without being concerned about a potential negative impact on a vendor application.

Description

The vendor interfaces provide services to

- maintain vendor exits and
- activate the specified vendor exits in the corresponding components (vendor exit hooks).

The PRODEXIT macro offers these services.

Note: Vendor exit hooks described in this chapter are product sensitive interfaces.

Services To Maintain Vendor Exits: Programs with a valid token returned by the existing PRODID service are allowed to use the PRODEXIT services.

The program may use these services to

- define an exit (PRODEXIT DEFINE)
- enable an exit (PRODEXIT ENABLE)
- disable an exit (PRODEXIT DISABLE)
- delete an exit (PRODEXIT DELETE)
- return from an exit (PRODEXIT RETURN)
- describe the contents of the input/output areas of the PRODEXIT services (PRODEXIT DSECT). This area is called **PRODEXIT area**.

The exits can be defined for given exit points in VSE/ESA components. A VSE/ESA component is called a **class**, the exit point **subclass**.

Classes are SUP, PSUP, SVC, FSVC, BAM, AIT, DOC, DOCP, LNG.

Subclasses can be defined for every class, e.g. a subclasses for class SUP is the POSTFCH exit, for class BAM the PRE-OPEN exit. If no subclass is specified, all invocations of the specified class will receive control.

The DEFINE and DELETE services expand to normal SVCs, the ENABLE and DISABLE services to fast path SVCs.

Exits may be specified for update (e.g. DTFs or JCL statements) or monitoring (no update). Monitoring exits receive control after exits defined for update. Multiple exits may be specified for a class/subclass, however, only one exit per valid PRODID token (returned by PRODID DEFINE). Exception: Monitoring and UPDATE exits for the same class/subclasses can coexist for the same TOKEN. The exits are activated in FIFO or priority order (dependent on PRODEXIT DEFINE specification).

Every exit has a **scope**. When the class/subclass is processed for:

system scope	the exit will be activated for all tasks.
partition scope	the exit will be activated for all tasks within the partition. **Note:** including system tasks which execute services for this partition
task scope	the exit will be activated for the current task only.

An eight byte user area may be given as input to the define service. This field can hold e.g. an anchor for a work area and will be given to the exit during activation.

Note: addressability to the work area has to exist for every possible task under which the exit can be invoked

Service For Vendor Exit Activation (PRODEXIT ACTIVATE): The activation service can be used by VSE/ESA components at given points and transfers control to defined vendor exits. Dependent on the environment PRODEXIT ACTIVATE expands

- in supervisor state to
  - BASSM interface to the activate processing
    (if no exit is enabled control is returned immediately)
- in problem program state to
  - a normal SVC.

Notes:

1. The RID (Routine Identifier) identifies the environment in which the task is running, e.g. the usage of save areas.
2. Vendor exits have to consider that SVCs in ICCF pseudo partitions are intercepted by ICCF, that may change the input to a specific service.

Supervisor State (SSTATE) Exits:
The exits receive control with the RID as described in the corresponding class/subclass paragraph (RID ¬= USERTID) and disabled for interrupts. Register 1 points to a parameter list holding exit related information (described by

PRODEXIT DSECT). Register 15 holds the exit routine address at exit entry and may be used by the exit routine. The exit has to return with PRODEXIT RETURN. The exit will receive control in primary mode and AMODE 31.The high order bit of the exit address (defined by PRODEXIT DEFINE) has to be set. AMODE 24 will only be accepted in VSE/ESA 1.3 MODE=370. General purpose and access registers are saved before activation of the exit and restored after PRODEXIT RETURN. All areas as well as the exit routine itself have to be fixed for RID = SYSTEMID and may be pageable for environments, that allow page faults (RID=REENTRID).

Problem Program State (PSTATE) Exits (RID = USERTID):
The PRODEXIT ACTIVATE service provides a save area and saves the status of the service issuer (PSW, general purpose register and access register). The vendor exit receives control in primary mode, problem program state (RID = USERTID) and AMODE 31. High order bit of the exit address (DEFINE service) has to be set to 1 if running in ESA environment. Register 1 points to a parameter list holding exit related information (described by PRODEXIT DSECT). Register 15 holds the entry point address. The exit has to return via the PRODEXIT RETURN service, which may schedule another exit or restores the saved registers and returns to the calling component.

Note: PSTATE exits are interruptible, however external interrupts will be delayed which drive IT, TT and OC exits.

Register Conventions:

On Exit Entry

register 1	point to the area as defined by PRODEXIT DSECT
register 14	exit return address
register 15	exit entry point address

There is no need to save and restore registers for the user of the PRODEXIT ACTIVATE service. However, in case of SSTATE ACTIVATE out of the SVA, the ACTIVATE issuer has to provide a save area via R13

On PRODEXIT Service Return: On return from any PRODEXIT service, the standard register convention adheres (original contents of R0, R1 and R15 may be destroyed). R14 must have the same value as on entry to the exit.

PRODEXIT Area Location At Exit Entry:
The location and storage key of the PRODEXIT area, which is input for the exit entry, is described in the corresponding class/subclass paragraph.

Vendor Exit Process:

The PRODEXIT ACTIVATE service returns

- after **all** vendor exits of same class/subclass are processed.

  Note: Exceptions are described in the according chapter of the Vendor Exit Definition.

If more than one vendor exit is established for a subclass, the priority defined (during PRODEXIT DEFINE) in the PRODEXIT area is used to determine the activation order. Exits defined with the same priority will be activated in the sequence the PRODEXIT DEFINE was issued (FIFO).

Vendor exits defined for **update** (via PRODEXIT DEFINE) are activated prior to all other exits (**monitoring** exits). Within the 'update' and 'monitoring' exits the priority is used to determine the activation sequence. 'Update' exits are processed before 'monitoring' exits. 'Monitoring' exits are, per definition, **not** allowed to change any system information, a check however is not possible.

If more than one exit is to be activated, the next exit will get the output (if any) of the former exits. Exceptions are described in the class/subclass.

Note: If there are more exits defined for **update**, the one with the highest priority will be invoked first. The one with the lowest priority will be invoked last and therefore this one has the final chance to change the output before the control will be given back to the 'ACTIVATE' component.

The PRODEXIT DEFINE allows to set an 8 byte field (IJBVUSW) for each exit.

Vendor exit routine must run in the SVA (24 / 31 bit) running in ESA mode, the AMODE must be 31. Further, reentrancy have to be considered.

Whenever a new task will be attached, it will be automatically ENABLEd for:

- an exit of scope 'system'
- an exit of type partition and the new task belongs to the partition for which an exit has been enabled

Note: An attach will fail, if the necessary control blocks for vendor exit activation cannot be GETVISed if there are vendor exits defined in the system which are 'flagged' to be critical ones (see PRODEXIT DEFINE). In case of Dynamic Partitions, the allocation will be done anyway, but an according MSG will be issued to the console. ==> Critical exit handling will not be retrofitted to VSE/ESA 1.3

Notes:

1. per task one PSTATE exit, one SSTATE exit with RID=REENTRID one SSTATE exit with RID=SYSTEMID (excluding FSVC CLASS) and one SSTATE exit with RID=SYSTEMID CLASS = FSVC can be active the same time.
2. a program check in the vendor exit or an abnormal termination will cause skipping of normal PC-exit or AB-exit routine
3. Any STXIT macro invocation out of an vendor exit will lead to ERR21 ==> 'illegal SVC'

Vendor Exit PSW Key:
The vendor exit will be activated with the same PSW key as the PRODEXIT ACTIVATE issuing exit subclass.

Deletion Of Vendor Exits:

It is recommended to delete an exit (PRODEXIT DELETE), when the exit is no longer needed. If no deletion occurs, an enabled exit will have a different 'enable' time dependent on its scope:

system scope	enabled until next IPL.
partition scope	enabled until de-allocation of partition. *)
task scope	enabled until completion of end-of-task process. *)

Note: *) if multiple partitions/tasks are enabled only the ending partition/task will be DISABLEd, any exit remains being DEFINEd until next IPL if not explicitly deleted via the PRODEXIT DELETE service

Recovery:

Whenever possible the VSE/ESA will recover from exit problems.

Notes:

1. In case of abnormal termination of an exit and a possible recovery,
   - the task will continue with abnormal termination. The dump routine will issue a cancel message and the corresponding failing address will indicate that a vendor exit caused the problem. Environments, where this is not possible, will be described. Other exits of the same subclass will be skipped.
2. Abnormal termination of vendor exits running with RID=SYSTEMID will lead to a hardwait.

Service To Check For Exits (PRODEXIT CHECK): PRODEXIT CHECK can be used by VSE/ESA components to check if an exit is enabled for the class/subclass in the current VSE task. This service expands to an SVC and is only possible for PSTATE exit types

PRODEXIT DEFINE Service

PRODEXIT DEFINE defines a vendor exit of a given class and subclass. If the subclass is omitted, the exit will be defined for all subclasses. The DEFINE service returns a PRODEXIT token, that allows to use PRODEXIT ENABLE, DISABLE and DELETE services. If the class/subclass is enabled, the exit is activated depending on the following scope:

system	for every VSE task,
partition	for the current partition (where ENABLE is issued),
task	for the current task (where ENABLE is issued).

Before an exit can be activated the exit has to be enabled via PRODEXIT ENABLE, because the exit is initially disabled.

If the PRODEXIT DEFINE service was given by a previous job step that ended abnormally, the PRODEXIT DEFINE service gives the previously returned token again.

Vendor exits **must** not issue PRODEXIT DEFINE.

Note: Multiple exits may be specified for a class/subclass, however, the same exit (subclass) can only be specified once per valid PRODID token (returned by PRODID DEFINE). Exception: Monitoring and UPDATE exits for the same class/subclasses can coexist for the same TOKEN.

Requirements For The Caller:

- Authorization: valid PRODID token (as returned by PRODID DEFINE)
- AMODE: ANY
- RMODE: ANY
- ASC-mode: Primary
- Invocation: SVC

```
[label] PRODEXIT DEFINE,AREA={(1)|name}
```

Input Parameter Description:

AREA= points to the PRODEXIT area where the input information is available. It may be specified in register notation or as the name of the input PRODEXIT area. The PRODEXIT area must be in an address range valid for this partition. The following fields of the PRODEXIT DSECT are used as input parameters:

IJBVCLS class of exit. The contents are described by PRODEXIT DSECT. Only one exit class can be specified. Class is required.

Note: Any CLASS and/or SUBCLASS specific restrictions for an exit definition will be decribed in the according exit chapter.

IJBVSCL subclass of exit. The contents are described by PRODEXIT DSECT. Some classes of exits do have multiple subclasses. If in that case IJBVSCL indication is omitted the exit will be defined for all subclasses. Multiple subclasses can be specified

also. If an exit has only one subclass, it is recommended to set the corresponding subclass indication because of future extensions. Subclass is optional. Subclass is optional for those Classes which allow the subclass specifications.

Note: Classes 'SVC' and 'FSVC' currently do not support single subclasses, therefore any subclass specification is invalid for them and will cause Return Code 12.

IJBVTOK holds a PRODID token previously obtained by PRODID DEFINE. Token is required.

IJBVEXT holds the exit routine address, the high order bit defines the AMODE on entry of the exit (=0 -> AMODE 24, =1 -> AMODE 31). The exit address is required. AMODE 31 is required if running in ESA mode. If high order bit is not on in VSE/ESA 2.1, DEFINE will issue RC 12. The exit routine has to be within the SVA!

Note: If the class contains multiple subclasses and only the class is specified, the exit routine will be invoked for each subclass.

IJBVSCP identifies the scope of the exit. Scope is required.

IJBVUSW The 8 byte user word is available for the vendor product, e.g. IJBVUSW may hold the pointer to a work area. The contents of IJBVUSW will be transferred to the exit. This field is optional and should be cleared to X'00' if no input is available.

IJBVPRIO This one byte field defines the priority of the exit. The priority can be in the range from 0 to 99 (decimal), where 99 is the highest priority and 0 the lowest. The exits are activated in the 'priority' order (high to low), exits with the same priority are activated in FIFO order dependent on the PRODEXIT DEFINE sequence. Exceptions for exit invocations are described in the according exit definition chapters.

IJBVFUPD (in flag IJBVFLAG). Exits defined for update (IJBVFUPD set) are processed prior to all other (monitoring) exits. Within the two groups (update or monitoring) the exits are activated in priority (IJBVPRIO) order.

Note: Proper usage of 'UPDATE' / 'MONITOR' is the vendor's responsibility, VSE does not perform any checks.

IJBVCRIT (in flag IJBVFLAG). An exit, defined as critical (IJBVCRIT set) indicates to the system that after the successful processing of that DEFINE service, the system will not ATTACH tasks or ALLOC partitions for which the internal control blocks for the exit acti-

vation cannot be obtained. This indication remains valid until all exit are DELETEd which do have that flag on.

Notes:

1. In case of dynamic partitions, allocation will be done anyway, but a MSG will be written to the console. In addition, the flag triggers what happens when a second PSTATE exit for the same task is invoked: if the critical bit is set for the second PSTATE exit, the program is cancelled with error 47 (reason code 3 'second vendor exit invocation invalid'), in the other case the invocation is ignored.
2. The IJBVCRIT bit will be ignored prior to VSE/ESA 2.1.

Output: is an 8 byte PRODEXIT token passed back in the input PRODEXIT area at label IJBVETK.

Return Codes

passed back in register 15
0 request accepted.
8
- => PRODID TOKEN index part out of range (>256)
- => Vendor Product Table not defined
- => Vendor Product Table Entry deleted
- => Vendor Product TOKEN doesn't match

12 control block format error
- => no CLASS specified
- => multiple CLASSes specified
- => specified CLASS not supported
- => specified SUBCLASS not supported
- => PRODEXIT AREA length invalid (<84)
- => SCOPE definition invalid
- => Priority definition invalid (>99)
- => no EXIT routine address specified
- => EXIT not in SVA
- => EXIT routine not AMODE=31 (required in VSE/ESA Version 2)
- => SUBCLASS specification invalid for current CLASS
- => SCOPE not SYSTEM for CLASS SUP, SUBCLASS=EXT
- => SCOPE not SYSTEM for CLASS AIT

16 System phase $IJBVEND not loaded.
20 GETVIS error, return code from GETVIS in register 0.
24 request rejected, an exit is already defined for at least one subclass or for the class for the specified TOKEN. The according TOKEN has been returned in the input PRODEXIT area at label IJBVETK.
28 Initial system setup for PRODEXIT services failed due to GETVIS problems

Cancel Conditions:

X'21' illegal SVC - unknown function code, invalid invocation of PRODEXIT DEFINE, ASC-mode not PRIMARY

X'25' parameter list address is invalid. Will result in an 'addressing exception'.

PRODEXIT ENABLE Service

This macro enables the exit defined by PRODEXIT DEFINE for exit activation,

system wide	if the IJBVSYS flag was set
for current partition	if the IJBVPAR flag was set
for current task	if the IJBVTSK flag was set

during exit definition (PRODEXIT DEFINE).

Note: Multiple ENABLE requests for the same exit are possible.

Requirements For The Caller:

- Authorization: valid PRODEXIT token
- AMODE: ANY
- RMODE: ANY
- ASC-mode: Primary
- Invocation: fast path SVC

```
[label] PRODEXIT ENABLE
```

Input Parameter Description:

register 1 and 2 must contain the PRODEXIT token returned by the PRODEXIT DEFINE service.

Output: None.

Return Codes

passed back in register 15

0 request accepted.
4 exit already enabled.
8 task disabled for PRODEXIT services (GETVIS problem)
16 System phase $IJBVEND not loaded, or no vendor defined in system

Cancel Conditions:

X'21' illegal SVC - invalid PRODEXIT token.

PRODEXIT DISABLE Service

This macro disables the exit activation,

system wide	If the IJBVSYS flag was set
for current partition	if the IJBVPAR flag was set
for current task	if the IJBVTSK flag was set

during exit definition (PRODEXIT DEFINE).

Note: Multiple DISABLE requests for the same exit are possible.

Requirements For The Caller:

- Authorization: valid PRODEXIT token
- AMODE: ANY
- RMODE: ANY
- ASC-mode: Primary
- Invocation: fast path SVC

```
[label] PRODEXIT DISABLE
```

Input Parameter Description:

register 1 and 2 must contain the PRODEXIT token returned by the PRODEXIT DEFINE service.

Output: None.

Return Codes

passed back in register 15

0 request accepted.
4 exit already disabled.
8 task disabled for PRODEXIT services (GETVIS problem)
16 System phase $IJBVEND not loaded, or no vendor defined in system

Cancel Conditions:

X'21' illegal SVC - invalid PRODEXIT token.

PRODEXIT DELETE Service

The exit will be disabled (if not yet done) and the exit definition will be deleted. PRODEXIT DELETE will wait for PRODEXIT RETURN, if the exit is active.

Vendor exits **must** not issue PRODEXIT DELETE.

Note: When a vendor product is deleted (via PRODID DELETE) from the system, all associated vendor exits (defined by PRODEXIT DEFINE) are deleted, too. If one of these exits is active, PRODID DELETE will wait for the PRODEXIT RETURN from these exits.
Vendor exits which have been defined via one DEFINE, can only be deleted together.

Requirements For The Caller:

- Authorization: valid PRODEXIT token
- AMODE: ANY
- RMODE: ANY
- ASC-mode: Primary
- Invocation: SVC

```
[label] PRODEXIT DELETE
```

Input Parameter Description:

register 1 and 2 must contain the PRODEXIT token returned by the PRODEXIT DEFINE service.

Output: None.

Return Codes

passed back in register 15
0 request accepted.
16 System phase $IJBVEND not loaded.

Cancel Conditions:

X'21' illegal SVC - invalid PRODEXIT token, may be exit already deleted, or invalid invocation of PRODEXIT DELETE.

PRODEXIT ACTIVATE Service (PSTATE)

The VSE component has to use the PRODEXIT ACTIVATE service to call the vendor exits for the according subclass. It receives control again after **all** vendor exits of the subclass are processed, that is after the PRODEXIT RETURN service issued by the last vendor exit.

Requirements For The Caller:

- Authorization: none
- AMODE: ANY
- RMODE: ANY
- ASC-mode: Primary
- Invocation: SVC
- PSW key at exit entry: PSW key of PRODEXIT ACTIVATE issuer

Vendor Exit Return Codes: Return Codes as defined by subclass (see corresponding 'class' paragraph),will be returned by the vendor exit in field IJBVRC of the PRODEXIT area.

PRODEXIT ACTIVATE Return Codes

The following return codes are passed back in R15 to the class/subclass, that issued the PRODEXIT ACTIVATE service:

0 PRODEXIT ACTIVATE successfully processed.
4 non-zero vendor exit return code passed in field IJBVRC.
12 control block format error
=> no CLASS specified
=> multiple CLASSes specified
=> specified CLASS not supported
=> specified SUBCLASS not supported
=> PRODEXIT AREA length invalid (<84)
=> no SUBCLASS specified
=> multiple SUBCLASSes specified
=> no PRODEXIT Extension addr. specified in IJBVINF
16 System phase $IJBVEND not loaded.

Cancel Conditions:

X'21' illegal SVC - invalid invocation, e.g. invalid function code or ASC-mode not PRIMARY

X'25' invalid parameter list address

X'47' Reason Code 3 - 'second vendor exit invocation invalid'

Macro Description For VSE Components Running In PSTATE:

```
[label] PRODEXIT ACTIVATE,AREA={(1)|name}
```

Input Parameter Description:

AREA= points to an PRODEXIT area where the input information is available. It may be specified in register notation or as the name of the input area. The PRODEXIT area must be in a valid address range. The following fields of the PRODEXIT DSECT are used as input parameters:

IJBVCLS	class of exit. The contents are described by PRODEXIT DSECT. Only one exit class has to be specified.
IJBVSCL	subclass of exit. The contents are described by PRODEXIT DSECT. Only one exit subclass has to be specified.
IJBVINF	This field holds an address pointing to an area prepared by the class/subclass. The information is given to the vendor exit on entry.
IJBVCEXT	If used, this field holds the address pointing to the PRODEXIT CLASS Extension area, otherwise it must be set to zero.

Note: The parameter list will be validated.

PRODEXIT ACTIVATE Service (SSTATE)

The VSE component has to use the PRODEXIT ACTIVATE service to call the vendor exits for the according subclass. It receives control again after **all** vendor exits of the subclass are processed, that is after the PRODEXIT RETURN service issued by the last vendor exit.

Requirements For The Caller:

- Authorization: supervisor state
- AMODE: ANY
- RMODE: ANY
- ASC-mode: Primary
- Invocation: BASSM
- PSW key at exit entry: PSW key of PRODEXIT ACTIVATE issuer
- The PRODEXIT ACTIVATE expansion code must be fixed
  (new save area VSEVISA is used, which may be overwritten by another PRODEXIT ACTIVATE if a page fault occurs)

Vendor Exit Return Codes: Return Codes as defined by subclass (see corresponding 'class' paragraph),will be returned by the vendor exit in field IJBVRC of the PRODEXIT area.

PRODEXIT ACTIVATE Return Codes

If the issuer of the PRODEXIT ACTIVATE service has specified RC=YES (default RC=NO), the following return codes are passed back in R15 to the issuer of the PRODEXIT ACTIVATE service:

0 PRODEXIT ACTIVATE successfully processed.

4 non-zero vendor exit return code passed in field IJBVRC.

Note: If RC=YES is specified for a class/subclass, for which the vendor exit does not pass a return code, a return code of zero is always returned to the PRODEXIT ACTIVATE issuer.

Macro Description For VSE Components Running In SSTATE: For VSE/ESA components, which invoke SSTATE exits, all subclass specific input for the according exits as well as the allocation of the related areas will be provided by the PRODEXIT ACTIVATE Service. The VSE/ESA component which wants to activate an SSTATE exit just has to code the following macro at the defined exit point.

```
[label] PRODEXIT ACTIVATE,ID=sclfct[,AREA=(1)][,RC=NO|YES]
```

Note: If the VSE/ESA component is residing in the SVA, make sure that:

- R13 points to a register save area!
- the following DSECTs with label are included:

```
SYSCOM,   label SYSCOM
SUPAVT,   label SUPAVT
SUPAVTEX, label SUPAVTEX

SGVEND    DSECT=YES
```

Input Parameter Description:

ID= identifies the SUBCLASS function which has to be provided.

The currently defined 'sclfct's are:

POSTSIO	Post SIO/SSCH Exit subclass of (class SUP)
IOINT	I/O interrupt Exit (class SUP)
PCKEX	Program Check Exit (class SUP)
EXT	External Interrupts Exit (class SUP)
EXPHASE	Program Retrieval - Exchange Phase Exit (class SUP)
PREFCH	Program Retrieval - Pre Fetch Exit (class SUP)
POSTFCH	Program Retrieval - Post Fetch Exit (class SUP)
MSG	Message Processing Exit (class DOC)
CMDREP	Command/Reply Processing Exit (class DOC)
SVC	Supervisor Call CLASS Exits (class SVC)
FSVC	Fast Path Supervisor Call CLASS Exits (class FSVC)

AREA= points to the PRODEXIT Extension area containing the subclass related information which gets passed to the exit. This optional parameter must be specified for those subclasses which do need more than 160 bytes for their PRODEXIT Extension area and/or want to provide the subclass specific information by themselves. If area is not specified, the PRODEXIT ACTIVATE service takes care of all the allocations and provides the information as specified for the class/subclass.

Note: For RID = SYSTEMID subclasses, no Page Faults are allowed!

RC= RC setting allows SUBCLASS to trigger Register 15 contents on RETURN from the exit.

RC=NO or RC not specified, causes Register 15 not being modified by the exit routine.

RC=YES, causes register 15 being set to 0 / 4 depending on the vendor return code setting in IJBVRC.

Output: As defined by subclass

PRODEXIT RETURN Service

The vendor exit has to return with PRODEXIT RETURN. RETURN will use the same PRODEXIT area given to the exit as input, therefore the area must not be changed (except IJBVRC). However, IJBVRC may not be changed if it is a vendor exit without output. Any information returned from the vendor exit will be input for the next exit to be called. E.g. the vendor exit return code IJBVRC may be analyzed/changed by a following exit. After the return of the last vendor exit (of the subclass) IJBVRC will be given to the caller.

Requirements For The Caller:

- Authorization: none
- AMODE: ANY
- RMODE: ANY
- ASC-mode: Primary
- Invocation: SSTATE = BSM, PSTATE = SVC

  Note: make sure that R14 will not be changed during exit processing, it contains the link/return address from the preceding 'ACTIVATE' exit invocation.

```
[label] PRODEXIT RETURN
```

Input Parameter: none

Output: None.

Cancel Conditions (PSTATE Only):

X'21' ==> illegal SVC - exit is not active, invalid function code,

X'2E' ==> deadlock situation during re-occupation of the LTA

Note: For SSTATE callers that condition will lead to unpredictable results.

PRODEXIT CHECK Service

The CHECK service checks, if a vendor exit has to be activated for the given class/subclass and current task. This service can only be used in problem state programs (PSTATE). The CHECK service may be used to skip the preparation for the exit input, when no exit is available or enabled for the given class/subclass.

Requirements For The Caller:

- Authorization: none
- AMODE: ANY
- RMODE: ANY
- ASC-mode: Primary
- Invocation: SVC
- PSTATE only

```
[label] PRODEXIT CHECK,AREA={(1)|name}
```

Input Parameter Description:

AREA= points to an area where the input information is available. It may be specified in register notation or as the name of the input area. The area must be in an address range valid for this partition. The following fields of the PRODEXIT DSECT are used as input parameters:

IJBVCLS class of exit. The contents are described by PRODEXIT DSECT. Only one exit class can be specified.

IJBVSCL subclass of exit. The contents are described by PRODEXIT DSECT. Only one subclass can be specified.

Note: The PRODEXIT area will only be validated from begin of area to IJBVSCP. Specification of class and subclass is mandatory.

Output: The contents of register 1 will be unpredictable

Return Codes: passed back in register 15

0 exit activation necessary.
8
=> exit not defined or enabled
=> PRODEXIT service not available for current task (GETVIS problems)
12 control block format error
=> no CLASS specified
=> multiple CLASSes specified
=> specified CLASS not supported
=> specified SUBCLASS not supported
=> PRODEXIT AREA length invalid (<38)
=> no SUBCLASS specified
=> multiple SUBCLASSes specified
16 System phase $IJBVEND not loaded.

Cancel Conditions:

X'21' Illegal SVC - unknown function code, invalid invocation of PRODEXIT CHECK or ASC-mode not PRIMARY

X'25' parameter list address is invalid.

X'47' Reason Code 3 - 'second vendor exit invocation invalid'

PRODEXIT DSECT Service

This macro generates one DSECT, that describes the in/output for the following services:

- the input and output expected by PRODEXIT DEFINE.
- the input and output expected by PRODEXIT ACTIVATE.
- the input and output expected by PRODEXIT CHECK.
- the output expected by PRODEXIT RETURN.

PRODEXIT DSECT describes the common part, that is input for the PRODEXIT services and the vendor exits itself as well as the output area for information returned to the PRODEXIT issuer. The PRODEXIT DSECT holds an address (at label IJBVINF) which points to the PRODEXIT extension of the corresponding sub-class at exit entry. The unique subclass extension part labels start with **IJBVx**, where **x** is defined as

B Class = BAM
C Class = SVC
F Class = FSVC (fast SVC)
L Class = LNG (languages)
P Class = PSUP (supervisor PSTATE)
S Class = SUP

The input/output for the classes AIT, DOCP, and DOC is not described by the PRODEXIT DSECT but by the following macros:

Class = AIT described by MAPARCMD (see “Vendor Exit - VSE/AF Console Support (Class = DOC)” on page 155)
Class = DOC described by IJBCSMX
Class = DOCP described by IJBCNMEX

The first fields of the extension are common to all exits.

Area Location:

- RMODE: ANY, in private or shared area dependent on PRODEXIT service.
- ASC-mode: Primary

```
[label] PRODEXIT DSECT [,CLASS={BAM|SVC|FSVC|LNG|PSUP|SUP|ALL}]
```

CLASS=nnn generates one PRODEXIT Extension DSECT for the specified CLASS.

CLASS=ALL generates the labels for all CLASSES (but AIT, DOCP and DOC)

PRODEXIT DSECT Layout

```
        ***************************************************************
        *
        * PRODEXIT DSECT
        *
        ***************************************************************
        PRODEXIT PRODEXIT DSECT
        *        SUPERVISOR - PRODEXIT - 5686-066-06
000000  PRODEXIT DSECT
000000  IJBVLN   DS    H                 LENGTH OF PRODEXIT AREA
000002  IJBVCLS  DS    XL4               FLAG-BYTES EXIT CLASS
        IJBVSUP  EQU   X'80'              EXIT CLASS   SUP
        IJBVPSUP EQU   X'40'              EXIT CLASS   PSUP
        IJBVSVC  EQU   X'20'              EXIT CLASS   SVC
        IJBVFSV  EQU   X'10'              EXIT CLASS   FAST PATH SVC
        IJBVBAM  EQU   X'08'              EXIT CLASS   BAM
        IJBVAIT  EQU   X'04'              EXIT CLASS   ATTENTION
        IJBVDOC  EQU   X'02'              EXIT CLASS   CONSOLE
        IJBVLNG  EQU   X'01'              EXIT CLASS   LANGUAGES
000006  IJBVSCL  DS    XL32              FLAG BYTES - EXIT SUBCLASS
        IJBVPEOT EQU   X'80'              END OF TASK IF CLASS=PSUP
        IJBVBPRO EQU   X'80'              PRE-OPEN IF CLASS=BAM
        IJBVBPRC EQU   X'40'              PRE-CLOSE IF CLASS=BAM
        IJBVBEOX EQU   X'20'              END OF EXTENT IF CLASS=BAM
        IJBVBCVH EQU   X'10'              COMMON VTOC HANDLER IF CLASS=BAM
        IJBVBPOO EQU   X'08'              POST OPEN IF CLASS=BAM
        IJBVBPOC EQU   X'04'              POST CLOSE IF CLASS=BAM
        IJBVSPIO EQU   X'80'              POST SIO/SSCH IF CLASS=SUP
        IJBVSIOI EQU   X'40'              I/O INTERRUPT IF CLASS=SUP
        IJBVSPCK EQU   X'20'              PROG. CHECK IF CLASS=SUP
        IJBVSEXT EQU   X'10'              EXT. INTERRUPT IF CLASS=SUP
        IJBVSEXP EQU   X'08'              EXCHANGE PHASE IF CLASS=SUP
        IJBVSPRF EQU   X'04'              PREFETCH IF CLASS=SUP
        IJBVSPOF EQU   X'02'              POSTFETCH IF CLASS=SUP
        IJBVDMSG EQU   X'80'              MESSAGE PROC. IF CLASS=DOC
        IJBVDCR  EQU   X'40'              COMMAND REPLY IF CLASS=DOC
        IJBVAPRE EQU   X'80'              PRE-SCAN IF CLASS=AIT
        IJBVAPOE EQU   X'40'              POST-SCAN IF CLASS=AIT
        IJBVLOPN EQU   X'80'              LNG-OPEN IF CLASS=LNG
        IJBVCLEN EQU   *                  END OF PRODEXIT CHECK INPUT
000026  IJBVSCP  DS    X                 FLAG BYTE - SCOPE OF EXIT
        IJBVSYS  EQU   X'80'              SYSTEM WIDE
        IJBVPAR  EQU   X'40'              PARTITION
        IJBVTSK  EQU   X'20'              TASK
000027  IJBVFLAG DS    X                 FLAG BYTE
        IJBVFUPD EQU   X'80'              EXIT WILL UPDATE INPUT
        IJBVCRIT EQU   X'40'              CRITICAL EXIT
        IJBVSUPD EQU   X'08'              UPDATE EXIT HAS BEEN SKIPPED
000028  IJBVRC   DS    F                 VENDOR EXIT RETURN CODE
00002C  IJBVINF  DS    F                 POINTER TO INPUT INFORMATION
000030  IJBVEXT  DS    F                 EXIT ROUTINE ADDRESS
000034  IJBVPRIO DS    X                 EXIT PRIORITY
000035           DS    3X                RESERVED
000038  IJBVUSW  DS    XL8               USER WORD, E.G. ANCHOR FOR WORK AREA
000040  IJBVTOK  DS    XL8               PRODID TOKEN
000048  IJBVETK  DS    XL8               PRODEXIT TOKEN
000050  IJBVCEXT DS    XL4               CLASS EXT. POINTER
        IJBVLEN  EQU   *                  END OF PRODEXIT AREA
```

Figure 21. DSECT Generated by 'PRODEXIT PRODEXIT DSECT'

PRODEXIT EXTENSION DSECT Layout (for 'multiple CLASSes')

```
[label] PRODEXIT DSECT,CLASS=ALL
```

```
      ********************************************************************
      *
      * PRODEXIT EXTENSIONS DSECT
      *
      * THIS DSECT GETS GENERATED VIA AN ADDITIONAL INTERNAL
      * PARAMETER 'CLASS=ALL' AND PROVIDES ALL EXTENSIONS FOR ALL
      * SPECIFYABLE EXIT CLASSES/SUBCLASSES
      *
      ********************************************************************
      PRODEEXT PRODEXIT DSECT,CLASS=ALL
      *        SUPERVISOR - PRODEXIT - 5686-066-06
000000 PRODEEXT DSECT
      * * * * * * * * * * *  COMMON EXTENSION  * * * * * * * * * * * * * *
000000 IJBVLENV DS    H                LENGTH OF VAR. PART
000002 IJBVPITI DS    0XL4             OVERLAY FOR NEXT TWO HALFWORDS
000002 IJBVPIK  DS    H                PIK OF CURRENT TASK
000004 IJBVTIK  DS    H                TID OF CURRENT TASK
      * DO NOT REARRANGE PREVIOUS TWO HALFWORDS
000006          DS    H                RESERVED
      IJBVELEN EQU   *                LENGTH OF FIXED PART
      IJBVELN1 EQU   *-IJBVLENV        LENGTH OF FIXED PART
      * * * * * * * * * * *   LNG EXTENSION    * * * * * * * * * * * * * *
000008          ORG   IJBVELEN          LNG EXTENSION
000008 IJBVLLVL DS    F                LEVEL NUMBER OF THE PARAMETER LIST
00000C IJBVLANG DS    CL8              LANG.INVOKING EXIT: COBOL,PL/I,LE
000014 IJBVLNAM DS    CL8              NAME FROM COBOL ASSIGN STMT, OR
      *                                PL/I FILE DECLARE STATEMENT
00001C IJBVLLUN DS    H                LOGICAL UNIT NUM. FROM CBL ASSIGN STM
      *                                OR PL/I FILE DCL STMT, ZERO IF OMMIT.
00001E IJBVLOPM DS    X                PROPOSED OPEN MODE:
      IJBVLIN  EQU   X'01'             - INPUT
      IJBVLIO  EQU   X'02'             - I-O (EG. TYPEFLE=INPUT+UPDATE=YES
      IJBVLOUT EQU   X'03'             - OUTPUT
      IJBVLEXT EQU   X'04'             - EXTEND (APPEND)
      IJBVLWRK EQU   X'05'             - WORK (EG. COMPILER WORK FILE)
00001F IJBVLDEV DS    X                DEVICE TYPE FOR FILE:
      IJBVLDSD EQU   X'01'             - SAM DISK
      IJBVLDDA EQU   X'02'             - DAM DISK
      IJBVLDVS EQU   X'03'             - VSAM DISK
      IJBVLDUT EQU   X'04'             - UNLABELED TAPE
      IJBVLDLT EQU   X'05'             - LABELED TAPE
      IJBVLDCD EQU   X'06'             - CARD
      IJBVLDPR EQU   X'07'             - PRINTER
      IJBVLDUA EQU   X'08'             - UNASSIGNED (UA)
      IJBVLDIG EQU   X'09'             - ASSIGNED TO IGNORE (IGN)
000020 IJBVLLBA DS    A               A(DBLB/TLBL) GOT BY LABEL FUNCT=GETLBL
000024 IJBVLLBL DS    F                LENGTH OF DLBL OR TLBL
000028 IJBVLLBB DS    F                LEN(BUFFER) CONTAINING DLBL/TLBL
```

Figure 22 (Part 1 of 3). DSECT generated by PRODEEXT PRODEXIT DSECT,CLASS=ALL

```
00002C IJBVLOPT DS    XL2               OTHER OPEN OPTIONS.
       IJBVLORV EQU   X'80'              - OPEN REVERSED SPECIFIED
       IJBVLNRW EQU   X'40'              - OPEN NO REWIND SPECIFIED
       IJBVLASC EQU   X'20'              - ASCII TAPE FILE
       IJBVLOPF EQU   X'10'              - OPTIONAL FILE (COBOL ONLY)
       IJBVLCBM EQU   X'08'              - MODE=C (COLUMN BINARY MODE)
       *                                   (3505 AND 3525 CARD DEVICES ONLY)
       IJBVLOMR EQU   X'04'              - MODE=O (OPTICAL MARK READ)
       *                                   (3505 CARD DEVICES ONLY)
       IJBVLRCE EQU   X'02'              - MODE=R (READ COLUMN ELIMINATE)
       *                                   (3505 AND 3525 CARD DEVICES ONLY)
00002E IJBVLPUB DS    X                 PUB DEVTYPE UNLAB TAPE/CARD /PRT
00002F IJBVLFEX DS    X                 FILE EXISTS FLAG
       *                                - X'00' IF NOT SET
       *                                - X'01' IF FILE CURRENTLY EXISTS, OR
       *                                - X'02' IF FILE DOES NOT EXIST
000030 IJBVLLUO DS    H                 LOGICAL UNIT NUM. DETERMINED BY EXIT
000032 IJBVLFID DS    CL44              FILE ID FOR FILE, ANY THIRD PARTY
       *                                DISK/TAPE MGR CONTR.CHAR.S REMOVED
00005E IJBVLLEX DS    H                 EXTENT NUM. OF LAST VOLUME, SAM ONLY
000060 IJBVLRFI DS    H                 RECORD FORMAT FROM PROGRAM:
       IJBVLRFF EQU   X'80'              - FIXED
       IJBVLRFV EQU   X'40'              - VARIABLE
       IJBVLRFU EQU   X'20'              - UNDEFINED
       IJBVLRFB EQU   X'10'              - BLOCKED
       IJBVLRFS EQU   X'08'              - SPANNED
       IJBVLRFA EQU   X'04'              - ASA CONTROL CHARACTERS
       IJBVLRFM EQU   X'02'              - MACHINE CONTROL CHARACTERS
       *                                EG. X'90' FOR FIXED BLOCKED.
000064 IJBVLRSI DS    F                 REC. LENGTH FROM PROGRAM
000068 IJBVLBSI DS    F                 BLOCK SIZE FROM PROGRAM
00006C IJBVLRFO DS    H                 REC FORMAT OF EXIST. FILE,
       *                                FLAGS AS IJBVLRFI.
000070 IJBVLRSO DS    F                 REC LENGTH OF EXIST FILE IF AVAIL.
000074 IJBVLBSO DS    F                 BLOCKSIZE OF EXSIT.FILE IF AVAIL.
000078 IJBVLLMA DS    A                 A(LIOCS MODULE) IN DTF AFTER OPEN
00007C IJBVLLMS DS    A                 ADDRESS WHERE TO STORE A(LIOCS MOD)
       *                                PRIOR TO BE OVERWRITTEN WITH ABOVE
       * * * * * * * * * * *  BAM EXTENSION  * * * * * * * * * * * * * * * *
000080         ORG   IJBVELEN          BAM EXTENSION
000008 IJBVBEXI DS    CL16
000018         ORG   IJBVBEXI
000008 IJBVBFLG DS    X                 FLAG BYTE
       IJBVBJCT EQU   X'80'               REQUEST FOR JOB COMTROL
000009         DS    3X                RESERVED
00000C IJBVBDTF DS    F                 DTF POINTER
000010         ORG   IJBVBDTF
00000C IJBVBCPL DS    F                 ADDRESS OF CVH PARM.LIST
000010         ORG   ,
       IJBVBLEN EQU   *                 LENGHT OF BAM EXTENSION
       * * * * * * * * * * *  SVC EXTENSION   * * * * * * * * * * * * * * * *
000080         ORG   IJBVELEN          SVC EXTENSION
000008 IJBVCPSW DS    CL8               OLD SVC PSW
000010 IJBVCIC  DS    CL4               LOW CORE BYTE 136-139
000014 IJBVCRID DS    CL2               RID OF INTERRUPTED TASK
000016         DS    CL2               RESERVED
000018 IJBVCARG DS    CL64              ACCESS REGISTERS
000058 IJBVCGRG DS    CL64              GENERAL REGISTERS
000098 IJBVCVIN DS    CL8               INFO RETURNED BY VENDOR EXIT
       IJBVCLNG EQU   *-IJBVCPSW        LENGTH OF SVC EXTENSION
```

Figure 22 (Part 2 of 3). DSECT generated by PRODEEXT PRODEXIT DSECT,CLASS=ALL

```
      * * * * * * * * * * * *  FAST SVC EXTENSION  * * * * * * * * * * * * *
0000A0          ORG   IJBVELEN           EXTENSION FOR FAST SVC
000008 IJBVFPSW DS    CL8                OLD SVC PSW
000010 IJBVFIC  DS    CL4                LOW CORE BYTE 136-139
000014 IJBVFRID DS    CL2                RID OF INTERRUPTED TASK
000016          DS    CL2                RESERVED
000018 IJBVFARG DS    CL64               ACCESS REGISTERS
000058 IJBVFGRG DS    CL64               GENERAL REGISTERS
000098 IJBVFVIN DS    CL8                INFO RETURNED BY VENDOR EXIT
       IJBVFLNG EQU   *-IJBVFPSW         LENGTH OF FSVC EXTENSION
       * * * * * * * * * * * *  PSUP EXTENSION   * * * * * * * * * * * * * * * *
0000A0          ORG   IJBVELEN           EXTENSION FOR PSUP
000008 IJBVPCC1 DS    X                  FIRST CANCEL CODE
000009 IJBVPCC2 DS    X                  SECOND CANCEL CODE
00000A IJBVPFL1 DS    X                  FLAG BYTE
       IJBVPFMT EQU   X'80'               MAIN TASK TERMINATION
       IJBVPFST EQU   X'40'               SUBTASK TERMINATION
00000B          DS    X                  RESERVED
       IJBVPLEN EQU   *                  LENGTH OF PSUP EXTENSION
       * * * * * * * * * * * * *   SUP EXTENSION   * * * * * * * * * * * * * * *
       ************************************************************************
       *    FETCH EXITS
       ************************************************************************
00000C          ORG   IJBVELEN            EXT. SUP SUBCL. XPHASE/PRE/POSTFCH
000008 IJBVSFOP DS    CL8                ORIGINAL PHASENAME
000010 IJBVSFEP DS    CL8                EXCHANGED PHASENAME (BY EXIT)
000018 IJBVSFSC DS    X                  SVC CODE
       IJBVSFL1 EQU   *-IJBVSFOP         LENGTH OF VARIABLE PART FOR EXPHASE
000019 IJBVSFLI DS    CL7                NAME OF LIBRARY
000020 IJBVSFSL DS    CL8                NAME OF SUBLIBRARY
000028 IJBVSFDE DS    CL40               DIRECTORY ENTRY
000050 IJBVSFRC DS    X                  RETCODE (SET ONLY FOR POSTFCH EXIT)
       IJBVSFLN EQU   *-IJBVSFOP         LENGTH OF VARIABLE PART
       ************************************************************************
       *    POSTSIO EXIT
       ************************************************************************
000051          ORG   IJBVELEN            EXT. FOR SUP SUBCLASS POSTSIO
000008 IJBVSCCB DS    CL4                CCB ADDRESS
00000C IJBVSCU1 DS    CL2                CUU OF DEVICE I/O WAS ISSUED TOO
       IJBVSPOL EQU   *-IJBVSCCB         LENGTH OF VARIABLE PART
       ************************************************************************
       *    IOINT EXIT
       ************************************************************************
00000E          ORG   IJBVELEN            EXT. FOR SUP SUBCLASS IOINT
000008 IJBVSCSW DS    CL8                CSW OF DEVICE CAUSING INTERR
000010 IJBVSCU2 DS    CL2                CUU OF DEVICE CAUSING INTERRUPT
       IJBVSIOL EQU   *-IJBVSCSW         LENGTH OF VARIABLE IOINT PART
       ************************************************************************
       *    EXT EXIT
       ************************************************************************
000012          ORG   IJBVELEN            EXT. FOR SUP SUBCLASS EXT
000008 IJBVSESW DS    CL8                OLD EXTERNAL INTERRUPT PSW
000010 IJBVSEIC DS    CL4                EXT.INTERRUPT CODE X'84'-X'87'
000014 IJBVSETI DS    CL2                TID OF TASK WHOSE TIMER EXPIRED
000016 IJBVSERI DS    CL2                RID
       IJBVSELN EQU   *-IJBVSESW         LENGTH
       ************************************************************************
       *    PCKEX EXIT
       ************************************************************************
000018          ORG   IJBVELEN            EXT. FOR SUP SUBCLASS PCKEX
000008 IJBVSPSW DS    CL8                OLD PROGRAM CHECK PSW
000010 IJBVSPIC DS    CL4                PROGRAM INTER. CODE X'8C'-X'8F'
000014 IJBVSPFA DS    CL4                ADDRESS CAUSING PAGE FAULT X'90'
000018 IJBVSPAR DS    CL64               ACCESS REGISTERS
000058 IJBVSPGR DS    CL64               GENERAL PURPOSE REGISTERS
000098 IJBVSPRI DS    CL2                RID AT INTERRUPTION TIME
00009A IJBVSPAI DS    CL1                EXECEPTION ACCESS IDENT. X'A0'
       IJBVSPLN EQU   *-IJBVSPSW         LENGTH OF SUBCLASS PCKEX
       IJBVSLEN EQU   *                  MAX. LENGTH OF EXT. FOR CLASS SUP
```

Figure 22 (Part 3 of 3). DSECT generated by PRODEEXT PRODEXIT DSECT,CLASS=ALL

PRODEXIT BAM CLASS EXTENSION DSECT Layout

The internal CLASS parameter 'BINT' generates the DSECT of the PRODEXIT CLASS Extension as it is required currently only for the CLASS 'BAM'

```
[label] PRODEXIT DSECT,CLASS=BINT
```

```
       ***************************************************************
       *
       * PRODEXIT CLASS EXTENSION DSECT
       *
       * THIS DSECT GETS GENERATED VIA AN ADDITIONAL INTERNAL
       * PARAMETER 'CLASS=BINT' AND PROVIDES THE LAYAOUT FOR THE
       * PRODEXIT CLASS EXTENSION AREA, WHICH IS CURRENTLY ONLY
       * USED FOR THE BAM SUBCLASSES WHICH ARE INVOKED OUT OF THE LTA
       *
       ***************************************************************
       BAMCEXT  PRODEXIT DSECT,CLASS=BINT
       *        SUPERVISOR - PRODEXIT - 5686-066-06
000000 BAMCEXT  DSECT
000000 IJBVBLN  DS    CL2               LENGTH OF BAM EXTENSION
000002 IJBVBCRI DS    CL2               COMREG INFO TO BE SAVED
000004 IJBVBNXT DS    CL8               NAME OF NEXT PHASE TO BE FETCHED
00000C IJBVBLTA DS    CL16              LTA INFO TO BE SAVED
00001C IJBVBR15 DS    CL4               REG 15 SAVE AREA
000020         DS    CL4               RESERVED
000024 IJBVBLSA DS    CL120             LTA SAVE AREA
       IJBVBLNG EQU   *                 LENGHT OF EXTENSION
```

Figure 23. DSECT generated by BAMCEXT PRODEXIT DSECT,CLASS=BINT

PRODEXIT Vendor Exit Definitions

Vendor Exit - VSE/AF Supervisor (Class = SUP): Description

The exit class SUP offers SSTATE exits.

The following exits are supported:

- I/O interrupt
- post SIO/SSCH
- external interrupt
- program check interrupt
- program retrieval (pre and post fetch)

Post SIO/SSCH (Subclass = POSTSIO)

Purpose	The exit will be implemented for the following reasons • Monitoring
Location	Before current SGVSEPT probe points in SGSCHED,
Exit Type	SSTATE
RID	SYSTEMID (=0, no page faults allowed)
Input	IJBVINF points to the communication area. The following information is passed to the exit routine: • The CUU that the I/O was issued to • The CCB address
Output	None

I/O Interrupt (Subclass = IOINT)

Purpose	The exit will be implemented for the following reasons: • Monitoring
Location	Before current SGVSEPT probe point
Exit Type	SSTATE
RID	SYSTEMID (=0, no page faults allowed)
Input	IJBVINF points to the communication area. The following information is passed to the exit routine. • CSW • CUU of device causing the interrupt

Program Check (Subclass = PCK)

Purpose The exit will be implemented for the following reasons

- Monitoring
- Error recovery

Note: PCK exits will not be invoked during processing of SVC exits!

Location Before current SGVSEPT probe point in SGPCK

Exit Type SSTATE

RID SYSTEMID (=0, no page faults allowed)

Input IJBVINF points to the communication area. The area is contained in the SVA, RMODE is ANY. The following information is passed to the exit routine.

- program check old PSW
- program interruption code (loc X'8C'-X'8F')
- RID at interruption time
- TID and PIK of interrupted task
- page fault address (location X'90') if page fault
- exception access identification (location X'A0') if page fault due to access to data space
- general purpose registers and access registers

Output A return code is set up in IJBVRC

00 continue normal program check handling

04 Only for 'update' exits (IJBVUPD). Continue with PSW, general purpose/access registers as specified in communication area.

Note: The first 'update' exit which returns with RC=4 causes skipping of all following 'update' exit invocations. Only 'monitor' type exits will then get control prior to the execution of the RC=4 processing. An exit invocation skipping gets posted in the PRODEXIT area (bit IJBVSUPD in byte IJBVFLAG).

08 Only for 'update' exits (IJBVUPD). Ignore program check and return to interrupted task.

External Interrupts (Subclass = EXT)

Purpose The exit will be implemented for the following reasons

- Control APPC/IUCV/VMCF
- Monitoring

Note: The only predictable exit invocation for SUBCLASS=EXT is provided for SCOPE=SYSTEM. Therfore any other SCOPE will be rejected during PRODEXIT DEFINE processing (RC=12).

Location Before current SGVSEPT probe point in SGNUC (after ENTEXT)

Exit Type SSTATE

RID SYSTEMID (=0, no page faults allowed)

Input IJBVINF points to the communication area. The area is contained in the SVA, RMODE is ANY. The following information is passed to the exit routine.

- External old PSW
- External interruption code (loc X'84'-X'87')
- TID and PIK of interrupted task
- TID of task whose timer expired (if any)
- RID of interrupted task

Output A return code is set up in IJBVRC

00	continue external interrupt processing
08	Only for 'update' exits (IJBVUPD). Ignore external interrupt.

I/F Supervisor (PRODEXIT issuer) - IJBVEND

Output A return code in register 9:

00	continue external interrupt processing
1	ignore external interrupt

VSE/AF Program Retrieval (SSTATE)

Program Retrieval - Exchange Phase (Subclass = EXPHASE)

Purpose

The exit will be implemented for the following reasons

- security
- monitoring
- to gain a probe point

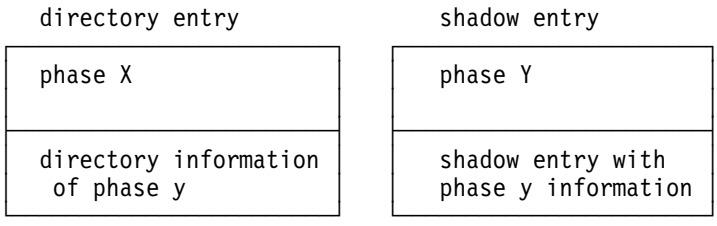
Location

The exit will be called for

- SVC 1 (FETCH), SVC 2, SVC 4 (LOAD), SVC 23, SVC 51, SVC 65 (CDLOAD)

Note, that the phasename in the user area is never exchanged.

- CDLOAD/CDDELETE Processing:
  The exit will be placed at the beginning of the CDLOAD/CDDELETE processing, before the CDLOAD anchor table is searched for the requested phase. This allows an exchange of the phasename. If the phasename is exchanged the follow-on processing is done with this exchanged phasename. The user provided phasename is no longer of interest. The entry in the anchor table will contain the exchanged phasename.
- SVC 2 Move Mode Processing
  The exit will be called before the SVA is searched for the requested phase.
  In case the phase is not found in the SVA, the common processing is called (with the original phase name).
  Note, that during this processing the exit is called again.
- Common Processing
  (called by SVC 1, SVC 4, SVC 23, SVC 51, SVC 65, and SVC 2 if phase is not in the SVA).
  The exit will be placed at the beginning of the common processing to allow an exchange of the phasename. From now on all processing is done with the changed phasename. The user provided phasename is no longer of interest. In case the user has provided a directory entry, both the directory entry and the shadow entry are filled with phase information of the exchanged phasename. The phasename in the user provided directory is not exchanged.



The directory entry contains a pointer to the shadow entry. If the DE is used for follow-on requests it is checked whether the directory entry and the shadow

entry describe the same phase. To avoid problems with the different phasenames the exit is called and must exchange the phasename. Otherwise the information in the directory entry is not considered and the original phase x is loaded.

Exit Type SSTATE

RID REENTRID (reentrant programming required), fast path SVCs of class A allowed, but not monitored.

PRODEXIT/input area RMODE ANY, shared area

Input The address of the input area is located in IJBVINF of PRODEXIT DSECT.

The input area holds the following information at exit entry:

- PIK of the current task (2 bytes)
- task-id (TID) of the current task (2 bytes)
- SVC code that generated the request (1 byte)
- name of program to be fetched (8 bytes)

Output The return code has to be returned in IJBVRC and the output (if any) is returned in the 'input area' (pointer to area in field IJBVINF):

- return code is set up in IJBVRC:
  - **0** continue processing
  - **4** Only for 'update' exits (IJBVUPD).
    Change of phase name requested.
    The phase name will be changed to the phase name supplied in the 'input area'.
  - **8** IGNORE
    Skip FETCH / LOAD / CDLOAD / CDDELETE processing.
- new phase name in IJBVSFEP (8 byte - if IJBVRC = 4)

Note: When a vendor exit requests to exchange the phasename, it is in the responsibility of the vendor product to guarantee the availability of the exchanged phase. If the phase cannot be loaded, results may be unpredictable.

Program Retrieval - Pre Fetch (Subclass = PREFCH)

Purpose The exit will be implemented for the following reasons

- security
- monitoring
- to gain a probe point

Location The exit will be called after directory task process (directory search) and will run as service owner (user task). It is only called, if the directory search ended with return code 0. It is note called, if the phase is in the SVA.

Exit Type SSTATE

RID REENTRID (reentrant programming required), fast path SVCs of class A allowed, but not monitored.

PRODEXIT/input area RMODE ANY, shared area

Input The address of the input area is located in IJBVINF of PRODEXIT DSECT. The input area holds the following information at exit entry:

- PIK of the current task (2 bytes)
- task-id (TID) of the current task (2 bytes)
- SVC code that generated the request (1 byte)
- name of program to be fetched (8 bytes)
- library name of program (7 bytes)
- sublibrary name of program (8 bytes)
- directory entry (DE=VSE format - 40 bytes)

Output return code is set up in IJBVRC:

0 continue processing
4 Only for 'update' exits (IJBVUPD).
Reject load request (security violation will be posted to the user).

Program Retrieval - Post Fetch (Subclass = POSTFCH)

Purpose The exit will be implemented for the following reasons

- security
- monitoring

Location The exit will be placed

- after the program fetch task processing (loading the phase) and will run as service owner (user task). It will be called even if an error occurred.
- after moving a move phase to the LTA during SVC 2 processing.

FETCH and LOAD (successful load/fetch). (For SVC 2 only if the phase was not found in the SVA).

Exit Type SSTATE

RID REENTRID (reentrant programming required), fast path SVCs of class A allowed, but not monitored.

PRODEXIT/input area RMODE ANY, shared area

Input The address of the input area is located in IJBVINF of PRODEXIT DSECT. The input area holds the following information at exit entry:

- PIK of the current task (2 bytes)
- task-id (TID) of the current task (2 bytes)
- SVC code that generated the request (1 byte)
- name of program to be fetched (8 bytes)
- library name of program (7 bytes)
- sublibrary name of program (8 bytes)
- return code
- directory entry (DE=VSE format - 40 bytes)

Output None.

Vendor Exit - VSE/AF Supervisor (Class = PSUP)

Description *:* The exit class PSUP offers PSTATE exits.

The following exits are supported:

- End-of-task ($IJBSEOT)

VSE/AF End-Of-Task (PSTATE)

End-Of-Task - $IJBSEOT Phase (Subclass = EOT)

Purpose The exit will be implemented to allow task clean-up.

Location The exit will be placed prior to FREEVIS ALL into the $IJBSEOT routine and will be called for main- and subtasks in the AMODE defined by the exit address.

Exit Type PSTATE

PRODEXIT/input area RMODE ANY, shared area

Input The address of the input area is located in IJBVINF of PRODEXIT DSECT. The input area holds the following information at exit entry:

- PIK of current task (2 byte)
- task-id (TID) of current task (2 byte)
- first cancel code (TIBCNCL) (1 byte)
- second cancel code (TIBCNCL2) (1 byte)
- flag byte to indicate main- or subtask termination (1 byte)

Output None.

Vendor Exit - VSE/AF Supervisor Call (Class = SVC)

Description *:* The exit class SVC offers SSTATE exits only. The vendor exit gets control whenever a SVC with exception of SVC 107 (fast path SVC) or SVC 124 (Vendor SVC) is issued.

Note: Subclass SVC code is subject for future extension. Any subclass specification during PRODEXIT DEFINE will be rejected with RC=12!

Purpose The exit will be implemented for the following reasons

- SVC screening
- Monitoring
- Adding vendor SVC numbers
- Ignoring SVCs
- Expansion on IBM SVC processing
- Security

Location Before current SGVSEPT probe point in SGNUC (after ENTSVC)

Exit Type SSTATE

RID SYSTEMID REENTRID (page faults allowed, but not monitored)

Input IJBVINF points to the communication area. The area is contained in the SVA, RMODE is ANY. The following information is passed to the exit routine.

- SVC old PSW
- SVC interruption code
- TID and PIK of interrupted task
- Register and access register at time of interruption

Output A return code is set up in IJBVRC:

00 continue normal SVC handling

04 Only for 'update' exits (IJBVUPD). Substitute SVC number and continue with general purpose/access registers as specified in communication area.

08 Only for 'update' exits (IJBVUPD). Ignore SVC and return to interrupted task if SVC old PSW address has not been modified.
Ignore SVC and go to routine where modified SVC old PSW address is pointing to (IJBVCPSW) using general purpose/access registers as specified in the communication area. (Changing the registers is possible for VSE/ESA releases starting with 2.1.1, not before.)

Note: The first 'update' exit which returns with RC=8 causes skipping of all following 'update' exit invocations.

		Only 'monitor' type exits will then get control prior to the execution of the RC=8 processing. An exit invocation skipping gets posted in the PRODEXIT area (bit IJBVSUPD in byte IJBVFLAG).
	12	Only for 'update' exits (IJBVUPD). Cancel user due to security violation. Only allowed when SVC issuer (user) had RID=USERTID. In case of security violation, the vendor exit can pass additional information in the prodexit extension area, field IJBVCVIN. This information will appear in the cancel message.

I/F Supervisor - SVC Handler (PRODEXIT issuer) - IJBVEND

Output	A return code in register 9:	
	00	continue normal SVC processing
	1	ignore SVC
	2	security violation. A program check is forced by ENTSVC processing.

Vendor Exit - VSE/AF Fast Path Supervisor Call (Class = FSVC)

Description *:* The exit class FSV offers SSTATE exits only. The vendor exit gets control whenever a SVC 107 (fast path SVC) is issued.

Note: Subclass FSV code is subject for future extension. Any subclass specification during PRODEXIT DEFINE will be rejected with RC=12!

Purpose The exit will be implemented for the following reasons

- SVC screening
- Monitoring
- Security

Location New probe point in SGNUC, after SVC 107 path is entered.

Exit Type SSTATE

RID SYSTEMID (=0, no page faults allowed)

Input IJBVINF points to the communication area. The area is contained in the SVA, RMODE is ANY. The following information is passed to the exit routine.

- SVC old PSW
- SVC interruption code
- RID of interrupted task
- TID and PIK of interrupted task
- Register and access register at time of interruption

Output A return code is set up in IJBVRC:

00	continue normal SVC handling
12	Only for 'UPDATE' exits (IJBVUPD). Cancel user due to security violation. Only allowed when SVC issuer (user) had RID=USERTID. Hardwait otherwise. In case of security violation, the vendor exit can pass additional information in the prodexit extension area, field IJBVFVIN. This information will appear in the cancel message.

Vendor Exit - VSE/AF Basic Access Method (Class = BAM)

Description *:*

Following is a description of the possible vendor exits for class BAM:

Pre-OPEN subclass: IJBVBPRO in IJBVSCL
Pre-CLOSE subclass: IJBVBPRC in IJBVSCL
Post-OPEN subclass: IJBVBPOO in IJBVSCL
Post-CLOSE subclass: IJBVBPOC in IJBVSCL
EOX processing for sequential disk
subclass: IJBVBEOX in IJBVSCL
Common VTOC handler
subclass: IJBVBCVH in IJBVSCL

Invocation

If the exit is invoked in a B-transient phase, the logical transient area is freed before the exit is taken.

Before the free-up of the LTA, the 'PRODEXIT ACTIVATE' service has to save LIOCSCOM in IJBVBCRI and the data at LTA+X'4A4' of length 16 in IJBVBLTA and LIOCSCOM has to be cleared to binary zeros. The last 'PRODEXIT RETURN' has to restore LIOCSCOM and LTA+X'4A4'. The B-transient to which the 'PRODEXIT RETURN' has to return is provided in IJBVBNXT by the 'PRODEXIT ACTIVATE' requestor.

Note: IJBVBLTA, IJBVBCRI and IJBVBNXT are fields in the BAM class extension pointed to by IJBVCEXT.

Pre OPEN (Subclass = PREOPEN)

Location at the beginning of OPEN, just before the system does any action with the DTF/ACB ($$BOPEN1).

Exit Type PSTATE

PRODEXIT and input area RMODE ANY, partition GETVIS

Input IJBVINF points to the input area, containing the following information:

- pointer to the DTF/ACB (4 bytes), IJBVBDTF
- Flag byte IJBVBFLG
  - X'80', IJBVBJCT, indicates request is from JCL

Output return codes from vendors in IJBVRC

00 continue with normal OPEN processing for this DTF/ACB

04 ignore OPEN of this DTF/ACB. No system action is done with this DTF/ACB.

12 change of DTF/ACB address requested. The DTF/ACB address in users parameter list will be replaced by the address returned by the exit in IJBVBDTF and OPEN continues with this DTF/ACB.

Note: If a vendor exit passes an invalid return code at return, the system continues normal processing.

To provide the Pre-OPEN exit for each DTF/ACB before any system action is done, the security check has to be done per DTF/ACB and only if the vendor return code is zero. Therefore the interface to DTSECAPP routine is changed, general register 1 now contains the address of the DTF/ACB to be checked.

Note: Prior to VSE/ESA 2.1.0, the security check has been done once per OPEN invocation on the whole DTF/ACB-list provided by the OPEN macro and DTSECAPP has been invoked with general register 0 pointing to the DTF/ACB-list.

Pre CLOSE (Subclass = PRECLOS)

Location at the beginning of CLOSE, just before the system does any action with the DTF/ACB ($$BCLOSE).

Exit Type PSTATE

PRODEXIT and input area RMODE ANY, partition GETVIS

Input IJBVINF points to the input area, containing the following information:

- pointer to the DTF/ACB (4 bytes), IJBVBDTF

Output return codes from vendors in IJBVRC

00 continue with normal CLOSE processing for this DTF/ACB

04 ignore CLOSE of this DTF/ACB

08 perform **no** rewind, only for DTFMT and DTFPH for tape

12 change of DTF/ACB address requested.
The DTF/ACB address in users parameter list will be replaced by the address returned by the exit in IJBVBDTF and CLOSE continues with this DTF/ACB.

Note: If a vendor exit passes an invalid return code at return, the system continues normal processing.

Post OPEN (Subclass = POSTOPEN)

Location at the end of OPEN process, before the next DTF will be handled or before return to user via SVC 11 if the last or only DTF has been handled ($$BOPEN1).

Exit Type PSTATE

PRODEXIT and input area RMODE ANY, partition GETVIS

Input IJBVINF points to the input area, containing the following information:

- pointer to the DTF (4 bytes), IJBVBDTF
- Flag byte IJBVBFLG
  - X'80', IJBVBJCT, indicates request is from JCL

Output return codes from vendors in IJBVRC

00 continue with normal OPEN processing

12 change of DTF address requested.
The DTF address in users parameter list will be replaced by the address returned by the exit in IJBVBDTF and normal processing continues.

Notes:

1. If a vendor exit passes an invalid return code at return, the system continues normal processing.
2. Post OPEN exit invocation is only guaranteed for DTFMT, DTFPH for tape and DTFs for disk. The Invocation for other DTFs is unpredictable.
3. Post Open exit does not get control for VSAM managed SAM files.

Post CLOSE (Subclass = POSTCLOS)

Location at the end of CLOSE process, before the next DTF will be handled or before return to user via SVC 11 if the last or only DTF has been handled ($$BCLOSE).

Exit Type PSTATE

PRODEXIT and input area RMODE ANY, partition GETVIS

Input IJBVINF points to the input area, containing the following information:

- pointer to the DTF/ACB (4 bytes), IJBVBDTF

Output return codes from vendors in IJBVRC

00 continue with normal CLOSE processing

12 change of DTF/ACB address requested. The DTF/ACB address in users parameter list will be replaced by the address returned by the exit in IJBVBDTF and normal processing continues.

Notes:

1. If a vendor exit passes an invalid return code at return, the system continues normal processing.
2. Post CLOSE exit invocation is only guaranteed for DTFMT, DTFPH for tape and DTFs for disk. The Invocation for other DTFs is unpredictable.

EOX Processing (Subclass = BAMEOX)

Location After EOX is recognized for sequential disk and before the next EXTENT if any is obtained. The exit is **not** invoked for end of extent during CLOSE.

Exit Type PSTATE

PRODEXIT and input area RMODE ANY, partition GETVIS

Input IJBVINF points to the input area, containing the following information:

- pointer to the DTF (4 bytes), IJBVBDTF

Output none

CVH Processing (Subclass = BAMCVH)

Location At the very beginning of the common VTOC handler (IJJHCVH0).

Exit Type PSTATE

PRODEXIT and input area RMODE ANY, partition GETVIS

Input IJBVINF points to the input area, containing the following information:

- address of CVH parameter list IJJHCPL (4 bytes), IJBVBCPL

Output return codes from vendors in IJBVRC

00 continue with normal Common VTOC Handler processing

04 immediate return to caller. In this case it is the vendors responsibility to maintain the Common VTOC Handler interface to the caller.
Return code 4 is not allowed for OPEN and CLOSE VTOC requests.

Note: If a vendor exit passes an invalid return code at return, the system continues normal processing.

Messages And Codes:

If no partition GETVIS is available for Pre/Post-OPEN or Pre/Post-CLOSE to get the PRODEXIT areas for LTA processing, the existing message 4n79I 'GETVIS FAILED RC=nnn' will be issued with a new OPEN/CLOSE return code.

4n79I **GETVIS FAILED RC=*nnn*...**
Explanation: ...
RC=0B The required space for PRODEXIT area is not available in partition GETVIS.
...
System Action: ...
Programmer Response: ...
Operator Response: ...

Vendor Exit - VSE/AF Console Support (Class = DOCP)

Full Scale Input Processing Exit

This exit allows to monitor every console input (entered on the command line), every timer interrupt, reconnect, message pending, action message received, alert posted, suspended console and unrecoverable error.

Class DOCP

Subclass FINPUT

Location The exit is invoked for every user input in console mode and for other events, as indicated in field CEXEVNT. The exit has the option to take over control of console output ('vendor mode'). In this mode, the screen is fully managed by exit code, while input is still handled by standard code and passed to the exit, as in concole mode.

Exit Type PSTATE

RID USERTID

Input IJBVINF points to the communication area(input and output area), which is described by the IJBCNMEX mapping macro. The area holds the following information at exit entry:

- Addresses of screen and message definitions
  Mappings defined in IJBDEF macro
  - address of panel data table
  - address of PF-key data table
  - address of message data table
- Terminal characteristics
  - number of screen lines
  - number of screen columns
  - DBCS capability: 'Y' or 'N'
  - ext data stream capability: 'Y' or 'N'
  - number of supported colors
  - supported color attributes
  - number of supported highlites
  - supported highlite attributes
- Event flags
  Flags CEXEMSGP, CEXEALRT and CEXESUSP, once set, remain on until the exit routine leaves vendor mode. These flags may therefore be set in combination with other flags, which are mutually exclusive:
  - user input
  - timer interrput
  - action/decision message received
  - reconnect (after PF3/PF4)
  - message pending
  - alert posted
  - console suspended
  - unrecoverable error

Output

The communication area pointed to by IJBVINF holds the following information at exit exit:

- Processing flags, timer interval
- The options 'use updated input' and 'enter vendor mode' are only processed for 'user input' events.
- Since the screen is controlled in vendor mode by the exit, the exit is also responsible for issuing a write, that unlocks the keyboard, for every user input passed to it
- When entering vendor mode, any pending timer interrupt is cancelled.
- The option 'set timer' may only be specified in conjunction with 'enter vendor mode' or while vendor mode is active. When specified in vendor mode, any already scheduled interrupt is cancelled. A new interrupt is set only if the value passed in CEXPTIMS is > 0.

Vendor Exit - VSE/AF Console Support (Class = DOC)

Message Processing Exit

This exit allows to monitor every message, issued via WTO/WTOR or SVC 0/15, to update control information and text or to suppress or automatically reply to the message. It is not intended for intercepting/redirecting messages through an alternate queueing mechanism.

Class DOC

Subclass MSG

Location The exit is invoked after converting the message to a common format, including the message prefix, and before queueing the message for delivery. If OCCF message processing is active, it is also invoked before the exit.

Exit Type SSTATE

RID SYSTEMID (no page faults allowed, emergency messages only) or REENTRID (page faults allowed)

Input/Output IJBVINF points to a parameter list, described in macro IJBCSMX and passed in turn to each exit, with following contents:

- Message parameters that cannot be changed (like originating jobname).
- The address of a message area, also described by IJBCSMX and containing data that may be updated:
  - Name of the target console
  - Routing and descriptor codes
  - Presentation attributes
  - Message text lines (space for up to 12 lines is provided)

  This area is initially filled from original message data.
- The address of an automatic reply area, consisting of a 2-byte length followed by up to 126 bytes of reply text. This area is initially empty (length field is zero). An automatic reply must be prepared in this area, as if it was entered from a console (but without the preceding reply-id). An automatic reply may be provided also when no 'read' was yet issued. In such a case, the reply is processed automatically for a following stand-alone read, a following stand-alone WTOR (that is, a WTOR with a 'just-one-blank' text) or discarded otherwise.
- A flag byte for requesting one of the following processing options:
  - Suppress the message
  - Reply to message automatically
  - Route message to a console with AUTO attribute
  - Update message control information
  - Update message text

The exit is not invoked for DOM requests.

In some cases, e.g. when no queue space is immediately available for an extended message, the exit may be invoked multiple times for the same message, with identical initial message contents.

Command/Reply Processing Exit

This exit allows to monitor every input, entered via MGCRE or SVC 30, and to reject it or to update input text.

Class	DOC
Subclass	CMDREP
Location	The exit is invoked after converting console input to a common format, and before processing or queueing it for delivery. If OCCF input processing is active, it is also invoked before the exit.
Exit Type	SSTATE
RID	REENTRID (page faults are allowed).
Input/Output	IJBVINF points to a parameter list, described in macro IJBCSMX and passed in turn to each exit, with following contents: • Input parameters that cannot be changed (like originating console name). • The address of an input area, consisting of a 2-byte length followed by up to 126 bytes of input text. This area is initialized with the original input. • A flag byte for requesting one of the following processing options: – Reject the input – Update input text

The exit is not invoked for automatic replies, generated by a message processing exit nor for EXPLAIN request.

Vendor Exit - VSE/AF Attention Routine (Class = AIT)

Command Processing Exits

The Attention Routine supports two exits, that allow to customize existing system commands or to add new commands. They are not intended for modifying command input (the CMDREP exit described in “Vendor Exit - VSE/AF Console Support (Class = DOC)” on page 155 should be used for this purpose).

Note: Scope=system is the only valid 'SCOPE' for the CLASS AIT exits. RC=12 will be posted in PRODEXIT DEFINE if SCOPE=PARTITION / TASK has been specified.

Pre-Scan Exit

Class	AIT
Subclass	CMD1 (Pre-Scan)
Purpose	This exit allows to intercept all IBM Virtual Storage Extended system commands, that are processed by or routed through the Attention Routine, and to provide customized versions of such commands.
Location	The exit is invoked before AR parses the command.
Exit Type	PSTATE
RID	USERTID
Input	IJBVINF points to an area containing the command string, along with other command attributes like its origin, and described by the mapping macro MAPARCMD.
Output	Return code in IJBVRC: **0** Command was taken over by the exit. **Note:** The first 'update' exit which returns with RC=0 causes skipping of all following 'update' exit invocations. Only 'monitor' type exits will then get control prior to the AR CMD processing. An exit invocation skipping gets posted in the PRODEXIT area (bit IJBVSUPD in byte IJBVFLAG). **4** Command not recognized

Post-Scan Exit

Class AIT

Subclass CMD2 (Post-Scan)

Purpose This exit allows to define and support new commands, that are routed through the Attention Routine but are not recognized as known IBM Virtual Storage Extended system commands.

Location The exit is invoked when the command is not recognized as an AR or active subsystem command.

Exit Type PSTATE

RID USERTID

Input IJBVINF points to an area containing the command string, along with other command attributes like its origin, and described by the mapping macro MAPARCMD.

Output Return code in IJBVRC:

0 Command was taken over by the exit.
Note: the first zero return code from an exit will cause any further exits to be skipped

4 Command not recognized

VSE/ESA LNG Exit

"How to Use this Exit" on page 163 describes the various approaches to using this exit.

Class LNG

Subclass LNGOPEN

Purpose The exit allows a disk or tape manager to intercept file open processing for LE/VSE-enabled languages, and allows it to provide some information or cause it to bypass some of the pre-open checks. These languages perform some pre-open checking to enable them to return correct statuses to the programs. When a disk or tape manager provides some information, such as logical unit, during open, the language checks will fail, and the open is not issued. This exit allows this information to be provided to the language, or to request the language to bypass these checks. If these checks are bypassed, some of the statuses returned may be incorrect.

Location The exit is invoked by LE/VSE when the LE/VSE Message file is opened, or when an open is requested in an LE/VSE-enabled language (COBOL/VSE or PL/I VSE). The exit is invoked before building the DTFSD, DTFDA, DTFDU, DTFMT, DTFPR, or DTFCD, and prior to opening the file. It is also invoked by the COBOL/VSE compiler prior to checking the logical unit for the compiler work files.

Notes:

1. If errors are detected after the exit is invoked, the OPEN is not issued.
2. The exit is not invoked for files that are accessed using an ACB.

Exit Type PSTATE

RID USERTID

Communication area Partition GETVIS, RMODE=ANY

Input and Output IJBVINF points to the communication area (input and output area). The area holds the information shown in the following table at exit entry.

The Type column contains "Input" if it is provided by the language, "Output" if it is provided by the exit, or "Input/Output" if it is provided by the language but may be modified by the exit.

Field	Size	Type	Description
IJBVLENV	H	Input	Length of area
IJBVPIK	H	Updated by supervisor	PIK of current task

Field	Size	Type	Description
IJBVTIK	H	Updated by super-visor	TIK of current task
Reserved	H		Reserved
IJBVLLVL	F	Input	Level number of the parameter list.
IJBVLANG	CL8	Input	The Language product that is invoking the exit, for example, COBOL, PL/I, LE.
IJBVLNAM	CL8	Input	The name from the COBOL ASSIGN statement, or PL/I File DECLARE statement.
IJBVLLUN	H	Input	The logical unit number from the COBOL ASSIGN statement, or PL/I File DECLARE statement, or zero if not provided.

IJBVLOPM	X	Input	Open mode with values as follows:
			`IJBVLIN  X'01' - Input`
			`IJBVLIO  X'02' - I/O (for example, TYPEFLE=INPUT and UPDATE=YES)`
			`IJBVLOUT X'03' - Output`
			`IJBVLEXT X'04' - Extend (append)`
			`IJBVLWRK X'05' - Work (for example, compiler work file)`

IJBVLDEV	X	Input/ Output	Device type for file with values as follows:
			`IJBVLDSD X'01' - SAM disk`
			`IJBVLDDA X'02' - DAM disk`
			`IJBVLDVS X'03' - VSAM disk`
			`IJBVLDUT X'04' - Unlabeled tape`
			`IJBVLDLT X'05' - Labeled tape`
			`IJBVLDCD X'06' - Card`
			`IJBVLDPR X'07' - Printer`
			`IJBVLDUA X'08' - Unassigned (UA)`
			`IJBVLDIG X'09' - Assigned to IGNORE (IGN)`
			This field can be modified by the exit to change the device type of the file. For instance, a labeled tape can be changed to an an unlabeled tape or a disk file can be changed to a tape file. This field should not be modified for DAM files or files with an open mode (IJBVLOPM) of WORK (IJBVLWRK).
			Note: The device type is set to SAM, DAM, or VSAM-managed SAM when a DLBL is present. It is set to labeled tape when a TLBL is present. It is set to one of the other device codes when neither a DLBL or TLBL is present, according to the type of device to which the logical unit number (IJBVLLUN) is assigned.

IJBVLLBA	A	Input/ Output	The address of the DLBL or TLBL retrieved by the language product. This is present for Disk, or labeled tape files. The fields within the DLBL or TLBL can be altered as required. The length of the DLBL or TLBL (IJBVLLBL) can be increased up to length of the buffer containing the DLBL or TLBL (IJBVLLBB). The address can be changed to the address of an area acquired by the exit, and the length (IJBVLLBL) updated to reflect the new length. The fields that are currently used from the DLBL are as follows: • File ID (if not provided in field IJBVLFID). • Logical unit number (for SAM DLBL if not provided in field IJBVLLUO) • Blocksize (for SAM DLBL if not provided in field IJBVLBSO) • Catalog name (for VSAM DLBL) The fields that are currently used from the TLBL are as follows: • File sequence number (for multi-file tape).

Field	Size	Type	Description
IJBVLLBL	F	Input/ Output	The length of the DLBL or TLBL retrieved by the language product using the `LABEL FUNCT=GETLBL` macro.
IJBVLLBB	F	Input	The length of the buffer containing the DLBL or TLBL retrieved by the language product using the `LABEL FUNCT=GETLBL` macro.

IJBVLOPT	XL2	Input/ Output	Other open options.
			`IJBVLORV x'80' - OPEN REVERSED specified`
			`IJBVLNRW x'40' - OPEN NO REWIND specified`
			`IJBVLASC x'20' - ASCII tape file`
			`IJBVLOPF x'10' - OPTIONAL file (COBOL only)`
			`IJBVLCBM x'08' - MODE=C (column binary mode)`
			`(3505 and 3525 card devices only)`
			`IJBVLOMR x'04' - MODE=O (Optical mark read)`
			`(3505 card devices only)`
			`IJBVLRCE x'02' - MODE=R (Read Column Eliminate)`
			`(3505 and 3525 card devices only)`

IJBVLPUB	X	Input/ Output	Pub code This field is provided by the language product for unlabeled tape, card and printer devices, and may be modified by the exit. It is used to determine the device type when a DTF is built.
IJBVLFEX	X	Output	File exists flag set by exit to X'01' if file currently exists, or X'02' if file does not exist. If it is not set (left as X'00'), and the file is being opened for input or I/O, the language attempts to determine if the file exists.
IJBVLLUO	H	Output	The logical unit number determined by the exit. If it is provided, the subsequent processing by the language uses this logical unit number. If it is not provided for a SAM file, the language attempts to determine the logical unit from the DLBL or COBOL ASSIGN statement or PL/I File DECLARE statement.
IJBVLFID	CL44	Output	This is the file ID for the file, with any vendor-supplied tape or disk manager control characters removed. If it is not provided, the language uses the file-id from the DLBL or TLBL statement for a SAM file.
IJBVLLEX	H	Output	Extent number of the last volume. This is applicable to SAM files only. If it is not set (left as zero), the language attempts to determine the extent number of the last volume from the DLBL/EXTENT statements. The extent number is used by COBOL when the CLOSE REEL/UNIT statement is used. **Note:** If the vendor-supplied tape or disk manager provides extents as required, this field can be set to -1. This disables the CLOSE REEL/UNIT statement for COBOL.

IJBVLRFI	H	Input	Record format from program (zero if not provided) as follows:
			`IJBVLRFF x'80' - Fixed`
			`IJBVLRFV x'40' - Variable`
			`IJBVLRFU x'20' - Undefined`
			`IJBVLRFB x'10' - Blocked`
			`IJBVJRFS x'08' - Spanned`
			`IJBVLRFA x'04' - ASA control characters`
			`IJBVLRFM x'02' - Machine control characters`
			For example, x'90' for Fixed Blocked.

IJBVLRSI	F	Input	Record length from program (zero if not provided)
IJBVLBSI	F	Input	Block size from program (zero if not provided)

Field	Size	Type	Description
IJBVLRFO	H	Output	The record format for the existing file if available, with the same flags as IJBVLRFI. If it is not set (left as zero), the file is being opened for input or I/O, and it is a VSAM-managed SAM file, the language determines the record format from the VSAM catalog. For a file opened for Input, I/O, or Extend, the record format from the program is checked to ensure it matches the record format for the input file.
IJBVLRSO	F	Output	The record length of the existing file if available. If it is not set (left as zero), and the file is being opened for input, I/O, or EXTEND, and it is a VSAM-managed SAM file, the language determines the record length from the VSAM catalog. For a file opened for Input, I/O, or Extend, the record length from the program is checked to ensure it matches the record length for the input file. If the record length is not specified in the program (for example, RECORD CONTAINS 0 CHARACTERS for COBOL), the record length provided in this field is used (if non-zero).
IJBVLBSO	F	Output	The block size of the existing file or output file. If it is not set (left as zero), the language attempts to determine the block size. For a VSAM-managed SAM file, the VSAM catalog is checked. For a SAM file, the block size from the DLBL is used if present. This block size overrides the block size specified within the program for a disk or tape file with the record format specified as blocked. If the record format is fixed, the block size must be a several of the record size.
IJBVLLMA	A	Output	The address of the LIOCS logic module to be placed in the DTF after open. This may be used to replace the IBM-supplied logic module.
IJBVLLMS	A	Output	The address of a fullword to return the address of the logic module in the DTF prior to being overwritten by the logic module address supplied above.

A return code is set up in IJBVRC:

00 Continue processing.

04 Continue processing, but bypass any pre-open checks. The open processing continues if the following errors are detected.

- No EXTENT card has been provided for a SAM DLBL, and the logical unit number was not provided on the COBOL ASSIGN statement or PL/I File DECLARE statement.
- The logical unit is not assigned.
- The file has been opened for Input or I/O, and the file does not exist.

Other Don't continue processing, and fail the open.

How to Use this Exit

Depending on how much effort a Vendor may wish to expend, and the capabilities of the Vendor product, the exit can do one of the following:

Approach 1 - Minimal Approach

The minimal approach would be for the Vendor Exit to set IJBVRC to 4. This causes the languages to skip the checking of the device. This would allow the files to open, but the restrictions that are currently present when the Vendor products are present would still apply.

Some of the known restrictions are as follows:

1. Support for OPTIONAL files in COBOL is not provided.
2. Some file statuses under COBOL are not returned correctly.
3. Support for CLOSE REEL/UNIT in COBOL is not provided.

Approach 2

The above restrictions are caused when the language product cannot determine whether the file exists, or cannot determine the file attributes. The language product cannot perform these checks because the file-id on the DLBL or TLBL might not be the actual file-id, or because the logical unit number is not available.

The exit could perform the updating of the DLBL or TLBL statement including the assignment of the logical unit during the exit processing instead of during the OPEN. If the file-id on the DLBL or TLBL statement is not the actual file-id, the actual file-id should be returned.

This would involve setting the following fields:

- Logical unit number
- File-id (if the file-id on the DLBL or TLBL does not match the actual file-id)
- File exists flag (Input and I/O only)
- Number of volumes

Approach 3

In addition to the processing for the previous approach, if the vendor product has information about the file such as record format, record length, and block size that is not available directly from the VSE/ESA operating system, the record format, record length, and block size can also be returned.

COBOL has the ability to enable a program to read a file with fixed length records of any length (RECORD CONTAINS 0 CHARACTERS), or with any block size (BLOCK CONTAINS 0 RECORDS). If this information is returned by the exit, it enables a COBOL program to read these files. This information is currently retrieved from the VSAM catalog by COBOL for VSAM-managed SAM files.

PL/I has the ability to enable a program to read a file with any record format, record length, and block size. If this information can be returned by the exit, PL/I could then open the file without requiring the program to specify this information.

This approach provides additional capabilities to the COBOL and PL/I languages when the vendor-supplied tape or disk manager is operating and able to provide the additional information. This brings the VSE versions of COBOL and PL/I closer to the capabilities provided by the MVS Operating system.

PRODID

The PRODID services are offered for programs - primarily system related programs - to notify the system, receive and give up the right to use some services, and delete its notification at termination.

The means of communication is an 8-byte token, received at notify, invalidated at deletion, as passport to use services up to now only available for IBM subsystems. A program may also check whether it was already initialized in the system or if a known other program - maybe incompatible - was initialized.

PRODID DEFINE Macro

Its purpose is twofold, it send the program's “visiting card” to the system and the system honors the card with a token given to the program. The information on the visiting card, later on called notification entry, is kept by the system, until removed by the program, for the benefit of all service organizations looking at a dump and wondering which programs might be involved in a certain problem.

The token given to the program is a key which enables the program to use a list of services up to now available for IBM subsystems only. As a key to a room may be lent to other people this token may be passed from the program obtaining it to routines executing under a different program but supported by the token owner. An example would be a program supplying a kind of access method to other programs. The access method would be initialized in its partition, obtain the token, and deposit it in an area shared with the supported programs. Then the token would be used whenever the routines of the access method, working for a supported program, need to be enabled for use of a service or disabled again if no longer needed.

PRODID DEFINE is issued in the initialization routine of a program to notify the system. It must be issued while the program is running in its own partition. The user, starting any program issuing DEFINE must have update right for phase IJBVEND residing in IJSYSRS.SYSLIB. This means the system administrator must have defined this user ID and the right in DTSECTAB. How to update DTSECTAB is described in the manual *VSE/ESA Guide to System Functions*. The phase IJBVEND is the part of the supervisor processing the PRODID request.

The input for PRODID DEFINE, the text of the “visiting card” is a string described by the PRODID DSECT macro. This input should be in character format so that no conversion is needed for printing the string. The input will be accepted as passed along, except it must not be empty, that means, none of the three fields IJBCOMPN, IJBPRODN, IJBVRM must be binary zero. In such case the request is rejected with a return code.

In return to this string passed on PRODID DEFINE the system will assign a token. Should the same string exists already in the system, further processing will depend on whether the program indicates that it will run only once in the system (parameter RUN=ONE). This is the only case supported now and aimed at the type of program working as for example, a monitor of some system activity.
In such a case the same token will be returned as before and an appropriate return code set. Please note the token is associated with the input string, not with a task or partition. It will be deleted as the result of an explicit PRODID DELETE request, not with any end-of-task or end-of-job processing.

Not yet supported:

For a program indicating that it may run concurrently in different partitions the processing is slightly different:

- The same token will only be returned when the request comes from the same partition.
- The token will be invalidated and the program's notification entry deleted - if still around - at end-of-job step.

Requirements for the Caller:

- Authorization: controlled by Access Control Facility
- AMODE: ANY
- RMODE: ANY
- ASC-mode: Primary

Invocation:

```
[label] PRODID   DEFINE,AREA= { (1) | name } [,RUN={ ONE| MULTIPLE } ]
```

Input Parameter Description:

AREA= points to an area where the input information is available. It may be specified in register notation or as the name of the input area. The area must be in an address range valid for this job.

The fields for identifying the company, the product and the release level should carry the following information.

IJBCOMPN should contain the name of the company owning the product issuing the DEFINE request. It may be a suitable, recognizable abbreviation, and it should always be the same one used across all products owned by this company.

IJBPRODN should contain the name of the product or a suitable, recognizable abbreviation, and it should be the same one across all different releases of the product.

IJBVRM contains the version, release, modification level of the program

The information in these different fields may use more or less than the length proposed by the DSECT. But it must not use more then the combined length of IJBCOMPN to IJBVRM. An example for this may be a complicated numbering scheme for indicating the release level which needs more than the 6 bytes allotted to IJBVRM. In this case the release level must start somewhere in IJBPRODN so its end coincides with the end of IJBVRM.

Note: The CHECK function does not support such overflowing information, but treats for example, the field IJBCOMPN as containing the company name and nothing else.

RUN= ONE indicates the program will run only once per system as for example, VSE/POWER. This is the default, and right now, the only supported value.

MULTIPLE indicates that the program may be run in different partitions concurrently as for example, CICS/VSE. Not supported yet.

Output:
is an 8 byte token passed back in the input area at label IJBTOKEN.

On return register 0 will be destroyed.

Return Codes:
passed back in register 15

0 The system accepted the input and returned a token

4 The same string is already defined for the system or partition, the same token was returned as previously

8 Maximum of 256 products already defined

12 Control block format error. At least one of the fields identifying company, product, or release level is binary zero, or IJBVIDL is incorrect.

16 IJBVEND not loaded.

20 GETVIS error, return code from GETVIS in register 0

Cancel Conditions:

X'0B' security check failed

X'21' an unknown function code is passed to PRODID (corrupted macro expansion)

X'25' invalid address of parameter list or parameter

PRODID DSECT Macro

This macro generates a DSECT describing:

- input expected by PRODID DEFINE
- output returned by PRODID DEFINE
- input expected by PRODID CHECK
- output returned by PRODID CHECK.

For a detailed discussion of the value of these fields see “PRODID DEFINE Macro” on page 165 and “PRODID CHECK Macro” on page 169 respectively.

The layout is:

```
IJBVIDL  DS    AL2(IJBFINST) LENGTH OF INPUT AREA
IJBVIFL1 DS    CL1           FLAGBYTE, RESERVED
IJBVIFL2 DS    CL1           FLAGBYTE, RESERVED
IJBCOMPN DS    CL14          NAME OF COMPANY, PROGRAM OWNER
IJBPRODN DS    CL16          NAME OF PROGRAM
IJBVRM   DS    CL6           VERSION AND RELEASE OF PROGRAM
IJBTOKEN DS    CL8           TOKEN RETURNED BY DEFINE
IJBCKLEN DS    CL2           LENGTH OF OUTPUT AREA FOR CHECK
IJBFINST EQU   *-IJBVIDL
IJBCKARE EQU   *             BEGIN OF OUTPUT AREA FOR CHECK
IJBCOMPO DS    CL14          NAME OF COMPANY, PROGRAM OWNER
IJBPRODO DS    CL16          NAME OF PROGRAM
IJBVRMO  DS    CL6           VERSION AND RELEASE OF PROGRAM
IJBTIDO  DS    CL2           TASK ID RETURNED BY CHECK
IJBPIKO  DS    CL2           PARTITION ID RETURNED BY CHECK
```

Input Parameters:
None.

Output:
None.

Return Codes:
None

PRODID AUTH Macro

This macro enables a program with a valid token to obtain the right to use the services listed below, or to give up this right. The system uses a fast path for processing of this request. The right is granted to the task issuing the request, and will be removed at end-of task if not already revoked. This need not be the same task having notified the system by PRODID DEFINE and obtaining the token.

This gives non-IBM software - or even an IBM program not included in SUBSID support - a status comparable to an IBM subsystem as for example, CICS. This also means the programs must use utmost care when using the services listed below. The checking in these services is not as extensive as required for a user interface. Therefore, any error in calling the services - for instance with incorrect parameters or at the wrong point in time - may damage the system.

Note: Each AUTH=YES request increases a one byte counter attached to the issuing task, each AUTH=NO request decreases it by one.

Services available:

- GETFLD FIELD=ALET (add to DUAL of current task)
- MODFLD FIELD=PASCOPE1
- MODESET MODE=SUP
- EXTRACT ID=PART
- EXTRACT ID=SVA
- TREADY COND=ICCF
- TREADY COND=NO
- PRODEXIT
- SYSDEF
- XMOVE

Requirements for the Caller:

- Authorization: valid token
- AMODE: ANY
- RMODE: ANY
- ASC-mode: Primary or AR

Invocation:

```
[ label ] PRODID  AUTH= { YES | NO }
```

Input Parameter Description:

Registers 1 and 2	must contain the token previously obtained by PRODID DEFINE
AUTH=	YES requests the right to use special services. If the token passed in registers 1 and 2 is valid, this right is granted. NO gives up the right.

Output:
None.

On return register 0 will be destroyed.

Return Codes:

passed back in register 15

0 right granted for AUTH=YES
right set off for AUTH=NO

4 right revoked for AUTH=NO, but was already revoked or never requested

8 right not granted, more than 255 consecutive AUTH=YES requests for this product within this task.

16 IJBVEND not loaded

Cancel Conditions:

X'21' invalid token
or an unknown function code is passed to PRODID (corrupted macro expansion)

PRODID CHECK Macro

PRODID CHECK searches the accumulated notifications entries for either a certain, known product, or all releases of such a product, or all products of a certain company, and returns any notification entries found. Also, notification entries of all programs may be retrieved which is useful information when trying to determine the cause of a problem.

With this macro the program checks for a known string.

All occurrences found are moved to the user's area starting at IJBCKARE if the length of this area is sufficient. If the length is not sufficient, a return code indicates this fact and the total number of matching entries is returned in register 0. Then the request may be repeated with a larger area.

Requirements for the Caller:

- Authorization: none
- AMODE: ANY
- RMODE: ANY
- ASC-mode: Primary

Invocation:

```
[label] PRODID     CHECK,AREA= { (1) | name }
```

Input Parameter Description:

AREA= points to the area where the search information is available.
The area must be in an address range valid for this job.
The layout of this string is described by PRODID DSECT. The following values may be passed as input:

- IJBCOMPN contains the name of the company owning the product which is looked for.
  If this field contains binary zeros then all notification entries are returned.
- IJBPRODN contains the name of the product which is looked for. If this field is set to binary zeros all notification entries with the same company name are returned.
- IJBVRMN contains the version, release, modification level of the program checked for. If this field is set to binary zeros all notification entries with the same company name and product name are returned.
- IJBCKARE supplied in the input pointed to by AREA is the begin of the output area. In this area the matching notification entries are moved provided they fit.
- IJBCKLEN specifies the length of the output area. The output area must be in an address range valid for this job.

Output:
The part from IJBCOMPN to IJBVRM of the found notification entry or entries - including the TIK and PIK of the partition where PRODID DEFINE was issued first - is moved to the area starting at field IJBCKARE in the length specified in IJBCKLEN. If the defining task terminated in the meantime the fields for TIK/PIK are set to binary zeros. See also “PRODID DSECT Macro” on page 167 for the layout of one returned notification entry.

On return register 0 will contain the number of entries found.

Return Codes:

codes passed back in register 15

- 0 entry/ies matching the search criteria was/were moved to user area.
- 4 entry/ies matching the search criteria was/were moved to user area. But there are more matching entries which could not be moved because the output area was full. Register 0 contains the number of all matching entries, so repeat request with larger area.
- 8 no match found or no entry existing
- 12 Control block format error (for example, incorrect IJBVIDL)
- 16 IJBVEND not loaded.

Cancel Conditions:

X'21' an unknown function code is passed to PRODID (corrupted macro expansion)

X'25' invalid address of parameter list or parameter

PRODID DELETE Macro

With this macro the program requests to delete its notification entry and invalidate its associated token. This should be used in the program's termination routine.

Requirements for the Caller:

- Authorization: valid token
- AMODE: ANY
- RMODE: ANY
- ASC-mode: Primary

```
[label] PRODID  DELETE, TOKEN={ (1) | name1 }
```

Input Parameter Description:

TOKEN points to the token previously obtained by PRODID DEFINE. The area must be in an address range valid for this job.

Output:

None

On return register 0 will be destroyed.

Return Codes:

passed back in register 15

0 entry deleted, AUTH=NO was issued before DELETE request

4 entry deleted, but no AUTH=NO was issued before DELETE. Authority is removed.

8 entry deleted. Authority was already reset before the DELETE request was issued. At least one PRODEXIT DEFINE has been removed

12 entry deleted. Authority was still granted. Authority has been removed. At least one PRODEXIT DEFINE has been removed.

16 IJBVEND not loaded.

Cancel Conditions:

X'21' passed token was invalid. This could be caused by a second DELETE request.
Or an unknown function code is passed to PRODID (corrupted macro expansion).

X'25' invalid address of parameter list or parameter

READHCF

The macro READHCF ensures that one HCF record is provided in the I/O area, specified by the second operand. The direction of the READ operation depends on the associated POINTHCF macro which implies this information or it depends on the last MODHCF macro given (if any).

The macro has the following format:

```
[name]   READHCF  {(hcfreg)|(1)},{ioarea|(0)}
```

hcfreg Is the general register containing the address of the HCFCB control block returned by the corresponding POINTHCF macro.

ioreg Is the symbolic name of the I/O area to where the HCF record is to be moved.

Output: Register 15 contains one of the following return codes.

0 (X'00')	Normal processing successfully completed.
4 (X'04')	Inconsistent input.
8 (X'08')	No record found, incorrect length.
12 (X'0C')	Unrecoverable I/O error.
16 (X'10')	HCF device is not ready.
20 (X'14')	Begin of HCF (first record in file).
24 (X'18')	End of HCF (last record in file).
28 (X'1C')	Record has already been overwritten by a new one.

Register Usage: The contents of general register 14 through 2 are destroyed by this macro.

REIPL

The macro REIPL allows special subsystems to request software Re-IPL.
If the program issuing the Re-IPL macro has no authorization, it is canceled with illegal SVC.
The macro has the following format:

```
[name]   REIPL [DEVICE=PRIMARY|ALTERNATE|CURRENT]
               ,SYMPTOM={name1|(r1)}
               ,IDUMPNO={name2|(r2)}
               [,ACTION=ERROR|RESET]
```

DEVICE Specifies the device to be used for IPL. The unit address is taken from the IPL command SYS. Valid parameters are:

- PRIMARY The unit specified as PRIMIPL.
- ALTERNATE The unit specified as ALTIPL.
- CURRENT The current IPL unit.

SYMPTOM Name/address of a user area where a variable length symptom string is provided. The area consists of a 2-byte length field followed by the symptom string. The specified length does not include the length of the length field. The symptom string contains those symptoms which are required to detect duplicate errors. The symptoms RIDS and PRCS are optional. The symptoms are separated by a blank character.

- PIDS SYMPTOM
  The PIDS symptom supplies the 9-byte component identification.
- RIDS SYMPTOM
  The RIDS symptom describes the piece of failing code with its CSECT name, macro name, module name or phase name.
- AB SYMPTOM and PRCS SYMPTOM
  The AB symptom supplies a one to six byte ABEND code in printable form. The VSE components are free to define their ABEND codes. A unique ABEND code should be defined for any single error source. If the already available ABEND codes are not unique, then a feedback code (PRCS symptom) may be added in order to provide unique error codes.
  The PRCS symptom supplies a one to eight byte feedback code in printable form. A PRCS symptom is required if the AB code is not unique.

IDUMPNO Name/address of an 8 byte area where the IDUMP number of the associated IDUMP has to be provided as it has been returned from the IDUMP macro invocation.

ACTION Determines the Re-IPL counting. Valid parameters are:

- ERROR
  The Re-IPL processing is controlled by counters to prevent an IPL loop.
- RESET
  The Re-IPL is not subject to counting, that means, an IPL loop may occur if not handled properly. The IPL counters are reset and a new counting cycle begins.

Note: During Re-IPL processing, the symptom string will be compared against that of the previous Re-IPL request in order to check for same reason. The result of this comparison may lead to a Re-IPL from the alternate device.

RLOCK

The RLOCK macro obtains access to a specified resource. If the resource is not available, the issuing task is set into the appropriate resource-bound condition.

The macro has the following format:

```
[name]   RLOCK COND={name1}
```

The operands have the following meaning:

COND= Specifies the resource that the requester wants to access.

ALLOCR Allows to lock or to wait for the access to fields related to the system 'real partition' and will be used by the ALLOCR sub-function of ALLOCATE.

Register Usage: R15 is used to pass the function code.

SECHECK

This macro can be used to perform several access control functions as listed:

Access Check (REQUEST=AUTH) The issuer of that macro is checked whether he is authorized to access the specified resource or not.

Service for POWER SLI Connect (REQUEST=CONNECT) This service is exclusively available for POWER. By means of this service, POWER ensures, that the access checking for SLI processing is done for the user running in the serviced partition. The JPL of the batch partition is temporarily given to POWER, so that the access checks for the members to be included via SLI INCLUDE are done for the user, that is active in the partition.

Service for POWER SLI Disconnect (REQUEST=DISCONNECT) This service is exclusively available for POWER. By means of this service, POWER returns the JPL that was requested by POWER via SLI Connect Service to Access Control.

Get User ID Service (REQUEST=GETUSER) This service is exclusively available for POWER. By means of this service, POWER gets the User ID of the specified partition.

The execution of this macro is restriced to callers in 24-bit mode and residing below the 16MB boundary. After execution of the macro register 15 contains the return code.

The macro has the following format:

```
[name]   SECHECK    REQUEST={AUTH|CONNECT|DISCONNECT|GETUSER}
                   [,AREA={name1|(r1)}]
                   [,NAME={name2|(r2)}]
                   [,PART={name3|(r3)}]
                   [,TYPE={LIB|SUBLIB|MEMBER|FILE}]
                   [,MODE={READ|UPDATE|CONNECT|ALTER}]
                   [,RETN={NO|YES}]
```

REQUEST Specifies the function to be executed. The default is AUTH, in which case a normal access check is done.

AREA Points to the Authorization Parameter List (see DTSAPL macro). It is a control block which contains information that can be specified by the other four parameters of the SECHECK macro. The length of this control block is defined in APLEN. This parameter is mandatory on all request types.

NAME Specifies in case of REQUEST=AUTH the name of the resource to be checked. The length of the name depends on the TYPE specification. For other requests, this parameter is invalid.

PART Specifies the register, which contains in the lower two bytes the SYSLOG ID of the corresponding partition. This parameter is not valid on AUTH requests.

TYPE Specifies in case of REQUEST=AUTH the type of the resource to be checked. TYPE is not valid on other requests and required when NAME is specified. The TYPE parameter can specify:

LIB The library is to be checked. The length of the resource name is 57 bytes (containing VOLID, FILE ID and library-name).

SUBLIB — The sublibrary is to be checked. The length of the resource name is 15 bytes (containing library-name and sublibrary-name).

MEMBER — The member is to be checked. The length of the resource name is 23 bytes (containing library-name, sublibrary-name and member-name).

FILE — The whole file is to be checked. The length of the resource name is 50 bytes (containing VOLID and FILE ID).

MODE — Specifies for REQUEST=AUTH the access mode of the specified resource. The parameter is invalid on other request types. Possible MODEs are:

READ — The user requires READ access

UPDATE — The user requires WRITE access

CONNECT — The user requires the authorization to ACCESS a member in a library or sublibrary. Applicable only for TYPE=LIB or TYPE=SUBLIB.

ALTER — The user requires the authorization to CREATE or DELETE a library or sublibrary. Applicable only for TYPE=LIB or TYPE=SUBLIB.

RETN — YES: Specifies that control is to be returned to the user after an access control violation.

NO: (default) Specifies, that the job is to be canceled in case an Access control violation is determined.

Output: Register 15 contains one of the following return codes, depending on the request type.

REQUEST=AUTH

0 (X'00')	Access allowed
4 (X'04')	No access control support
8 (X'08')	Access control violation
12 (X'0C')	In a protected library: the sublibrary is not in the access control resource table (DTSECTAB). In a protected sublibrary: the member is not in the DTSECTAB.
16 (X'10')	Access control function processing error. One possible reason is corrupted control blocks. Check the passed APL.

REQUEST=CONNECT

0 (X'00')	SLI preparation processing successfully completed.
4 (X'04')	No access control support
8 (X'08')	Requester is not authorized, that means, requester is not POWER
12 (X'0C')	Sequence Error. REQUEST=CONNECT followed by REQUEST=CONNECT.
16 (X'10')	Specified partition is not available.

REQUEST=DISCONNECT

0 (X'00')	SLI cleanup processing successfully completed.
4 (X'04')	No access control support
8 (X'08')	Requester is not authorized, that means, requester is not POWER.
12 (X'0C')	Sequence error. REQUEST=DISCONNECT without REQUEST=CONNECT.

REQUEST=GETUSER

0 (X'00')	Function completed successfully.
4 (X'04')	No access control support
8 (X'08')	Requester is not authorized.
12 (X'0C')	Partition is not available.

SENTER

Pass control to the entry point of a predefined SVA-resident phase of a system component and associate the component capability with the issuing task.

The address of the instruction following the macro call is passed to the entered phase in register 14, the entry point itself in register 15. SENTER is therefore equivalent to a BALR 14,15 instruction. All other registers are passed unchanged.

The macro has the following format:

```
ASSEMBLER:
 [name]   SENTER   LIBR
```

LIBR The librarian component is to be entered. The assumed entry phase is a module within phase $IJBLBR.

Register Usage:

R14 Return address
R15 Function code

SETLIMIT

The SETLIMIT macro (see also SVC 84) changes partitions sizes or sets the PFIX limits for a partition.
Execution Mode is AMODE 24 and RMODE 24 (including the parameterlist).

SETLIMIT - Size Processing

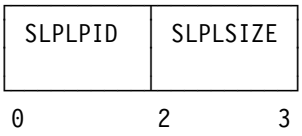
The macro is used by the job control SIZE command processing, the attention SIZE command processing, and job control EXEC statement processing, if the SIZE parameter is specified.
The SIZE command is available for static partitions only.
It may also be used by other system components if applicable.

The format is as follows:

```
[name]   SETLIMIT  [SLPL={name1|(1)}][,MODE=({PERM|TEMP},{V|R})]
```

SLPL Defines the parameter list into which specified operands have to be placed before issuing this macro. The address of the parameter list may be supplied either as an operand or in register 1.

The parameter list has the following format:



SLPLPID It specifies the partition ID 'BG', 'Fn', 'O4',... X'FFFF' means that the operand is omitted, and that the limits are reset for the partition issuing the command.

SLPLSIZE It specifies in KB the amount of contiguous virtual storage of a partition which is used for job execution. The remaining space is the partition GETVIS area. X'FFFF' means that the minimum SIZE value should be taken.

To generate the layout of the parameter list, the SLPL macro may be used:

```
[name]   SLPL  [DSECT=YES]
```

MODE indicates whether the limit is to be changed permanently or temporarily, and whether it is a virtual or real mode partition.

PERM,V The limit is to be changed permanently. A permanent limit value is applicable for a virtual partition only. It is retrieved from the SIZE operand.

PERM,R Invalid specification. For a real partition the limit can be changed temporarily only.

TEMP,V The limit is to be changed temporarily for a partition which will execute in virtual mode.

TEMP,R — The limit is to be changed temporarily for a partition which will execute in real mode. For a real partition the limit can be changed temporarily only. Its value is submitted on the job control EXEC statement.

If the MODE parameter is omitted, the parameter is expected to be supplied via register 0.

A value of zero means PERM,V; a value of one means TEMP,V; a value of three means TEMP,R.

Output: Register 15 contains one of the following return codes:

0 (X'00')	The specified limits are stored.
8 (X'08')	The new SIZE limit is not stored. The partition occupies at present dynamic storage. Re-issue the command right after End-Of-Job or End-Of-Job Step.
12 (X'0C')	The new SIZE limit is not stored. The SIZE specification does not leave the minimum GETVIS space. Reduce the SIZE value for this partition.
16 (X'10')	The new SIZE limit is not stored. The SIZE value exceeds the virtual storage of the partition. Reduce the SIZE value for this partition.
20 (X'14')	The new SIZE limit is not stored. The size of the program area specified by the SIZE value is below minimum. Increase the SIZE specification for this partition.
24 (X'18')	A permanent change of the partition size is only allowed for static partitions.
28 (X'1C')	The minimum GETVIS area below 16MB cannot be preserved. Decrease the SIZE specification for this partition.

SETLIMIT - SETPFIX Processing

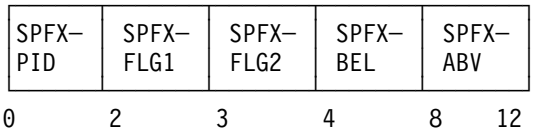
The macro is used by the job control SETPFIX command processing, the end of job processing and the unbatch processing. The format is as follows:

```
[name]   SETLIMIT SPFXPL={name1|(1)}
```

SPFXPL Defines the parameter list into which specified operands (PFIX limits) have to placed before issuing the macro. The address of the parameter list maybe supplied either as an operand or in register 1.

The parameter list has the following format:

SPFXPL



SPFXPID Specifies the 2-byte partition id of the partition for which the PFIX limits are to be set. The limits are only set if SPFXPID specifies the currently active partition or if it is set to X'FFFF'. In that case the partition id of the currently active partition is taken.

SPFXBEL Specifies the number of pages that will be reserved in the PFIX(below) area.
X'FFFFFFFF' means that the PFIX limit is not to be changed.
X'00000000' means that the PFIX limit is to be reset.

SPFXABV Specifies the number of pages that will be reserved in the PFIX(above) area.
X'FFFFFFFF' means that the PFIX limit is not to be changed.
X'00000000' means that the PFIX limit is to be reset.

If the SPFXBEL and/or the SPFXABV value causes an error condition (return code = 12), the following flag bytes are set on return to JCL:

SPFXFLG1 Error during SPFXBEL processing

SFLG1INV EQU X'80' Invalid value specified
SFLG1ALR EQU X'40' ALLOC R already given for that partition
SFLG1EXA EQU X'20' Request would exceed area
SFLG1ANA EQU X'10' There is no PFIX(below) area available.

SPFXFLG2 Error during SPFXABV processing

SFLG2INV EQU X'80' Invalid value specified
SFLG2EXA EQU X'20' Request would exceed area
SFLG2ANA EQU X'10' There is no PFIX(above) area available.

To generate the layout of the parameter list, the SPFXPL macro can be used.

```
[name]              SPFXPL              [DSECT=YES]
```

Output: Register 15 contains one of the following return codes:

Return Code 0	All specified limits set.
Return Code 8	SPFXPID does not specify the active partition.
Return Code 12	One or both PFIX limits cannot be set (See SPFXFLG1 and SPFXFLG2 for a detailed error description).
Return Code 16	The sum of the SPFXBEL and the SPFXABV value is greater than the size of the virtual partition.

Notes:

1. If SPFXPL is not specified, SIZE processing takes place.

SETPRTY

The SETPRTY macro conveys to the supervisor the dispatching and balancing sequence of the partitions as specified by the operator or JCL in the PRTY command.

Only the attention task and the task, where JCL is active, are allowed to issue this macro. Any other program using this macro is canceled due to illegal SVC (ERR21).

The macro has the following format:

```
[name]   SETPRTY  {name1|(1)}
```

The operand has the following meaning:

name1 Address of the area where the information is to be fetched from. The layout of this area is the same as described for the output of GETPRTY TYPE=ALL.

SFREEVIS

The SFREEVIS macro allows supervisor components to free system GETVIS storage (SVA or dynamic space GETVIS).

- Registers 0, 1 and 15 are destroyed.
  All other registers are saved in, and restored from the TCB - SVC work area (SVCSV3), or in the area specified by the SAVE operand.
- The specified area will be cleared always.

Execution mode is AMODE ANY and RMODE ANY.
The format of the macro is as follows:

```
[name]    SFREEVIS  [LENGTH={name1|(0)}]
                    [,SPID={name2|(1)}]
                    [,ADDRESS={name3|(1)}]
                    [,SAVE={name4|(1)}]
                    [,SPACE={YES|NO}]
                    [,PMSPACE={YES|NO}]
```

SAVE — Is the area where the requester wants the general registers to be stored (in case the TCB SVC work area is not available). The save area must be 18 fullwords long, according to the first part of the DSECT SVEARA.

PMSPACE — Specifies that storage is to be freed in the PMR space.

Note: — All other operands have the same meaning as described in the FREEVIS macro, *z/VSE System Macros Reference*, SC33-8230.

SGENL

The SGENL macro provides the ability to generate a local directory list of SDL-like directory entires and is intended to be used by the librarian only.
RMODE 24 is required for the local directory list.

The macro has the following format:

```
[name]   SGENL  name1(,name2(,name3(,... ))
```

name1 — Name of a phase that is to be included in the local directory list.

Up to 200 phase-names may be specified. The phase-names will be alphamerically sorted by the macro expansion.

SGETVIS

The macro SGETVIS allows supervisor components to request System GETVIS storage (SVA or dynamic Space GETVIS).

- Registers 0, 1 and 15 are destroyed.
  All other registers are saved in, and restored from the TCB - SVC work area (SVCSV3), or in the area specified by the SAVE operand.
- It will be forced that a requested area will not cross a page boundary unless the requested area is larger than a page.

Execution mode is AMODE ANY and RMODE ANY.
The format of the SGETVIS macro is as follows:

```
[name]   SGETVIS  [LENGTH={name1|(0)}]
                  [,SPID={name2|(1)}]
                  [,ADDRESS={name3|(1)}]
                  [,SAVE={name4|(r4)}]
                  [,PAGE={YES|NO}]
                  [,PFIX={YES|NO}]
                  [,FTCHPR={YES|NO}]
                  [,EXCREQ={YES|NO}]
                  [,SPCNTRL={YES|NO}]
                  [,SPACE={YES|NO|FTCHPR|PARTKEY}]
                  [,LOC={RES|BELOW|ANY}]
                  [,PMSPACE={YES|NO}]
```

SAVE Is the area where the requester wants the general registers to be stored (in case the TCB SVC work area is not available). The save area must be 18 fullwords long, according to the first part of the DSECT SVEARA.

FTCHPR Specifies whether the area is to be fetch protected.

YES The corresponding GETVIS storage will be fetch-protected. Fetch-protection is a property of the subpool, that means, for all requests for that subpool FTCHPR=YES must be specified.

NO The corresponding GETVIS storage will not be fetch-protected.

EXCREQ Specifies whether the requester may use SVA GETVIS space excessively.

YES The requester identifies itself as a SVA GETVIS user, who may occupy SVA GETVIS space in an excessive manner, triggered by various user functions such as FETCH, LOCK, XECB, etc. Requesters, who specify this parameter, should tolerate a GETVIS return code for this request. The GETVIS function will check in this case, if the current GETVIS request exceeds a predefined high water mark. This high water mark is calculated dependent on the globals AGMAXEX and AGMISGE in IOTAB. The high water mark is evaluated only for the system GETVIS (24-bit) area.

NO The Requester may exceed high water mark (default value).

PMSPACE Specifies whether GETVIS is requested in the PMR space. If YES has been specified, some checking is skipped during GETVIS processing.

Note: All other operands have the same meaning as described in the GETVIS macro, *z/VSE System Macros Reference*, SC33-8230.

SKIPHCF

The macro SKIPHCF provides the ability to skip the number of specified records, or to skip to the end or the begin of of the file, depending on the current direction of reading (initially set by POINTHCF or set by MODHCF). Skipping to the end or to the begin of the file implies an unconditionally READ direction change.

The macro has the following format:

```
[name]   SKIPHCF  {(hcfreg)|(1)},{{count|(0)}|EOF}
```

hcfreg Is the general register containing the address of the HCFCB control block returned by the corresponding POINTHCF macro.

count Is the symbolic name of a 2-byte field containing the number of records to be skipped or the value is given in a register. A negative number is not allowed.

EOF Forces a skip to the begin-of-file if the direction of reading is backward or to the end-of-file if the direction of reading is forward.

Output: Register 15 contains one of the following return codes.

0 (X'00')	Normal processing successfully completed.
4 (X'04')	Inconsistent input.
8 (X'08')	No record found, incorrect length.
12 (X'0C')	Unrecoverable I/O error.
16 (X'10')	HCF device is not ready.
20 (X'14')	Begin of HCF (first record in HCF). Register 0 contains the number of records that the service was unable to skip due to this begin-of-file.
24 (X'18')	End of HCF (last record in HCF). Register 0 contains the number of records that the service was unable to skip due to this end-of-file.

Register Usage: The contents of general register 14 through 2 are destroyed by this macro.

SLEAVE

Release capability currently associated with issuing task and, optionally, return to caller with specified return code.

Must be issued by system components called via the SENTER macro. If the RETURN parameter is specified (see below), SLEAVE is equivalent to a BR 14 with a return code in register 15.

The macro has the following format:

ASSEMBLER:

```
[name]   SLEAVE [RETADD={name1|(r1)|(14)}][,RETCOD={name2|(r2)|(0)}]
```

RETADD Name of a fullword or register containing the address to which control is to be passed after execution of LEAVE. If the parameter is omitted, control is returned to the next sequential instruction.

RETCOD Name of a fullword or register containing the return code to be passed in register 15. If the parameter is omitted, a return code of zero is assumed.

Return Codes: As specified by RETCOD.

Register Usage:

```
R0:   Return code for input.
R14:  Return address for input and output
R15:  Function code for input, return code for output.
```

SLOAD

The SLOAD macro can be used to LOAD a phase into the SVA, or to retrieve SDL-like directory entries.
Execution mode is AMODE ANY and RMODE 24.

The macro has the following format:

```
[name]   SLOAD  phname|(r1)[,{loadadr|(r0)}]
                [,DE={YES|NO|SDLFORM|VSEFORM}]
                [,TXT={YES|NO}]
                [,SYS={YES|NO}]
                [,SVAUPD={NO|YES}]
                [,SDL={YES|NO}]
                [,RET={NO|YES}]
                [,MFG={name1|(S,name1)|(r2)}]
                [,LIST={name2|(r3)}]
```

The operands have the following meaning:

phname Name of the phase that is to be loaded.

loadadr Address where the phase is to be loaded.

DE Directory entry information

This option should be used to determine whether a phase is available in the system and / or to avoid the directory search in case the phase is to be loaded more than once during program execution.

- NO No directory entry is available.
- YES A valid directory entry in the length of 38 bytes and the indication X'0000000D' in offset 8 is provided.
- VSEFORM A VSE directory entry in the length of 40 bytes with the indication X'FFFFFF0E' in offset 8 has been provided. VSEFORM may be abbreviated VSE and it must be used when the System Directory List (SDL) is somehow affected by the SLOAD function.
- SDLFORM A SDL (System Directory List) like directory entry of the length of 68 bytes has been provided. SDLFORM may be abbreviated SDL.

TXT Text processing information.

- NO The phase is not to be loaded. Useful with DE=YES to determine whether the requested phase is available and/or for accessing directory entry information only in case the phase is to be re-loaded during program execution.
- YES The phase is to be loaded. The search sequence is taken as for $-phases.

SYS Search sequence information.

- YES Search sequence as for $-phases.
- NO The normal search sequence which is:
  SDL, temporarily concatenated sublibraries (if any), permanently concatenated sublibraries (if any) and at last SYSLIB sublibrary is taken.

SDL System directory list.

YES Default value.
The currently active library concatenation chain for this partition is to be used.

NO The directory entry is or will be a part of the SDL, the SDL is not searched.

SVAUPD SVA update

NO Default value. The phase is to be loaded into the user partition.

YES The phase is to be loaded into the SVA and the associated SDL entry is to be updated accordingly.

LIST If specified, a pointer to a local list of directory entries is passed. It is recommended to generate this list by the GENL macro for directory entries in VSE format, respectively by the SGENL macro for entries in the SDL format.

RET Return error information

NO Default value
The user does not want return codes to be passed back. The user Program will be canceled in case of permanent errors.

YES The user request return codes to be passed back in register 15. The service caller is not canceled in the case of error situation.

- 0 (X'00') SLOAD completed successfully
- 4 (X'04') Phase not found (preventing cancel code X'22').
  Reasons:
  1. No directory entry was found during directory search.
  2. A directory entry was provided by the user but the corresponding phase is already deleted (the directory search is not restarted).
- 8 (X'08') Unrecoverable I/O error during SLOAD service (preventing cancel code X'2B').
- 12 (X'0C') Invalid library structure detected by SLOAD (preventing cancel code X'29').
- 16 (X'10') Invalid address provided by SLOAD caller (preventing cancel code X'25').
- 20 (X'14') Security violation (preventing cancel code X'0B').
- 24 (X'18') Inconsistent directory entry. SLOAD cannot use the directory entry and confirms an inconsistency between provided directory entry and sublibrary entry. The provided DE is replaced by the sublibrary entry. This return code may by used by programs with own storage management to ensure that the SLOAD service does not overwrite any storage when a phase has been replaced by a longer version in the meantime.
- 28 (X'1C') Phase does not fit in partition OR LTA (Logical Transient Area, preventing cancel code X'28').

32 (X'20') PFIX failure. The phase area could not be PFIX'D (preventing cancel code X'3D').

36 (X'24') RMODE violation. The loadpoint and/or end of phase is above 16MB, but the phase's RMODE is 24 (preventing cancel code X'45').

MFG Macro format information.
Address of a sufficiently large work area (due to the various directory entry formats) where the parameter list is to be generated.

Notes:

1. All registers must be different from each other and register 0 must not be used.
2. If the phase name is specified via register notation, a valid directory entry as specified by DE must be provided.
3. The SVA parameter is still allowed but has no effect.
4. Note that all data areas (phasename, MFG, DE,..) must be located below 16MB.

Note: Refer to *VSE/ESA System Macros Reference*, SC33-6616 for information on 31-bit support.

SPFIX / SPFREE Macros

The macros SPFIX and SPFREE are provided to to PFIX / PFREE SVA 24-bit or 31-bit areas. The macros are similiar to SGETVIS / SFREEVIS

```
[name]  SPFIX     ADDRESS={(R1)|address}
                 [,RLOC={BELOW|ANY}]
                 [,GATE={YES|NO}]
                 [,RETURN={NO|YES}]
                 [,SAVE={(R2)|address}]
```

```
[name]  SPFREE    ADDRESS={(R1)|address}
                 [,GATE={YES|NO}]
                 [,SAVE={(R2)|address}]
```

The operands have the following meaning:

ADDRESS
: specifies the address of the fixlist. If register notation is selected, the specified register contains the address of the related list; otherwise the specified field is regarded as a 4 byte pointer containing the address of the related list.

  The fixlist must be provided in the new format:
  Parmlist := list(Parm)
  Parm := (Begin addr. of area BIN FIXED(31), length(area)-1 BIN FIXED(31))
  End indication of the list is a negative begin address (sign bit = 1).

RLOC=BELOW|ANY
: specifies the location of real storage for PFIX requests, below 16MB (BELOW) or anywhere (ANY). Default is BELOW.

GATE=YES|NO
: specifies whether the gating for SRQSPFIX is to be done by the service (YES) or is provided by the requester itself (NO). Default is YES.

RETURN=NO|YES
: specifies whether the the PFIX service returns to the requester (YES) instead of waiting for free frames (NO). Default is NO.

SAVE
: specifies the area where the requester wants the registers to be saved (in case the TCB SVC work area is not available). The save area must be 18 fullwords long.

Registers 1, 15 are destroyed by the macro expansion.

Return Codes: Register 15 contains one of the following return codes.

0	Successful
4	SPFIX cannot be satisfied: the number of pages to be pfixed exceeds the total number of fixable frames in arae1 + arae3.
8	SPFIX cannot be satisfied at the actual time.
12	One of specified addresses was invalid.
16	A SPFIX request was given with RLOC=BELOW, but at least one page of the requested area is already pfixed in a frame above 16MB by a previous request. The request is ignored, no page is pfixed. A following SPFIX request with the same list and RLOC=ANY does pfix the area.
20	Function code in Register 15 invalid, no page is pfixed.

Note: ALTERNATIVE for SVA Phases:
Instead of doing your own SPFIX for SVA phases, you can use the LOAD service with the PFIX option.

SPMRSERV ID=ALLPMRT|FREPMRT

The supervisor offers the macro SPMRSERV as an interface macro to invoke page manager services by using the macro GETPMT.

1. Registers 1 and 15 are destroyed.
   All other registers are saved in, and restored from the area specified by the SAVE operand. The save area must be 18 fullwords in length according to the first part of the DSECT SVEARA. The registers are stored beginning at the specified address + 16.

Execution mode is AMODE ANY and RMODE ANY.

SPMRSERV ID=ALLPMRT

The service is called by partition and data space allocation processing to allocate page manager tables.

```
[name]   SPMRSERV  ID=ALLPMRT
                  [,SCBPNT={name1|(1)|(Rn)}]
                  ,SAVE={name2|(Rm)}
```

SCBPNT	**SCB pointer**
	SCB of space for which page manager tables are to be allocated. The following fields must be set in the SCB:
SCBFLG	type of space (address/data space)
SCBPASZ	Size (multiple of 4KB) of the maximum possible allocation value within the space. This size must be determined during space creation. For this size, page manager tables are allocated (virtual). Real allocation is done only for the actual allocation request (SCBEXSZ).
SCBEXSZ	Actual allocation request (multiple of 4KB). For this size, page manager page manager tables will be reserved (real).
SCBCUSZ	Size (multiple of 4KB) for which page manager tables are already reserved (real).

Output:

Register 15	**Return Code**
Return code 0	Function completed successfully
Return code 4	No virtual storage available
Return code 8	No real storage available
Updated SCB	
SCBEXSZ	0
SCBCUSZ	SCBCUSZ = SCBCUSZ(input) + SCBEXSZ(input)
SCBVSTO	virtual address of segment table
....	

Internal Processing: The service requests allocation of page manager tables within the existing page manager address spaces by using the macro GETPMT. In case there is not enough virtual storage available, a new page manager space is created.

SPMRSERV ID=FREPMRT

The service is called by address/data space deallocation processing to free page manager tables.

```
[name]   SPMRSERV  ID=FREPMRT
                  [,SCBPNT={name1|(1)|(Rn)}]
                  ,SAVE={name2|(Rm)}
```

SCBPNT **SCB pointer**

SCB of the space for which page manager tables are to be freed.

Output:

Register 15	**Return Code**
Return code 0	Function completed successfully
Updated SCB	
SCBCUSZ	SCBCUSZ = 0
SCBVSTO	SCBVSTO = 0
....	

SPMRSERV ID=RELDS

1. Register 1 is destroyed.
   All other registers are saved in, and restored from the area TCB work area SVCSV3. The save operand must not be specified.

Execution mode is AMODE ANY and RMODE ANY.

SPMRSERV ID=RELDS

The service is called by the DSPSERV RELEASE processing to release pages.

```
[name]   SPMRSERV  ID=RELDS
                  [,SCBPNT={name1|(1)|(Rn)}]
```

SCBPNT **SCB pointer**

SCB of space for which page manager tables are to be allocated.

SRCHFLD

Return the physical unit number of a device which is identified by its cuu-address or by its device type code.

The physical unit number of the matching device is returned right adjusted in register 1.

The macro has the following format:

```
ASSEMBLER :
 [name]    SRCHFLD FIELD={CHNUNIT|DEVTYP},VALUE={name1|(r1)|(1)}
                   [,PU={name2|(r2)|(0)}]
```

FIELD Symbolic identification of the field to be searched for.

CHNUNIT 2-byte channel and unit address in the form cuu.

DEVTYPE 1-byte device type code.

VALUE Name of a 4-byte field or a register containing the right adjusted value of the field to be searched for.

Register 0 may not be used for register notation.

PU Name of a 2-byte field or register containing the physical unit number (same as PUB index = PUB offset/8) of the device at which searching has to start. Search is in ascending order and stops at the first match. If this parameter is omitted or zero, search starts with the lowest physical unit number.

Output:

Register 15: contains one of the following return codes.

0 (X'00')	An appropriate PUB index was returned in register 1.
4 (X'04')	No matching PUB found.

Register 1: Physical unit number

Register Usage:

```
R0:    Physical Unit number for input.
R1:    Search argument for input
R15:   Function code for input
```

STARTP

The STARTP macro is used to call the phase $IJBSTRT which starts the partition requested in the macro.
The service is available for static partitions only.

The macro has the following format:

```
[name]   STARTP PART={number|(r1)}
                SAVE={address|(r2)}
```

The operands have the following meaning:

PART This keyword operand specifies a partition number (0 for BG, 1 for F1, etc.) If number is specified, the operand contains a decimal number up to 15. If (r1) is specified, the item within the brackets specifies a register that contains a binary value.

SAVE If address is specified, the operand contains the assembler label of an 18-fullword save area. If (r2) is specified, the item within the brackets specifies a register that contains the address of the save area at execution time.

Output: Register 15 contains one of the following return codes:

0 (X'00')	Successful execution
4 (X'04')	Area not available
6 (X'06')	No ASI bit on for partition to be started
8 (X'08')	No GETVIS in issuing partition

STXIT

STXIT Macro - Define DIE appendage

Disabled Timer Appendage (DIE)

The STXIT macro is used to define a DIE appendage. The issuer of the STXIT macro must be in supervisor state and must run with PSW key zero.

```
[name]  STXIT DIE,rtnaddr|(1)
```

Input to the STXIT macro is the DIE appendage address specified with rtnaddr. Register notation may be used. In this case register 1 is the default register. The specified DIE appendage must reside in the SVA und must be PFIXed. Bit 0 of the input register defines the addressing mode of the DIE appendage.

Output from STXIT

- Register 0 contains a TOKEN, designed for input to the delete function of the STXIT macro and to the Set-TOD routine.
- Register 1 contains the entry address and addressing mode of the Set-TOD routine.
- Register 15 contains a return code.

Delete DIE appendage - STXIT Macro:

The STXIT macro is used to delete a DIE appendage. The issuer of the STXIT macro must be in supervisor state and must run with PSW key zero.

```
[name]  STXIT DIE,rtnaddr|(1),TOKEN=(0)
```

Input to the STXIT macro

- rtnaddr must be zero.
- Only register notation is supported for the TOKEN parameter. The specified register must be loaded with the TOKEN returned with the define function of the STXIT macro.

On output register 15 contains a return code.

Define Set-TOD routine.: The Set-TOD routine must be called by means of BASSM 14,12. The caller must be in supervisor state, PSW key zero, disabled for external and I/O interrupts and must own the NP lock. The routine is designed to be called from a normal user task (normally first time) and from the DIE appendage (normally all other requests).

Input to the Set-TOD routine

- Register 0 must contain the TOKEN returned by the define function of the STXIT macro.
- Registers 1 and 2 must contain the TOD value of the point of time where the DIE appendage expects to get control.
- Register 12 must contain the base address of the Set-TOD routine returned by the define function of the STXIT macro.
- Register 13 must contain a pointer to a savearea of 16 words.
- Register 14 must contain the return address

On output register 15 will contain a return code.

DIE - appendage routine:

The DIE appendage routine is called by the FLIH for external interrupts by means of BASSM 14,15 when the point of time specified with the Set-TOD routine is reached. The routine is entered in supervisor state, with PSW key zero and disabled for I/O and external interrupts and in the addressing mode specified with the STXIT macro. No page fault is allowed while the DIE appendage is active otherwise the sytem enters hardwait FFF. The DIE appendage must not modify any register. It must not use any SVC except fast path SVCs (SVC 107) of class A e.g. TREADY IO macro. It can call the Set-TOD routine to set a new TOD value where it expects control next time. The shared space is set up when the DIE appendage gets control.

SUBSID

The SUBSID macro must be used to keep the supervisor informed about the state (active, inactive) of a specific subsystem thus allowing other subsystems to inquire the state of another subsystem or the currently loaded supervisor itself.

The following table lists the SUBSID names defined for the subsystems which are accepted by the supervisor. The four byte SUBSID name is a unique name and must always be used for the SUBSID services.

SUBSID Parm	Subsystem – name	Subs. attribute Allowed Once in	Parameter passed 'SPARM'	
VSPT	Performance tool	Task	No	–
PSF	PSF/VSE	Partition	No	–
DSNX	DSNX	System	No	–
ICK	ICKDSF	Partition	No	–
ARI	SQL/DS	Partition	No	–
CIC3	CICS 3.X	Partition	Yes	Ptr to IJBAFCB
CIC4	CICS/TS	Partition	No	–
CUST	VSE Cust.	System	No	–
DLI	DL/1	Task	No	–
FTP	FTP	Task	No	–
ICCF	ICCF	System	No	–
ISPF	SPF	Task	No	–
NETV	NetView	Partition	No	–
NPDA	NPDA	System	No	–
OCCF	OCCF	System	Yes	Ptr to OCCF COMREG in IJBOCFCM
PWR	VSE/POWER	System	No	–
SOEM	SOEMI	Task	No	–
SSX	SSX	Partition	No	–
SUP	Supervisor	System	No	–
VCNA	VM/VCNA	System	No	–
VTAM	VTAM	System	No	–
DMF	DMF	System	No	–
SECS	Sec. Server	System	No	–

The macro has the following format:

```
[name]   SUBSID  {NOTIFY[,UPDATE]|
                  REMOVE|INQUIRY}
                 ,NAME={name1|(r1)|(S,name1)}
                 [,SPARM={name2|(r2)|(S,name2)}]
                 [,AREA={name3|(r3)|(S,name3)}
                 ,LEN={n|name4|(r4)|(S,name4)}
                 [,PID={name5|(r5)|(S,name5)}]
                 [,MFG={name6|(r6)}]
                 [,LVLTEST={NO|YES}]]
```

NOTIFY This option allows a VSE/AF task to be considered as a subsystem by the VSE/AF supervisor and to request certain privileged supervisor functions. A subsystem may be active more than once in one VSE/AF system environment according to the subsystem attribute.

UPDATE This option allows a VSE/AF task to update its current subsystem information. This option is only valid for active subsystems which have done a NOTIFY before.

REMOVE This option will change the requesting VSE/AF task to be no longer considered as a subsystem by the VSE/AF supervisor.

INQUIRY This option can be used to determine the state and probably the level of a defined subsystem or the supervisor itself.

The layout of the supervisor entry is described by the mapping DSECT generated by the macro MAPSSID.

The format is as follows:

```
[name]   MAPSSID
```

DEC	HEX	Label	Description
0 – 1	0 – 1	IJBSSID1	Partition id
2 – 5	2 – 5	IJBSNAME	Program name
6	6	IJBSVERS	Version number
7	7	IJBSREL	Release number
8	8	IJBSMOD	Modification number
9	9	IJBSVARL	Length of variable part
10	A	IJBSFLAG	Flags (varying length)
		IJBSFL01	Flag byte 1
		IJBSF370	X'80' 370 support (always on)
		IJBSFEXX	X'40' always off (former e-support)
		IJBSFCKD	X'20' CKD support
		IJBSFFBA	X'10' FBA support
		IJBSFAPR	X'08' 3800 support
		IJBRCHAN	X'04' Relocating channels
		IJBSVMLE	X'02' always off (former vmle support)
		IJBSVMAC	X'01' Running on VM system
11	B	IJBSFL02	Flag byte 2
		IJBSFAF	X'80' AF support
		IJBSFPAG	X'40' 4K page size used
		IJBSUNAT	X'20' Unattended environment
		IJBSESAS	X'10' ESA supervisor (always on)
		IJBSACCR	X'08' Access registers available (always on)
			X'04' Reserved
			X'02' Reserved
			X'01' Reserved

Figure 24 (Part 1 of 2). Output as Described by Macro MAPSSID

DEC	HEX	Label	Description
12	C	IJBSFL03	Flag byte 3
		IJBSFSEC	X'80' Security support
		IJBSFSHR	X'40' DASD sharing support
		IJBSFSAT	X'20' JIB replaced by SAT
			X'10' Reserved
			X'08' Reserved
			X'04' Reserved
			X'02' Reserved
			X'01' Reserved
13	D	IJBSFL04	Flag byte 3
			X'80' Reserved
			X'40' Reserved
			X'20' Reserved
			X'10' Reserved
			X'08' Reserved
			X'04' Reserved
			X'02' Reserved
			X'01' Reserved
14 – 15	E – F	IJBSLCON	Library concatenation chain length
10	A	IJBSFIXL	Length of fixed part
16	10	IJBSSLEN	Total length of DSECT

Figure 24 (Part 2 of 2). Output as Described by Macro MAPSSID

The number for Version, Release and Modification level of the VSE supervisor will always increase with new releases. For VSE/ESA 2.1.0 the numbers will be x'060100' (IJBSVERS / IJBSREL / IJBSMOD).

NAME Is the address of a field describing the subsystem.

The field contains information such as:

DEC	HEX	Description
0 – 3	0 – 3	Name field containing the unique subsystem name (this field is the only one required for REMOVE or INQUIRY)
4 – 6	4 – 6	Subsystem specific information (applies to NOTIFY only)
7	7	Containing the length (0–24 bytes) of the optional variable part appended to the required fixed part (byte 0 – 7 —— applies to NOTIFY only)
8 – 31	8 – 1F	Maximum of 24 bytes containing subsystem parameters. This field may contain such information as version and release number as well as features supported by this subsystem. (applies to NOTIFY only)

Figure 25. Subsystem Descriptor

SPARM Is the address of a field containing information that is to be saved in a predefined field within the supervisor. This operand is accepted from special subsystems only (see table above)

AREA Is the address of an area where the user want the information describing the subsystem to be stored.

The returned information does contain the PIK (byte 0-1) followed by at least 8 bytes (byte 2-9) containing the subsystem specific information as passed with the NOTIFY option. (Macro MAPSSID describes the layout of the supervisor entry).

LEN Specifies the area length either as an integer, a self-defining term, or as value in a register, or, in S-type notation as the name of a halfword containing the value.

The value must be in the range from 10 to 34 bytes.

PID Is the address of a halfword containing the PIK of the partition which is to be interrogated whether the specified subsystem is active. In case this operand is omitted, or if the PIK is zero, the whole internal subsystem table is scanned until the first matching entry is found which will then be returned to the requester.

MFG Specifies the address of a work area where to build the parameter list.

LVLTEST YES: Generates code which ensures that the IPLed supervisor does support the SUBSID.

Output: Register 15 contains one of the following return codes.

NOTIFY

0 (X'00')	Information stored
4 (X'04')	Subsystem already active in system
8 (X'08')	Byte string too long, SUBSID rejected
12 (X'0C')	Subsystem table is full
16 (X'10')	Subsystem not active in system

UPDATE

0 (X'00')	Information stored
4 (X'04')	Not used for UPDATE
8 (X'08')	Byte string too long, SUBSID rejected
12 (X'0C')	Not used for UPDATE
16 (X'10')	Subsystem not active in system

REMOVE

0 (X'00')	Information for the specified subsystem removed
4 (X'04')	No matching subsystem entry found

INQUIRY

0 (X'00')	Information returned
4 (X'04')	Information returned, however, the same subsystem is currently active in another partition. Register 0 contains the PIK of that partition in its high-order two bytes.
8 (X'08')	Returned information truncated, the area is too short. Register 0 contains the total length in the low-order two bytes.
12 (X'0C')	Return codes 4 and 8 together.
16 (X'10')	Name not found in subsystem table.
20 (X'14')	Supervisor does not support SUBSID service (LVLTEST=YES only)

Register Usage:

```
R0   contains the function code and return information.
R1   pointer to the parameter list, byte string or name.
R15  pointer to the special parameter and return code register.
```

SUPRET

The SUPRET macro is used by all A-transient routines (error recovery and recording transients) to properly return to the supervisor.

The macro has the following format:

```
[name]   SUPRET  [ENTRY=][,PLSBASE=]
```

The operands have the following meaning:

ENTRY Identifies to the supervisor the type of transient that wants to return control to the supervisor (error recovery or recording transient). If nothing is specified, an error recovery transient is assumed. A recording transient must specify ENTRY=AE .

PLSBASE Indicates for PL/S coding the register containing the address of the error block, if the PL/S version of the ERBLOC macro has been used. The register must be enclosed by parentheses.

Output: In-line code for return to supervisor is generated.

SVALLIST

This macro produces the assembler source code of a load-list. The load-list contains the names of the phases that are to be loaded automatically into the SVA during IPL under control of one or more load conditions.

The macro has the following format:

```
[name]   SVALLIST  [llname,]
                     (phname1[,cond1][,cond2]...[,condn])
                   {,(phname2[,cond1][,cond2]...[,condn])
                   {,(phnamen[,cond1][,cond2]...[,condn])}}
```

The operands have the following meaning:

llname Specifies the phase name of the LOADLIST to be generated. This name has to be predefined in the master LOADLIST $$A$SVA. Llname is mandatory in the first SVALLIST macro call within one assembly; subsequent SVALLIST macro calls ignore the llname specification and assume to be a continuation of the first macro call or load-list, respectively.

phname Specifies the names of the phases that are to be included into LOADLIST specified by llname.

cond Specifies the condition that the IPL'ed supervisor must meet in order to get the corresponding phase automatically loaded into the SVA.

FBA Supervisor must provide FBA support.
RPS Supervisor must support rotational position sensing
SEC Security was activated at IPL time.

If none of the load conditions is specified the corresponding phase is loaded unconditionally.

SYSDEF

The macro has the following format:

```
[name]   SYSDEF     ID={DSECT|QUERY|UPDATE}
                   [,AREA={name1|(r1)}]
```

The operands have the following meaning:

name Specifies the label, which is used. This label defines the name of the generated DSECT in case of SYSDEF ID=DSECT.

ID Specifies the requested service. Possible values are: DSECT, QUERY and UPDATE

AREA This parameter specifies the address of the input parameter list. As part of the parameter list, a pointer to a work area must be specified. Both the input parameter list and the work area must be PFIXed.

This macro can be used to invoke one of the following services:

SYSDEF ID=DSECT The generated DSECT (DSPSYSDF by default) is a mapping of the user-supplied area. This area consists of a header to describe the requested function with its parameters and a data area for the input and output fields of the requested service. For a detailed description of the various input and output fields see SYSDEF ID=DSECT macro expansion.

SYSDEF ID=UPDATE This service offers a means to change the data space system defaults/limits. The user of this service has to indicate in the generated DSECT (SYSDEF ID=DSECT) which defaults/limits are to be updated. The macro updates the DSECT specified in the area parameter and expands to the SVC.

SYSDEF ID=QUERY This service returns information about the data space system defaults/limits or the actual data space usage situation of the system. The user of this service is responsible for proper indications in the DSECT (SYSDEF ID=DSECT). The macro will update the DSECT specified in the area parameter and expand to the SVC.

Usage restrictions: The usage of this service is restricted to the attention routine, to Job Control and to authorized vendors. For information about the vendor interface, refer to the corresponding chapter in this book. The service is only available on an ESA Supervisor. Callers may run in either 24- or 31-bit addressing mode. The caller may reside above the 16MB boundary. However, the ASC mode must be Primary Mode.

Cancel Conditions: The SYSDEF macro is canceled, if the address of the passed parameter list as specified by the AREA parameter or the address of the GETVIS area as pointed to by the parameter list are not part of the caller's storage (that means, either partition space or SVA).

Register Usage:

Reg. 1 Address of Input Control Block (refer to SYSDEF ID=DSECT)

Reg.15 Return code

Output: Register 15 contains one of the following return codes:

0 (X'00')	The function completed successfully.
4 (X'04')	Work area does not suffice. Either the work area is smaller than required minimum or area is not able to contain all records.
8 (X'08')	Data information in user supplied area is incorrect.
12 (X'0C')	Requester is not authorized. Requester is not Attention Routine, JCL or a authorized vendor.
20 (X'14')	Address of input parameter list of work area is invalid.
24 (X'18')	Header information in user supplied area is incorrect.

SYSIO

The SYSIO macro requests the initiation of an I/O operation ahead of all other I/O operations requested by EXCP(SVC 0). It will observe the priority for headqueuing assigned to the different system tasks. The WAIT for the completion of the I/O request is implied when using SYSIO.

The macro has the following format:

```
[name]  SYSIO  {name1|(1)}
```

The operands have the following meaning:

name1 Is the address of a CCB or IORB established for the device. It can be given as a symbol or in register notation.

Output: The traffic bit in the CCB or IORB is posted when the system task gets back control. If an error occurred the disastrous error indicator in the CCB is posted and must be checked by the system task. Transient error recovery procedures are skipped for headqueue requests and headqueue errors will not be recorded onto the record file. A normal user task issuing SYSIO or SVC 15 is canceled due to illegal SVC, cancel code 21.

TRANSCSW

The TRANSCSW macro returns the address of the original users CCW that corresponds to the address of a given, copied CCW - see z/VSE Supervisor Diagnosis Reference, Chapter "Channel Program Translation". It is intended to be used by the ERP routines only.

The macro has the following format:

```
[name]  TRANSCSW  {ccwaddr|(0)},{ccbaddr|(1)}
```

The operands have the following meaning:

ccwaddr Contains the copied CCW address.

ccbaddr Contains the address of the copied CCB. The address may be passed in any register except R0.

Output: Register 0 points to the address in the user's program.

TREADY

The TREADY macro must be used to set a specified task "ready-to-run" which includes the ability to abnormally terminate the task(s).

The macro has the following format:

```
[name]   TREADY COND={LQ|VCANCEL|OCANCEL|ATTINT|
                ALLOCR}|

                PART={name1|(r1)|(0)}
                [,COND={START|OC|IO
                       {CANCEL,CODE={name2|(r2)|(1)}}]|

                TASK={name3|(r3)|(0)}
                [,COND={IO|NO|ICCF|
                       VTAM[,RETURN={YES|NO}]|
                       {CANCEL,CODE={name4|(r4)|(1)}}|
                       {POWER,
                       PU={name5|(r5)|(1)}}]
                       [RETURN={YES|NO}]
                       }}]
```

The operands have the following meaning:

COND= Specifies the condition that one or more tasks or even a partition must meet in order to be set "ready-to-run".

LQ (LOG task only)
All tasks waiting for the LOG task are to be set ready.

VCANCEL (VTAM only)
Indicates that all tasks, which are communicating with VTAM (at least one VTAM ACB is open) are to be canceled with cancel code X'40'.

OCANCEL (OCCF only)

Will cancel the OCCF main task. Cancel requests are only allowed from OCCF subtasks. Cancel code is X'3C'.

ATTINT (OCCF only)
Indicates that an attention interrupt has to be simulated. Register 0 must be set by the caller and indicates to process to be performed.

R0 1 Indicates command available
2 Request cancel
R1 Must contain the address of a field containing the length of the command in bytes 0-1 immediately followed by the command itself.

ALLOCR Unlock accessed fields related to the system 'real partition' and posts all tasks waiting for accessing the segment table.

PART= Name of a 2-byte field or register containing the identifier (PIK) of the partition to be started or canceled. The PIK is available in field PID of the corresponding COMREG, or in the same field of the BG COMREG, if the partition is active.

This operand is required for COND=START and COND=OC.
It is also required for COND=CANCEL in case TASK is omitted.

START (JOB CONTROL and VSE/POWER only)
Valid with PART only.
Indicates that the partition is to be removed from the unbatched or stopped state. This is the default option if PART was specified. The main task of the partition is scheduled for EOJ with cancel code X'10' and the number of active virtual partitions (field IJBAPNO in SYSCOM) is incremented. The user must ensure that the partition area has been allocated.

OC (System internal use only)
Valid with PART only.
Indicates that the operator communication exit for the specified partition has to be activated, if available.

IO (key zero)
Indicates that all subtasks of the specified PART are made ready only if they are waiting as a result of a WAIT or WAITM macro.

Output:

0 (X'00')	Activation successful
4 (X'04')	OC exit routine already active
8 (X'08')	No OC exit routine available

CANCEL (VSE/POWER, VTAM, ICCF only)
Valid with PART or TASK only.
Indicates that the maintask of the specified partition or the specified task is to be canceled with the cancel code specified in the CODE operand.

CODE= The CODE operand refers to the name of a 1-byte field or to a register containing the cancel code. Register 0 must not be used.

TASK= Name of a 2-byte field or register containing the TID of the task to be posted or canceled. The TID is available in the SYSCOM field TID when the task is active.
This operand is required for COND=(NO, IO, VTAM, ICCF and POWER).
It is also required for COND=CANCEL in case PART is omitted.

IO (Key zero)
Valid with TASK only.
Indicates that the task is to be made ready only if it is waiting as a result of a WAIT or WAITM macro.

COND=IO is the default value if TASK is specified

NO (IPL, LOG task only)
Valid with TASK only.
Indicates that the task is to be made ready unconditionally.

VTAM (VTAM only)
Valid with TASK only.
Indicates that the task is to be made ready as for COND=IO and, in addition, that the VTAM AP exit has to be taken the next time the task is dispatched. If the specified task does not exist the caller would normally be canceled. This is not done in case RETURN=YES was specified. Instead a return code of x'80000010' is set in register 15 on return.

ICCF (ICCF only)
Valid with TASK only.
Indicates that the task is to be put into the ready state from the ICCF wait state.

CANCEL (VSE/POWER, VTAM, ICCF only)
Valid with PART or TASK only.
Indicates that the maintask of the specified partition or the specified task is to be canceled with the cancel code specified in the CODE operand.

CODE= The CODE operand refers to the name of a 1-byte field or to a register containing the cancel code. Register 0 must not be used.

POWER (VSE/POWER only)
Valid with TASK only.
Indicates that an I/O request spooled by VSE/POWER is completed for the specified task and that the task has to be posted if it is waiting on a WAIT or WAITM macro. In addition, all tasks waiting for VSE/POWER service within the same partition are posted. If the specified task does not exist the caller would normally be cancelled. This is not done in case RETURN=YES was specified. Instead a return code of x'80000010' is set in register 15 on return.

PU= Name of 2-byte field or register containing the physical unit number (PUB index) of the device for which posting is requested.

Register Usage:

R0 PIK or TID for input.
R1 Cancel code for input, whenever applicable.
R15 Function code for input, return code for output whenever applicable.

TSTOP

Deactivate the current task or partition.

The macro has the following format:

```
[name]    TSTOP [COND={SYSBND|STOP|UNBATCH}]
                [,RETURN={NO|YES}]
```

The operands have the following meaning:

COND Specifies the condition into which the issuing task is to be set.

SYSBND (System only)
This is the default value and indicates that the issuing task is to be set into the "system-bound" condition. It is to be used by non-resident system tasks to deactivate themselves.

STOP (JOB CONTROL only)
Indicates that processing has to be stopped in the current partition. The status is saved at the invocation point, and processing will resume at the next sequential instruction, as soon as the partition is started again (TREADY COND=START macro, see above). The main task of the partition is made undispatchable and the number of active virtual partitions (field IJBAPNO in SYSCOM) is decremented.

UNBATCH (JOB CONTROL only)
Indicates that processing has to be stopped in the current partition and, in addition, that the partition has to be invalidated. The status at the invocation point is not saved in this case. The partition has to be reinitialized before it is started again (TREADY COND=START macro, see above). The main task of the partition is made undispatchable and the counter of active virtual partitions (field IJBAPNO in SYSCOM) is decremented.

RETURN (Valid with COND=SYSBND only)

NO This is the default option and indicates that the status as present at the time this service was invoked is to be saved and that processing is to be resumed at the next sequential instruction, as soon as this task is activated again.

YES Control returns immediately to the calling program without status saving.

Register Usage:

R15 Function code for input.

VALID

The VALID macro can be used to check if addresses of user specified storage area is contained within the user's addressing limits.
For each page of the area specified by BEGIN and END (see below) it is checked whether the service owner is allowed to access it in the requested way. An appropriate return code is returned in register 15.

The macro has the following format:

```
[name]   VALID     BEGIN={addr|(1)}
                   ,END={addr|(2)}
                   ,CHECK={READ|UPD}]
```

The operands have the following meaning:

BEGIN Specifies the begin address of the area to be handled.

END Specifies the end address of the area to be handled.

CHECK Specifies the type of check to be done.

READ User wants to check if read access within the specified area is possible. This assumes that,

- None of the pages within the specified area is fetch protected.
- None of the pages within the specified area is flagged invalid and any of the pages has a storage protection key that is valid to be accessed by the issuing task.

UPDate User wants to check if write access within the specified area is possible. This assumes that,

- None of the pages within the specified area is flagged invalid and any of the pages has a storage protection key that is valid to be accessed by the issuing task.

Output: Register 15 contains one of the following return codes.

0 (X'00')	Requested access is allowed to the total area
4 (X'04')	Reserved
8 (X'08')	CHECK=READ: storage is fetch protected or key mismatch CHECK=UPD: key mismatch
12 (X'0C')	Addressed area is invalid

VIO

For a description of the VIO macro refer to “SVC 114 (X'72' - VIO)” on page 292.

VIO CLOSE

This macro has the following format:

```
Assembler :
 [name]   VIO       CLOSE
                   [,{VIORB={(r1)|(1)}|
                    SCOPE={STEP|JOB}}]
```

The operands have the following meaning:

VIORB Register containing the VIORB pointer (as returned by VIO OPEN) of the work area to be deallocated. If both VIORB and SCOPE are omitted, Reg.1 is assumed to contain the VIORB pointer.

SCOPE Unconditional deallocation of all VIO work areas belonging to the issuing partition with the specified lifetime (see also VIO OPEN). This operand is reserved for system usage. The specification is passed in Reg.0.

Return Codes in Register 15:

0 (X'00') Successful deallocation.

VIO EXTND

Assembler :

```
[name]   VIO       EXTND
                   ,SIZE={nK|(r1)|(0)}
                   [,VIORB={(r2)|(1)}]
```

The operands have the following meaning:

SIZE Amount of additional space to be allocated.
The unit is KB for absolute notation and bytes for register notation. The specified value is passed in Reg.0.
The requested increment is interpreted relative to the actually allocated size available in the VIORB. The additional space is logically contiguous to the existing area.

VIORB Register containing the VIORB pointer (as returned by OPEN) to the work area to be extended. If the parameter is omitted, Reg.1 is assumed to contain the VIORB pointer.

Return Codes in Register 15:

0 (X'00')	Additional space allocated
8 (X'08')	No more space available

VIO MOVE

Assembler :

```
[name]   VIO      MOVE
                  ,FROM={name1|(r1)|(r11,r12)}
                  ,TO={name2|(r2)|(r21,r22)}
                  ,LEN={n3|(r3)}
                  [,MFG={name4|(r4)|(1)}]
```

The operands have the following meaning:

FROM — Address of source data area.
If symbolic or single register notation is used, an address in virtual storage is assumed. Double register notation must be used to specify a VIO address. In this case, r11 is interpreted as the VIORB pointer of a VIO area and r12 as an offset (RBA) within the VIO area.

TO — Address of target area.
The notation convention is the same as for the FROM operand.

LEN — Number of contiguous bytes to be moved.
The maximum specification for absolute notation is 4095. With register notation, any number compatible with the size of program/VIO areas may be specified. Crossing of VIO block boundaries is supported.

MFG — Address of a 20-byte area, in which the parameter list is to be build. If omitted, an in-line parameter list area is generated.

Register Usage:

R0	Not used.
R1	Address of parameter list.
R2-R12	May be used for register notation.
R13	Assumed to contain the address of a 72-byte save area.
R14	Link register.
R15	Address of MOVE routine (from the VIORB) for input. Return code for output.

Return Codes in Register 15:

0 (X'00')	Operation successful
4 (X'04')	End of File reached on VIO file
8 (X'08')	Unrecoverable error
12 (X'0C')	Inconsistent state on VIO access
16 (X'10')	Invalid input

Restrictions: At least one of the FROM/TO parameters must designate a VIO address. If both are VIO addresses, they may not refer to the same VIO area (r11 ¬= r21). Control returns to the caller only after completion of the MOVE operation, independently of VIO PROC options.

VIO OPEN

Assembler :

```
[name]   VIO       OPEN
                   ,SIZE={nK|(r1)|(0)}
                   [,SCOPE={STEP|JOB}]
                   [,PROC={SYNCH|ASYNCH}]
```

The operands have the following meaning:

SIZE Amount of space to be allocated.
The unit is KB for absolute notation and bytes for register notation. The specified value is passed in Reg.0.

SCOPE Lifetime of the work area.

STEP The work area is automatically deallocated at end of job step (default).

JOB The work area is automatically deallocated at end of job.

PROC Processing mode.

SYNCH After VIO POINT (see below), the issuing task is implicitly set to wait until the block is available (default).

ASYNCH After VIO POINT, control returns immediately to the issuing task and WAIT or WAITM must be issued before accessing the requested block.

Output: An address is returned in reg.1, which points to a system control block (VIORB). The VIORB address uniquely identifies the allocated work area and must be specified for all subsequent requests referring to this area. The VIORB contains the actual size of the allocated area, which can be larger then the requested size, depending on the internal allocation unit. The VIORB also contains the size of a VIO block, which is identical with the page size. The user is recommended to treat the block size as a variable, to be obtained from the VIORB after VIO OPEN. Note that a POINT request is necessary before a block of a VIO area becomes addressable to the program.

Return Codes in Register 15:

0 (X'00')	Successful allocation
8 (X'08')	No more space available

VIO POINT

Assembler :

```
[name]   VIO       POINT
                  ,RBA={(r1)|(0)}
                  [,{VIORB={(r2)|(1)}
```

The operands have the following meaning:

RBA Register containing a relative byte address within the VIO area, to which addressability is requested. The specified value is passed in Reg.0.

VIORB Register containing the VIORB pointer (as returned by OPEN). of the work area to be accessed. If the parameter is omitted, Reg.1 is assumed to contain the VIORB pointer.

Output: The results of a VIO POINT are returned to the requester in the VIORB. The layout of the VIORB is described by the MAPVIORB macro, see below.

Bit 0 in byte 2 of the VIORB indicates completion of a POINT request (same as the traffic bit in a CCB). If PROC=SYNCH was specified for OPEN, the request is always complete when the task regains control after a POINT request. If PROC=ASYNCH was specified, the traffic bit must be checked (usually by WAIT or WAITM) before processing based on a previous POINT request can continue. Note that the traffic bit is exclusively maintained by the system. Since the VIORB is allocated in protected storage, the user program can only retrieve information from it.

Error conditions are indicated by a return code in field VIORBRTC (see MAPVIORB).

After a successfully completed POINT request (no error flag on), a VIO block containing the requested RBA is addressable under the virtual address returned in field VIORBPNT. The RBA of the first byte of this block is returned in field VIORBRBA. The block remains addressable to the user program until a POINT request for another RBA is issued or the work area is deallocated by VIO CLOSE see below. Any later attempt to access the block leads to unpredictable results.

When a block is accessed the first time after VIO OPEN or EXTND it is cleared to binary 0's.

Performance Note:
If PROC=ASYNCH was specified for OPEN, POINT is always executed on a fast path without redispatching. For PROC=SYNCH, the fast path is taken whenever the requested block is already available in real storage.

Path length figures are not yet available.

MAPVIORB Macro

MAPVIORB generates a DSECT describing the VIORB.

This macro has the following format:

```
Assembler :
 [name]   MAPVIORB
```

Bytes Dec	Bytes Hex	Labels	Description
0 – 1	0 – 1		Reserved
2	2	VIORBCM1	Communication byte
		VIORBTRB	X'80' Point request complete
3	3	VIORBRTC	Return code
		VIORBEOF	X'04' Requested block outside area
		VIORBERR	X'08' Unrecoverable error
		VIORBINC	X'0C' Inconsistent state
4 – 7	4 – 7	VIORBASZ	Actual size of area in bytes
8 – 11	8 – B	VIORBBSZ	Size of a block in bytes
12 – 15	C – F	VIORBPNT	Virtual address of current block
16 – 19	10 – 13	VIORBRBA	Relative byte address of current block

Figure 26. DSECT Generated by Macro MAPVIORB

VSIUCVU, VSIUCVPL, VSIUCV

Macros for VM/VCNA (VTAM Communication Network Application) Support

There are control blocks by which an application program passes requests to the subsystem support for VM/VCNA (also referred to as VSE/Advanced Functions IUCV). Macros are provided to build these control blocks and to map them for symbolic reference within a program. The blocks are defined by the macros VSIUCVU and VSIUCVPL.

A single macro, VSIUCV, is used to request subsystem support functions provided by VSE/Advanced Functions.

VSIUCV

The VSIUCV macro is used to request subsystem support functions provided by VSE/Advanced Functions.

The macro has the following format:

```
[name]   VSIUCV CB={addr|(r1)}
               ,OP={OPEN|CLOS|CONN|ACPT|SEVR|SSTE}
              [,VSIUCVU={addr|(r2)}]
              [,{TRGTID=(id,name)|PATH=number|(r3)}]
              [,UDATA={data|(r4)}]
              [,PRTY={YES|NO}]
              [,QUIES={YES|NO}]
              [,MSGLIM={value|(r5)}]
              [,ID=name]
              [,EXIT={addr|(reg)}]
```

The operands have the following meaning:

CB — Is the address of the function related control block. It is the address of either the VSIUCVU control block, (OP=OPEN, OP=CLOS or OP=SSTE), or it is the address of the VSIUCVPL control block (OP=CONN, OP=ACPT or OP=SEVR).

OP — Specifies the requested service.

OPEN — Identifies a program to VSE/Advanced Functions IUCV and establishes the environment necessary to connect one program to another program.

CLOS — Indicates that an application program wants to drop the connection with the VSE/Advanced Functions IUCV. It immediately severs all paths associated with this user ID.

CONN — Initiates a connection between the issuing user and another user of IUCV.

ACPT — Indicates that a program accepts a connection request initiated by another user.

SEVR — Terminates a previously established connection request or it forces an incoming connection request to be rejected.

SSTE — Issuer requests supervisor state. This enables the requester to issue macros for VM/VCNA (VTAM communication Network Application) support.

VSIUCVU — The address of the VSIUCVU block that identifies the user.

trgtid — 'id' is the VMID of the virtual machine of the target user.
'name' is the identifier by which the target user is known.
(TRGTID and PATH are mutually exclusive).

PATH — Is the path ID or the path number.
It is the path ID passed when the user is notified of the incoming connection request or it is the path ID of the path being terminated or rejected.
(PATH and TRGTID are mutually exclusive).

UDATA Any 4 bytes of user information.

PRTY Specifies whether priority messages will be used. (Default=NO).

QUIES A connection is to be initiated in quiesced mode. (Default=NO).

MSGLIM The message limit value.

ID Is a VSE/Advanced Functions provided password.

EXIT The address of the exit routine that is to be given control when an interrupt-related event for this user occurs.

Operands Interrelationship with Keywords

OP	CB	VSIUCVU	TRGTID	PATH	UDATA	PRTY	QUIES	MSGLIM	ID	EXIT
OPEN	R	N/A	N/A	N/A	N/A	N/A	N/A	N/A	O	O
CLOS	R	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A
CONN	R	O	O	N/A	O	O	O	O	N/A	N/A
ACPT	R	O	N/A	O	O	N/A	O	N/A	N/A	N/A
SEVR	R	O	N/A	O	N/A	N/A	N/A	N/A	N/A	N/A
SSTE	R	N/A	N/A	N/A	N/A	N/A	N/A	N/A	O	N/A

R = Required parameter O = Optional parameter N/A = Not Applicable

Register Usage:

R0 Used by the macro to pass the operation to the SVC.
R1 Used for the address of the appropriate control block.

Output: Register 15 contains one of the following return codes.

0 (X'00')	No error	All
4 (X'04')	Issuer not in SUPVR state	All
8 (X'08')	Unknown operation code	All
12 (X'0C')	User already active	OPEN
16 (X'10')	Insufficient resources	OPEN
20 (X'14')	User inactive	CONN, ACPT, SEVR, CLOS
24 (X'18')	IUCV detected error	CONN, ACPT, SEVR
28 (X'1C')	No connect pending	ACPT
32 (X'20')	Invalid path ID	ACPT, SEVR
36 (X'24')	Issuer is not path owner	ACPT, SEVR
40 (X'28')	Path inactive	SEVR

A requester of the subsystem support for VM/VCNA (SVC 141) will be canceled with 'ILLEGAL SVC' (ERROR 21) under the following conditions:

- The requester is not authorized (not VM/VCNA).
- The requester for supervisor state is not a main task.
- IUCV is not present in the VM/System Product.
- IUCV is present in the VM/System Product, but IUCV 'QUERY' failed during IPL of VSE/Advanced Functions.

A requester of the subsystem support for VM/VCNA (SVC 141) will be canceled with 'Invalid Address' (ERROR 25) when either the VSIUCVU or VSIUCVPL are not in the corresponding partition. The following examples show how the VSIUCV macro should be used to perform the various operations.

(1) VSIUCV OPEN

OPEN is used to identify a program to VSE/Advanced Functions IUCV support and to establish an environment by which a program can be connected to another program.

```
[name]  VSIUCV CB=addr|(reg),OP=OPEN
               [,ID=name]
               [,EXIT=addr|(reg)]
```

CB must point to a VSIUCVU block.

(2) VSIUCV CLOS

CLOS is used to stop usage of VSE/Advanced Functions IUCV by an application. It immediately severs all paths associated with this user ID.

```
[name]  VSIUCV CB=addr|(reg),OP=CLOS
```

CB must point to a VSIUCVU block.

(3) VSIUCV CONN

CONN is used to initiate a connection between the issuing user and another user of IUVC.

```
[name]  VSIUCV CB=addr|(reg),OP=CONN
               [,VSIUCVU=addr|(reg)]
               [,TRGTID=(id,name)]
               [,UDATA=data|(reg)]
               [,PRTY=YES|NO]
               [,QUIES=YES|NO]
               [,MSGLIM=value|(reg)]
```

CB points to a VSIUCVPL block.

(4) VSIUCV ACPT

ACPT is used to accept a connection request initiated by another user.

```
[name]  VSIUCV CB=addr|(reg),OP=ACPT
               [,VSIUCVU=addr|(reg)]
               [,PATH=number|(reg)]
               [,UDATA=data|(reg)]
               [,QUIES=YES|NO]
```

CB points to an VSIUCVPL block.

PATH is the path ID passed when the user is notified of the incoming connection request.

(5) VSIUCV SEVR

SEVR is used to terminate a previously established connection or to reject an incoming connection request.

```
[name]  VSIUCV CB=addr|(reg),OP=SEVR
               [,VSIUCVU=addr|(reg)]
               [,PATH=number|(reg)]
```

CB points to a VSIUCVPL block.

PATH is the path ID of the path being terminated or rejected.

(6) VSIUCV SSTE

SSTE is used to give the requester supervisor state. This enables the requester to issue macros for VM/VCNA (VTAM communication Network Application) support.

```
[name]  VSIUCV CB=add|(reg),OP=SSTE
               [,ID=name]
```

CB must point to a VSIUCVU block.

ID must be the VSE/Advanced Functions provided password.

VSIUCVPL

This macro is used to create or map the control block used to request the sub-system support for VM/VCNA to perform connection-related services.

The macro has the following format:

```
[name]   VSIUCVPL  [DSECT={YES|NO}]
                   [,VSIUCVU=addr]
                   [,TRGTID={(id,name)|PATH=number}]
                   [,UDATA=data]
                   [,PRTY={YES|NO}]
                   [,QUIES={YES|NO}]
                   [,MSGLIM=value]
```

The operands have the following meaning:

DSECT If DSECT =NO is coded or defaulted to, the macro produces a 48-byte area with a label as specified. If DSECT=YES is coded, the macro produces a DSECT of the area with a label as specified.

VSIUCVU =addr
The address of the VSIUCVU block that identifies the user.

TRGTID 'id' is the VMID of the virtual machine of the target user.
'name' is the identifier by which the target user is known.

PATH number
The path ID or number. PATH and TRGTID are mutually exclusive.

UDATA data
Any 4 bytes of information.

PRTY Specifies whether priority messages will be used. Default=NO.

QUIES Specifies whether a connection is to be initiated in quiesced mode. Default=NO.

MSGLIM value
The message limit value.

VSIUCVU

This macro is used to create or map the control block used to identify the using program to VSE/Advanced Functions IUCV. It is pointed to when the program executes VSIUCV with OP=OPEN, OP=CLOS or OP=SSTE.

The macro has the following format:

```
[name]  VSIUCVU  [ID=name]
                 [,EXIT=addr]
                 [,DSECT={YES|NO}]
```

The operands have the following meaning:

ID An 8-character name that is the identifier by which this user is to be known to VSE/Advanced Functions and to other IUCV users to which this program will be connected. The default is all blanks.

EXIT The address of the exit routine that is to be given control when an interrupt-related event for this user occurs. The default is zero.

DSECT YES: Forces a mapping of the control block to be generated.

NO: Forces a CSECT of the control block to be generated.

WRITEHCF

The macro WRITEHCF ensures that the record specified by the first operand will be written onto the HCF. The record address has initially been provided by the POINTHCF macro and is automatically updated by any subsequent WRITEHCF request (except FORCE=YES). The HCF is written in wrap-around mode.

Note: Only the HCF system task is allowed to write into the hard-copy file.

The macro has the following format:

```
[name]   WRITEHCF {ioarea|(0)}
                  [,FORCE={NO|YES}]
                  [,DUMP=YES]
```

ioarea Symbolic name of the I/O area which contains the record to be written on the HCF.

FORCE YES: Forces the I/O buffer to be immediately written onto the HCF without the necessity that the I/O buffer is full.
The first operand will be ignored, that is, no record is inserted into the I/O buffer.

NO: The I/O buffer will not be written onto the HCF before the I/O buffer is full.

DUMP YES: Indicates that the DUMP program is the issuer of the WRITEHCF macro.

Output: Register 15 contains one of the following return codes.

0 (X'00')	Normal processing successfully completed.
4 (X'04')	Inconsistent input, no WRITE authority
8 (X'08')	No record found, incorrect length.
12 (X'0C')	Unrecoverable I/O error.
16 (X'10')	HCF device is not ready.
20 (X'14')	HCF has just entered the overlay mode.
24 (X'18')	Warning message (HCF is close to overlay mode) must be issued.

Register Usage: The contents of general register 14 through 2 are destroyed by this macro.

XMOVE

The Cross Partition Data Move function (format 1 and 2) allows authorized users to move data from a data area in one partition to a data area in another partition (in any address space). It is not required that one of the data areas resides in the requesters partition.

The supplier of XMOVE is responsible that source and target address cannot become invalid during execution of the XMOVE request.

The XMOVE macro will load Access Register 2 and Access Register 4 with FROM and TO address and will enter Access Register mode. After execution of the XMOVE macro, Access Register mode is switched off.

Format 3 of the Cross Partition Data Move function allows authorized users to post the traffic bit of an ECB and change the status of all tasks, waiting on that ECB, from I/O bound to ready-to-run.

The XMOVE macro requires registers 0 to 5 as input and work registers, register 15 is used as output register and contains the return code.

Invocation

Name	Operation	Operands	Format
[name]	XMOVE	LIST={addr1|(1)}	1
[name]	XMOVE	FADDR={addr1|(reg1)} TOADDR={addr3|(reg3)} ,TOALET={addr6,(reg6)} ,FALET={addr7,(reg7)} DATALEN={addr5|(reg5)}	2
[name]	XMOVE	ECB={addr1|(1)},TASK={addr2|(0)}	3

LIST points to a buffer header followed by a number of list entries. Macro MAPXMOVE contains a DSECT (bilingual) of the buffer list.

bytes 0- 3 Header
- byte 0 number of list entries
- bytes 1,2,3 reserved

bytes 4-27 first list entry
- bytes 4- 7 FADDR
- bytes 8- 11 FALET
- bytes 12- 15 TOADDR
- bytes 16- 19 TOALET
- bytes 20- 21 not used
- bytes 22- 23 not used
- bytes 24- 27 data length

bytes 28-51 next list entry
etc.

FADDR specifies the address from where the data have to be moved.

TOADDR specifies the address to which the data have to be moved.

The addresses specified with TOALET and FALET point to fullwords containing Access-List-Entry Tokens (ALET). They can be retrieved from the supervisor by means of a new GETFLD service (GETFLD FIELD=ALET). SUBSID is necessary for authorization. The ALETs are required. If the requester has not supplied ALETs, the corresponding fields in the parameter list are zero, therefore the result is a move within the requesters partition. Only no ALET or both ALETs can be specified. If register notation is used, the ALET must be contained in the specified register.

DATALEN points to a fullword containing to data length to be moved. If register notation is used, the register has to contain the length. DATALEN is limited to X'FFFFFF' for a list element.

ECB points in both notations to a four-byte field, the ECB.

TASK points in the address notation to a two-byte field, containing the task ID of the task, related to the ECB. In register notation, the specified register has to contain the TASK ID.

Notes:

1. Authorized programs are VTAM and VSE/PT.
2. In case of a ESA supervisor or a VM supervisor the requester has to execute in PSW key zero because XMOVE will issue a normal MVCL.
3. If a IT, OC or PC Exit gets control while XMOVE works in Access Register Mode, the Access Register Mode is removed. When the program gets control lateron, the Access Register Mode is reestablished.
4. If a AB Exit gets control while XMOVE works in Access Register Mode, the Access Register Mode is removed.

Output

After the execution of a XMOVE request, register 15 will contain a return code indicating successful execution or reason why the request is rejected.

When register 15 contains one of the return codes 8 to 28 and if LIST was specified with the request, reg 1 will contain the address of the list entry, responsible for the error return code. All preceding list entries are processed successfully, the following list entries are not processed.

RC 0 : request is executed successfully, data are moved
or task is ready-to-run and traffic bit posted.

RC 4 : request is rejected, the requester is not authorized.

RC 20 : request is rejected, TASK is invalid.

RC 28 : request is rejected, task specified with TASK is in termination.

RC 36 : request is rejected, the function code is invalid.

XPCC

The new FDSCR=INCREP parameter for the XPCC-SENDR function informs the cross partition communication facility, that either the defined reply area (IJBXRADR) may be too small to contain the whole reply or the receiver is allowed to receive the data in several pieces. If not all data are transmitted with the XPCC-REPLY or XPCC-RECEIVE the sender/receiver can receive the rest of the data by one or several additional XPCC-RECEIVE requests. The new FDSCR parameter requires, that both related XPCCB's are generated with parameter VERSION=2.

Buffer lists may contain now up to 256 list entries. The BUFFER parameter in the XPCC-REPLY function may be specified now for a list of data areas.

XPCC SENDR Function

Name	Operation	Operands
[name]	XPCC	FCT=SENDR, ..., [FDSCR={INCREP|(reg)}]

Input : own XPCCB - fields used additionally

Name	Description
IJBXFDSC	Set to IJBXIREP when FDSCR=INCREP specified (If both FDSCR=POSTRCV and FDSCR=INCREP have to be used, register notation is required and the register must be loaded with (IJBXPOST + IJBXIREP))

Output : own XPCCB - fields used additionally

Name	Description
IJBXFDSC	Set to X'00'
IJBXRETC	- Set to IJBXNSTO when no system storage available to contain the specified buffer list. Register 15 will contain 8 in this case. - Set to IJBXOVER when the own XPCCB is not VERSION=2 Register 15 will contain 8 in this case. - Set to IJBXPVER when partners XPCCB is not VERSION=2 Register 15 will contain 8 in this case.

Output : partners XPCCB - fields used additionally

Name	Description
IJBXBSIZ	Size of first data area (if FDSCR=INCREP)

When a SENDR request completes with return code 0 or 4 in register 15, the associated connection becomes busy. It stays busy until all REPLY-ed data are transmitted to the sender's side. Until that point, the SENDR-RECEIVE-REPLY-RECEIVE sequence can be terminated by the sender through a CLEAR request and by the receiver through a PURGE request.

When the SENDR request was specified with FDSCR=INCREP IJBXBSIZ on the other side will contain the length of the first data area (with non zero length) of the sender's buffer list.

The FDSCR=INCREP parameter can only be used, when the associated XPCCB and MAPXPCCB are defined with VERSION=2 parameter. Return codes IJBXOVER or IJBXPVER are returned in IJBXRETC (together with 8 in reg 15) if either the own or the partners XPCCB has not the correct version.

XPCC REPLY Function

Input : own XPCCB - fields used additionally

Name	Description
IJBXIND	B'1XXXXXXX' if only one data area is to be transmitted B'0XXXXXXX' if a list of data areas is provided
IJBXAD31	Address of data area to be transmitted or address of list of data areas.
IJBXBLN	Length of data area to be transmitted. Not used in case of data area list.

Output : own XPCCB - fields used additionally

Name	Description
IJBXFDSC	Set to X'00'
IJBXRETC	- Set to IJBXNSTO when no system storage available to contain the specified buffer list. Register 15 will contain 8 in this case. - Set to IJBXWLST when list contains more then 256 entries Register 15 will contain 8 in this case. - Set to IJBXNOTR when not all data transmitted Register 15 will contain 4 in this case. (i.e. REPLY executed successfully but didn't complete until now. Wait on IJBXSECB required)
IJBXSECB	Reset Wait Bit.

Output : partners XPCCB - fields used additionally

Name	Description
IJBXSLN	Number of bytes transmitted
IJBXSLNR	Total number of bytes not transmitted (Zero if all data transmitted)
IJBXBSIZ	Size of next data area not transmitted

The BUFFER parameter can either define a single data area or point to a list of data areas of 256 entries at most. The layout of a list is not changed.

When the reply buffer on the sender's side is large enough to contain all replied data, the REPLY request completes with return code 0 in register 15 and the connection is free for the next SEND request. In the sender's XPCCB, field IJBXSLN

will contain the number of bytes transmitted to the reply area and field IJBXSLNR will contain 0.

When not all replied data fit into the sender's reply area and the SENDR request was defined with parameter FDSCR=INCREP, the REPLY request completes with return code 4 in register 15 and IJBXRETC is set to IJBXNOTR. The Wait Bit in IJBXSECB is reset and the replier can wait on IJBXSECB until the sender has received the rest of the data. In the sender's XPCCB, field IJBXSLN will contain the number of bytes transmitted to sender's reply area, field IJBXSLNR the number of bytes not transmitted and field IJBXBSIZ the number of bytes of the next data area (minimum BUFFER size for next RECEIVE on sender's side).

Only complete data areas are transmitted with the REPLY function.

PURGE is possible as long as not all data are moved to the sender's side.

XPCC RECEIVE Function

Name	Operation	Operands
[name]	XPCC	FCT=RECEIVE, ...

Output : own XPCCB fields used additionally

Name	Description
IJBXFDSC	Set to X'00'
IJBXRETC	- Set to IJBXMODA when not all data received with this RECEIVE Register 15 will contain 4 in this case. - Set to IJBXWRAR when the specified buffer is too small to contain the data sent with SEND or SENDR (FDSCR¬=INCREP) or the specified buffer is too small to contain at least one data area of the data sent with SENDR FDSCR=INCREP or REPLY Register 15 will contain 8 in this case.
IJBXSLN	Number of bytes transmitted
IJBXSLNR	Number of bytes still not transmitted Zero if all data transmitted
IJBXBSIZ	Size of next data area to be transmitted

The XPCC-RECEIVE function is required when an application is posted by a SEND, SENDR or a REPLY request. When posted by SEND or SENDR (with FDSCR ¬= INCREP) the RECEIVE function works as described in *System Macros Reference (SC33-6616) and System Macros User's Guide (SC33-6615)*. When posted by SENDR FDSCR=INCREP or REPLY the application is not forced to receive the data with a single RECEIVE. The receive buffer may be defined with a minimum size as retrieved from field IJBXBSIZ. XPCC will transmit only as many data areas of the sender/replier as fit completely into the receive buffer.

When XPCC-RECEIVE completes, the receive buffer contains as many data as indicated in IJBXSLN. IJBXSLNR contains the total number of bytes not transmitted until now and IJBXBSIZ the size of the next data area not transmitted.

If IJBXSLNR is not zero, the RECEIVE completes with IJBXMODA in field IJBXRETC and register 15 contains 4.

If IJBXSLNR becomes zero after RECEIVE, all data are transmitted. IJBXSECB on the other side is posted and the sender/replier is taken out of wait state.

If the input area is too small to contain at least one data area of the sender/replier, return code 8 is loaded into register 15 and IJBXRETC is set to IJBXWRAR. Field IJBXBSIZ contains the minimum length required for BUFFER parameter to execute RECEIVE successfully.

VSE Supervisor Call Table

SVC Code		Imperative Macro that Issues the SVC	Activation Option to be Specified	Function	Execution Mode xyzab x=1|0: AMODE ANY|24 y=1|0: ARMODE YES|NO z=1|0: RMODE ANY|24 b=1|0: XMODE YES|NO
DEC	HEX				x y z a b
0	00	EXCP	none	Execute channel program	1 0 1
1	01	FETCH	none	Fetch a phase, except a transient phase	1 0 0
2	02		none	Fetch a logical transient phase ($$B.....)	1 0 1
3	03		none	Quiesce I/O	1 0 1
4	04	LOAD/SLOAD	none	Load a phase	1 0 0
5	05	MVCOM	none	Modify the partition communication-region	1 0 1
		if issued by ERP-Task	none	Fetch a physical transient ($$A.....)	
6	06	CANCEL	none	Cancel a problem program or a task	1 0 1
7	07	WAIT	none	Wait for the posting of a control block (CCB, IORB, ECB, TECB)	1 0 1
8	08		none	Transfer control from a logical transient to a problem program	0 0 0
9	09	LBRET	none	Return from the problem program to the logical transient which issued SVC 8	0 0 0
10	0A	SETIME	none	Set interval timer	1 0 1
11	0B		none	Final return from a logical transient	0 0 0

Figure 27 (Part 1 of 12). VSE Supervisor Calls

SVC Code		Imperative Macro that Issues the SVC	Activation Option to be Specified	Function	Execution Mode xyzab x=1|0: AMODE ANY|24 y=1|0: ARMODE YES|NO z=1|0: RMODE ANY|24 b=1|0: XMODE YES|NO
DEC	HEX				x y z a b
12	0C		none	Reset switches in the partition communication region (COMREG)	1 0 1
13	0D		none	Set switches in the partition communication region (COMREG)	1 0 1
14	0E	EOJ	none	Terminate a job and go to job control for end of job step processing	1 0 1
15	0F	SYSIO	none	Head queue I/O request and execute the channel program	1 0 1
16	10	STXIT PC	none	Establish/reset linkage to user's PC routine for program check interrupts	1 0 1
17	11	EXIT PC	none	Return from the user's PC routine	1 0 1
18	12	STXIT IT	none	Establish/reset linkage to user's IT routine for interval timer interrupts	1 0 1
19	13	EXIT IT	none	Return from the user's IT routine	1 0 1
20	14	STXIT OC	none	Establish/reset linkage to user's OC routine in case of attention MSG command	1 0 1
21	15	EXIT OC	none	Return from the user's OC routine	1 0 1
22	16		none	SEIZE or RELEASE the system; enable or disable for external and I/O interrupts; set the key in a user's PSW	0 0 0

Figure 27 (Part 2 of 12). VSE Supervisor Calls

SVC Code		Imperative Macro that Issues the SVC	Activation Option to be Specified	Function	Execution Mode xyzab x=1|0: AMODE ANY|24 y=1|0: ARMODE YES|NO z=1|0: RMODE ANY|24 b=1|0: XMODE YES|NO
DEC	HEX				x y z a b
23	17		none	Store the LOAD ADDRESS of a phase at a defined user address	0 0 0
24	18	SETIME	none	Set TIMER INTERVAL and establish accessibility to user's TECB	1 0 1
25	19	HALTIO	none	Issue an HDV for a telecommunication device or for any device if issued by OLTEP.	1 0 1
26	1A		none	Validate address limits	0 0 0
27	1B		none	Issue an HDV for a tele—communication device without dequeueing the the CHANQ entry	0 0 0
28	1C			Reserved	
29	1D	WAITM	none	Wait for the posting of one of the control blocks specified	1 0 1
30	1E		none	Submit command to AR	1 0 1
31	1F	REIPL	none	Force software Re-IPL	0 0 0
32	20	WTO/WTOR	none	WRITE/READ to operator	1 0 1
33	21	COMRG	none	Force task selection	1 0 1
34	22	GETIME	none	Provide the time and update	1 0 1
35	23		TRKHLD=YES in FOPT	Hold a track for exclusive use by the requesting task	0 0 0
36	24	FREE	TRKHLD=YES in FOPT	Free a track held by the requesting task	0 0 0

Figure 27 (Part 3 of 12). VSE Supervisor Calls

SVC Code		Imperative Macro that Issues the SVC	Activation Option to be Specified	Function	Execution Mode xyz x=1|0: AMODE ANY|24 y=1|0: ARMODE YES|NO z=1|0: RMODE ANY|24 x y z
DEC	HEX				
37	25	STXIT AB	none	Establish/reset linkage to user's AB routine for abnormal termination of a task	1 0 1
38	26	ATTACH	none	Initialize a subtask and establish its processing priority	1 0 1
39	27	DETACH	none	Terminate a subtask; free resources that might be held by the subtask	1 0 1
40	28	POST	none	Indicate occurrence of an event and ready any waiting task	1 0 1
41	29	DEQ	none	Indicate that a previously enqueued resource is available again	1 0 1
42	2A	ENQ	none	Prevent two or more tasks from simultaneously manipulating a shared resource (e.g. data area)	1 0 1
43	2B	DYNCLASS	none	Tasking services	1 0 1
44	2C		none	Force a unit check record to be written onto the recorder file	0 0 0
45	2D		none	Reserved.	0 0 0
46	2E		none	Allow OLTEP to run in supervisor state	0 0 0
47	2F			Reserved	0 0 0
48	30		none	PLF Processing	1 0 1

Figure 27 (Part 4 of 12). VSE Supervisor Calls

SVC Code		Imperative Macro that Issues the SVC	Activation Option to be Specified	Function	Execution Mode xyzab x=1|0: AMODE ANY|24 y=1|0: ARMODE YES|NO z=1|0: RMODE ANY|24 b=1|0: XMODE YES|NO
DEC	HEX				x y z a b
49	31		none	Allow ACF/VTAM to initiate the execution of a channel program	1 0 1
50	32		none	Used by LIOCS to cancel user indicating illegal SVC	1 0 1
51	33	HIPROG	none	Make directory entry information for a phase available to the requesting task	1 0 0
			none	Calculate the highest address of an overlay structure of phases or of one phase only and store it in the COMREG	
52	34	TTIMER	none	Return the remaining time interval or cancel a time interval	1 0 1
53	35		none	Allow ACF/VTAM to schedule a user exit in an application program	1 0 1
54	36		none	Release page frames to selection pool	0 0 0
55	37		none	Allow SDAID to acquire processor storage needed for program initialization	0 0 0
56	38	CPCLOSE	none	Support the VSE/POWER–CP interface when VSE operates under VM.	0 0 0
57	39	GETPRTY	none	Return partition priorities to the requesting task	1 0 1

Figure 27 (Part 5 of 12). VSE Supervisor Calls

SVC Code		Imperative Macro that Issues the SVC	Activation Option to be Specified	Function	Execution Mode xyzab x=1|0: AMODE ANY|24 y=1|0: ARMODE YES|NO z=1|0: RMODE ANY|24 b=1|0: XMODE YES|NO
DEC	HEX				x y z a b
		SETPRTY	none	Change partition priorities as specified	1 0 1
58	3A	INVPART	none	Initialize partition	0 0 0
59	3B		none	Reserved	0 0 0
60	3C	GETDADR	none	Return the virtual equivalent of a real I/O area plus offset	0 0 0
61	3D	GETVIS	none	Request allocation of storage within the same partition, the SVA or the dynamic space getvis area	1 0 1
62	3E	FREEVIS	none	Free storage requested through a GETVIS macro	1 0 1
63	3F	USE	none	Indicate system resource is in USE	0 0 0
64	40	RELEASE	none	RELEASE a system resource	0 0 0
65	41	CDLOAD	none	Load a phase in the requesting partition's GETVIS area unless that phase is already in the SVA	1 0 1
		CDDELETE	none	Delete a phase previously loaded by a CDLOAD	1 0 1
66	42	RUNMODE	none	Return the system's operating mode	0 0 0
67	43	PFIX	none	FIX pages in processor storage	0 0 0
68	44	PFREE	none	FREE pages in processor storage	0 0 0
69	45	REALAD	none	Return the REAL address corresponding to a given virtual address	1 0 1

Figure 27 (Part 6 of 12). VSE Supervisor Calls

SVC Code		Imperative Macro that Issues the SVC	Activation Option to be Specified	Function	Execution Mode xyzab x=1|0: AMODE ANY|24 y=1|0: ARMODE YES|NO z=1|0: RMODE ANY|24 b=1|0: XMODE YES|NO
DEC	HEX				x y z a b
70	46	VIRTAD	none	Return the virtual address corresponding to a given real address	1 0 1
71	47	SETPFA	none	Establish or terminate linkage to a user Page Fault Appendage routine	0 0 0
72	48	GETCBUF	none	GET Copy buffer for IDAL of tape ERP	1 0 1
		FREECBUF		FREE Copy BUFfer for IDAL of tape ERP	1 0 1
73	49	SETAPP	none	Allow linkage to channel–end appendage routines	1 0 1
74	4A	PFIXREST	none	Fix page(s) in processor storage for restart	0 0 0
		PFIXCHPT	none	Build parameter list for PFIXREST during checkpointing	
75	4B	SECTVAL		Sector value calculation	1 0 1
76	4C		none	Initiate recording on VM recorder file	0 0 0
77	4D	TRANSCSW	none	Returns the virtual address of an ERP CCW address copied from the pertinent CSW	0 0 0
78	4E	CHAP	none	Change the processing priority of the requesting task	1 0 1
79	4F		none	X-mode SVC	1 1 1 0 1
80	50		none	Reserved	0 0 0

Figure 27 (Part 7 of 12). VSE Supervisor Calls

SVC Code		Imperative Macro that Issues the SVC	Activation Option to be Specified	Function	Execution Mode xyzab x=1|0: AMODE ANY|24 y=1|0: ARMODE YES|NO z=1|0: RMODE ANY|24 b=1|0: XMODE YES|NO
DEC	HEX				x y z a b
81	51		none	Reserved	0 0 0
82	52		none	Set monitor call and/or branch, for ICCF	1 0 1
83	53	ALLOCATE	none	Allocate real or virtual partitions	1 0 0
84	54	SETLIMIT	none	Set partition sizes	0 0 0
85	55	RELPAG	none	Release the contents of one or more pages	0 0 0
86	56	FCEPGOUT	none	Force a page—out operation for more pages	0 0 0
87	57	PAGEIN	none	Request a page—in operation for more pages	0 0 0
88	58	TPIN	none	Start TP balancing	0 0 0
89	59	TPOUT	none	Stop TP balancing	1 0 1
90	5A	PUTACCT	JA=YES in IPL SYS—CMD	Provide interface with VSE/POWER for additional, user—provided account information	0 0 0
91	5B		JA=YES in IPL SYS—CMD	Provide interface with VSE/POWER for standard account information	0 0 0
92	5C	XECBTAB	none	Define, delete, or check an entry in the cross—partition ECB table	0 0 0
93	5D	XPOST	none	Set the traffic bit in a cross—partition ECB and ready any waiting tasks	0 0 0
94	5E	XWAIT	none	Wait for a cross—partition ECB to be posted	0 0 0

Figure 27 (Part 8 of 12). VSE Supervisor Calls

SVC Code		Imperative Macro that Issues the SVC	Activation Option to be Specified	Function	Execution Mode xyzab x=1|0: AMODE ANY|24 y=1|0: ARMODE YES|NO z=1|0: RMODE ANY|24 b=1|0: XMODE YES|NO
DEC	HEX				x y z a b
95	5F	EXIT AB	none	Return from a user's abnormal termination routine	1 0 1
96	60		none	Reserved	0 0 0
97	61		none	Reserved	0 0 0
98	62	EXTRACT	none	Extract system control information	1 0 1
		MODCTB	none	Modify a PUB2 table entry	1 0 1
99	63	GETVCE	none	Return a specific volume characteristics and/or track balance information	
100	64			Reserved	0 0 0
101	65	MODVCE	none	Update the volume characteristics table	1 0 1
102	66	GETJA	JA=YES in IPL SYS—CMD	Update the fields in the requesting partition's job accounting table	0 0 0
103	67		none	Execute I/O operations for SYSFIL on on FBA device, if FBA supported	0 0 0
104	68	EXTENT	none	Add, return, or delete DASD extent information	0 0 0
105	69	SUBSID	none	Accept, return, and delete subsystem identification information.	1 0 1

Figure 27 (Part 9 of 12). VSE Supervisor Calls

SVC Code		Imperative Macro that Issues the SVC	Activation Option to be Specified	Function	Execution Mode xyzab x=1|0: AMODE ANY|24 y=1|0: ARMODE YES|NO z=1|0: RMODE ANY|24 b=1|0: XMODE YES|NO
DEC	HEX				x y z a b
106	6A		none	Set the storage key for a specific area to the value in Register 0 (ICCF)	0 0 0
107	6B	DEVREL	none	Release a device that was "in use"	1 0 1
		DEVUSE		Force a device to be set "in use"	
		GETFLD		Retrieve task-related information	
		MODFLD		Modify task-related information	
		RLOCK		Obtain access to a specified resource or wait for it	
		SENTER		Enter a sub-system	
		SLEAVE		Leave a sub-system	
		TREADY		Post or cancel a task	
		TSTOP		Deactivate current task or partition	
		VIO POINT		Piont to VIO control block (VIORB)	
108	6C	SECHECK	SEC=nn in IPL SYS-CMD	Check user's authority for accessing the specified resource	0 0 0
109	6D	PAGESTAT	none	Return status of a page or a set of pages	1 0 1
110	6E	LOCK/UNLOCK	none	Protect or release a serially re-usable resource against concurrent access of two or more tasks	1 0 1
111	6F		none	Reserved	0 0 0
112	70	MSAT	none	Build, return, or delete stored assignment information	1 0 1
113	71	XPCC	none	Cross-partition communication services	1 0 1

Figure 27 (Part 10 of 12). VSE Supervisor Calls

SVC Code		Imperative Macro that Issues the SVC	Activation Option to be Specified	Function	Execution Mode xyzab x=1|0: AMODE ANY|24 y=1|0: ARMODE YES|NO z=1|0: RMODE ANY|24 b=1|0: XMODE YES|NO
DEC	HEX				x y z a b
114	72	VIO	none	Allocate, deallocate or extend VIO file	0 0 0
115	73	PWROFF	none	Disconnect VM Guest	0 0 0
116	74	NPGR	none	Allocate or reallocate programmer LUB's	0 0 0
117	75		none	TD start	1 0 1
118	76	CPCOM	none	CP command interface (CPCOM macro)	0 0 0
119	77		none	SPDT support	0 0 0
120	78	XMOVE	none	Cross partition move/post	1 0 1
121	79		none	Page manager services	1 0 1
122	7A	SYSDEF	none	SYSDEF command: define data space characteristics	1 0 1
123	7B	SDUMP(X)	none	Dump virtual storage (address/data space)	1 1 1
124	7C	PRODID PRODEXIT	none	Vendor I/F Vendor Exit Services	1 0 1
125	7D		none	Reserved	0 0 0
126	7E		none	Reserved	0 0 0
127	7F		none	Reserved	0 0 0
128	80		none	Reserved	0 0 0
129	81		none	Reserved	0 0 0
130	82		none	Check exection mode of BAM caller	0 0 0

Figure 27 (Part 11 of 12). VSE Supervisor Calls

SVC Code		Imperative Macro that Issues the SVC	Activation Option to be Specified	Function	Execution Mode xyzab x=1|0: AMODE ANY|24 y=1|0: ARMODE YES|NO z=1|0: RMODE ANY|24 b=1|0: XMODE YES|NO
DEC	HEX				x y z a b
131	83		none	SIM SVC	1 1 1
132	84		none	SIM SVC	1 1 1
133	85		none	JCL SVC	1 0 0
134	86		none	Reserved	0 0 0
		•	•	•	
		•	•	•	
		•	•	•	
140	8C		none	TCP/IP SVC	1 0 1
141	8D	VSIUCV	none	Provide subsystem support for VM/VCNA (VTAM Communication Network Application)	1 0 1
142	8E		none	Reserved	
143	8F		none	Reserved	
144	90		none	Reserved	
145	91		none	Reserved	
146	92		none	Reserved	
147	93		none	Reserved	
148	94		none	Reserved	
149	95		none	Reserved	
150	96		none	CICS SVC	1 0 1
151	97		none	Reserved	
		•	•	•	
		•	•	•	
		•	•	•	
255	FF		none	Reserved	

Figure 27 (Part 12 of 12). VSE Supervisor Calls

SVC 0 (X'00' - EXCP)

Executes a channel program (EXCP). The address of the user's control block (CCB/IORB) is contained in general register 1.

An internal I/O scheduling priority is assigned to each partition to determine the proper place within the I/O request chain queued to the requested I/O device. This scheduling priority is initially equal for all partitions and can be changed by the PRTYIO command only.

SVC 1 (X'01' - FETCH)

Fetches a phase. A FETCH loads a phase from the IJSYSRS.SYSLIB sublibrary (SYSLIB) or one of the concatenated private sublibraries (PSUBLIB) and branches to the entry address in that phase.

The directory entry may be found in the SYSTEM DIRECTORY LIST (SDL), in local entries, in one of the PSUBLIB directories (if there are concatenated sublibraries assigned), or in the SYSLIB directory. A phase residing in the SVA is not loaded into the user partition.

The load and entry addresses are obtained from the directory entry for the phase being fetched. The storage address of the phase name or the address of the parameter list must be supplied in general register 1 before this SVC is issued. For a relocatable phase the ENTRY and LOAD address is relocated. The entry address contained in the associated directory entry can be overridden by a user-supplied entry address in general register 0.

If the access control option is active, the SVC routine checks whether the issuer is authorized to fetch the phase.

SVC 2 (X'02')

Fetches a logical transient ($$B- or B-Transient).
Loads a B-transient from the SVA, the IJSYSRS.SYSLIB sublibrary (SYSLIB) or one of the concatenated private sublibraries (PSUBLIB) into the Logical Transient Area (LTA) and enters the logical transient at its load address plus 8 bytes. The directory entry for the phase may be found in the SDL, in the SYSLIB directory, or in one of the PSUBLIB directories (if there are concatenated sublibraries). For a more detailed description, refer to z/VSE Supervisor Diagnosis Reference,Chapter "Program Retrieval".

If the VSE/Advanced Functions "Fast B- and C-Transient Fetch" is supported, the logical transients can be loaded into the LTA without activating the Fetch (FCH) system tasks.

The SVC 2 (X'02) routine moves the phase from the SVA directly into the LTA if all of the following conditions are met:

- The directory entry of the phase has been found in the SDL.
- The directory entry of the phase has already been activated.
- The phase resides in the SVA and it is self-relocatable.

The storage address of the logical transient phase name must be supplied in general register 1 before this SVC is issued. To ensure system integrity, it is required that the logical transient phase name starts with $$B.

The logical transient is loaded at the origin of the LTA and this address is put into general register 15, which may then be used by the transient as a base register.

Only one task can use the LTA at a time. If the area is already occupied by another task, the task requesting the LTA will be set "LTABND" until the LTA is released by the occupying task.

Special handling is done for the following routines:

- $$BACLOS
- $$BDUMP
- $$BEOJ3A
- $$BEOJ4
- $$BJDUMP
- $$BPDUMP
- $$BCHKPD
- $$BCHKPT

If an SVC 2 (X'02') has been issued for one of these phases, the Terminator is entered and the SVA resident routine gets control.

If Access Control option is active, B-Transient routines must reside in protected libraries.

Note: A VTAM request issued in AMODE 31 is cancelled with execution mode violation although AMODE ANY is generally allowed for SVC 2.

SVC 3 (X'03')

Provides an interface between the supervisor and $IJBSEOT. An SVC 3 (X'03') waits for termination of I/O requests that belong to the partition or task that is being canceled or has reached the End-Of-Job step.

SVC 4 (X'04' - LOAD)

A phase from the IJSYSRS.SYSLIB sublibrary (SYSLIB) or one of the concatenated private sublibraries (PSUBLIB) is loaded and control is returned to the requesting task.

The directory entry may be found in the SDL, in local entries, in one of the active PSUBLIB directories or in the SYSLIB directory. A phase residing in the SVA is not loaded into the user partition.

The storage address of the phase name or the address of a parameter list must be supplied in general register 1 before this SVC is issued. The user may override the link edited load address by supplying a load address in general register 0.

Upon return to the user, general register 1 contains the actual (relocated) entry point address of the phase. General register 0 points to the active directory entry in storage if a local directory list was supplied (parameter LIST=) and the phase was found in this local list. If the phase was not found in the list, register 0 contains 0. (In this case, the load must be performed from the sublibraries or the SDL/SVA).

If the access control option is active, the SVC routine checks whether the issuer is allowed to load this phase.

SVC 5 (X'05' - MVCOM)

When issued by a user through a MVCOM macro, it modifies the partition communication region in the supervisor as specified by the parameters of the MVCOM macro.

When issued by the ERP system task, another physical transient phase, the name of which is contained in the Error Block (ERBLOC) is requested to be loaded into the physical transient area (PTA), and to be entered at its load address plus 10 bytes. If the transient name is $$ABERRL resident ERP message writer SGEMSG is entered.

SVC 6 (X'06' - CANCEL)

Cancels a program, task, tree of tasks, or partition. This is usually achieved by the requesting program, task, or subtask issuing a CANCEL or CANCEL ALL macro.

If a subtask issues CANCEL, that subtask is canceled. If the canceled subtask is not running in OS/390 compatability mode, then only that subtask is canceled. Otherwise all subtasks in the subtree starting at the subtask issuing CANCEL are canceled. Is a maintask issues CANCEL, or a subtask issues CANCEL ALL, then the entire partition is canceled. The maintask is always the last one to be terminated.

- CANCEL macro issued by a maintask without subtasks:
  the issuing task is terminated normally:
  - Cancel code 35 (X'23') is posted to the issuer's TIB.
  - Message '(issuer*) A CANCEL OR CANCEL ALL MACRO WAS ISSUED'.
  - Message '(issuer*) JOB (jobname) WAS CANCELED'.
- CANCEL macro issued by a subtask:
  the issuing subtask is terminated normally, and the action of the SVC 6 (X'06') routine for this subtask is the same as described above for a maintask without subtasks:
  - Cancel code 35 (X'23') is posted to the issuer's TIB.
  - Message '(issuer*) A CANCEL OR CANCEL ALL MACRO WAS ISSUED'.
  - Message '(issuer*) SUB TASK (sub id) WAS CANCELED'.
- CANCEL macro issued by a subtask running in OS/390 compatability mode and with subtasks attached:
  the issuing subtask is terminated normally; subtasks in the tree starting at the subtask issuing CANCEL are terminated abnormally.
  - Cancel code 29 (X'1D') is posted to each subtask TIB.
  - Cancel code 35 (X'23') is posted to the issuer's TIB.
  - Message '(subtask*) MAIN TASK TERMINATION'.
  - Message '(subtask*) SUBTASK (sub name) CANCELED'.
  - Message '(issuer*) A CANCEL OR CANCEL ALL MACRO WAS ISSUED'.
  - Message '(issuer*) SUB (sub id) CANCELED'.
- CANCEL macro issued by a maintask with subtasks attached:
  the maintask is terminated normally; attached subtasks are terminated abnormally.
  - Cancel code 29 (X'1D') is posted to each subtask TIB.
  - Cancel code 23 (X'17') is posted to the maintask TIB.

  - Message '(subtask*) MAIN TASK TERMINATION'.
  - Message '(subtask*) SUB TASK (sub name) CANCELED'.
  - Message '(issuer*) A CANCEL OR CANCEL ALL MACRO WAS ISSUED'.
  - Message '(issuer*) JOB (jobname) WAS CANCELED'.
  - A dump is forced at the start of the termination of the maintask if the DUMP option is active (DUMP=YES).

- CANCEL ALL macro issued by a subtask:
  the issuing subtask is terminated normally; other subtasks and the maintask are terminated abnormally.
  - Cancel code 35 (X'23') is posted to the issuing subtask TIB.
  - Cancel code 28 (X'1C') is posted to each of the other subtasks PIBs and to the maintask TIB.
  - Message '(issuer*) A CANCEL OR CANCEL ALL MACRO WAS ISSUED'.
  - Message '(issuer*) SUB TASK (sub name) CANCELED'.
  - Message '(main or subtask*) CANCELED DUE TO CANCEL ALL MACRO'.
  - A dump is forced at termination of the subtask, if the DUMP option is active (DUMP=YES).

Note: * This is the program name as contained in the task or subtask save area

If linkages to a user's AB routines have been established through the STXIT (AB) macro, these routines are entered for all tasks that are terminated abnormally by the task that issues the CANCEL or CANCEL ALL macro.

A task, however, that issues an SVC 6 (X'06') never enters its AB routine assuming the CANCEL (ALL) was not issued from within the LTA or a functionally related routine residing in the SVA.

An AB routine normally terminates through a DETACH, EOJ, or CANCEL macro, but an abnormal condition encountered in an AB routine also terminates that AB routine.

SVC 7 (X'07' - WAIT)

Provides the supervisor service for the WAIT macro and waits for the completion of a special event. General Register 1 points to the control block that is to be inspected for the occurrence of a special event (CCB/IORB/TECB/ECB). The different events that a user can wait on are:

- I/O Interrupt,
- Timer Interrupt,
- Program POST request.

If the event bit (byte 2, bit 0 in the control block) *has not* been turned on, the task status of the issuing task is set to "I/O-BOUND" and its PSW is set up to reissue the SVC 7 (X'07'), whereas, in case the event *has already* occurred, control is directly passed back to the issuing task to the instruction following the SVC X'07' instruction.

SVC 8 (X'08')

Provides the supervisor support to temporarily return from a logical transient to the problem program. This SVC may be issued only from the logical transient area (LTA) and *does not* free this area. The entry address to the problem program must be specified in general register 14. The problem program gets always control in AMODE 24, independent on the original AMODE.

The task selection routine loads the problem program registers. General registers 0 and 1 are passed unchanged to the problem program. To return to the logical transient, the problem program must issue an SVC 9 (X'09').

SVC 9 (X'09' - LBRET)

Provides the supervisor support to pass back control from the problem program to the logical transient routine. An SVC 9 (X'09') can only be issued by the problem program. The task selection routine loads the logical transient registers. General registers 0 and 1 are passed unchanged to the logical transient routine.

SVC 10 (X'0A' - SETIME)

Sets a timer interval. The specified time interval is added to the present value of the TOD clock to get the absolute time, when the clock comparator interrupt is to occur (Expiration Time). If an IT-exit control block (macro EXITBDY) is not available it is created and its address is saved in TCBXTAVS in the TCB-extension. Expiration Time is stored into the IT-exit control block and is also stored into the TIBITREQ field of the TIB. The task's TIB is inserted in ascending order of the calculated timer value into the ITREQ-chain (anchor ITREQPTR in macro SMICR). If the new value is smaller than all the values already contained in the chain, that is, the new value is the first entry in the chain, and if the current clock comparator value is larger, than this value is stored into the clock comparator. When the clock comparator interrupt occurs a previously defined Timer Exit routine gets control and the related TIB is removed from the ITREQ-chain.

SVC 11 (X'0B')

Returns from a B-transient to the problem program releasing the B-transient area (LTA). SVC 11 (X'0B') is invalid if issued by a phase or program other than an active B-transient. The logical transient area is released for use by other tasks.

SVC 11 (X'0B') is also used to return from the SVA resident Terminator or EOT routine to the supervisor. The terminator routine in the supervisor releases the SVA resident terminator routine for use by other tasks.

SVC 12 (X'0C')

Register 1 must contain a non-zero value. The function of this SVC depends on byte 0 of register 1.

- If byte 0 of general register 1 is X'FF', this SVC provides the supervisor support to set or reset flags in a specified byte of the partition's communication region. The user has provided a displacement in byte 2 and a mask in byte 3 of general register 1. The mask is ANDed with the byte at the specified displacement in the partition communication region. If byte 2 and 3 are X'00FF', SVC 12 (X'0C') supplies the Partition protection key in the PSW.
- If the byte contains a value other than X'FF', the SVC performs an AND operation on the linkage editor control byte at displacement 57 of the partition's communication region using as a mask the byte pointed to by bytes 1 through 3 of register 1.

  Note: If this interface is used AMODE 24 and RMODE 24 is required.

SVC 13 (X'0D')

Register 1 must contain a non-zero value. The function of this SVC depends on byte 0 of general register 1.

- If this byte is not X'FF', this SVC supplies the supervisor support to set flags in the linkage control byte (displacement 57 in the partition communications region). The user has provided the address of a mask (1 byte) in general register 1. This mask is ORed with the linkage control byte.

  Note: If this interface is used AMODE 24 and RMODE 24 is required.

- If byte 0 of general register 1 is X'FF', this SVC supplies the supervisor support to set flags in a specified byte of the partition communications region. The user has provided a displacement in byte 2 and a mask in byte 3 of general register 1. The mask is ORed with the byte at the specified displacement in the partition communication region. If byte 2 and 3 are zero, the SVC 13 (X'0D') supplies the PSW key zero.

SVC 14 (X'0E' - EOJ)

This is the normal End Of Job (EOJ) service. Cancel code 16 (X'10') is posted to the task information block (TIB) for the program issuing the SVC 14 (X'0E'). The next time the terminated program is selected by the task selection routine, a branch is made to the Terminator routines.

If any EOJ clean-up routine returns with SVC 14 from SVA, the LTA is freed (if owned by clean-up routine) and clean-up (EOJ-) processing is continued.

SVC 15 (X'0F' - SYSIO)

Provides the supervisor service for the SYSIO macro and executes a channel program *prior to* normal (SVC 0) I/O requests, already enqueued for the same device but not yet started (head queueing). The address of the I/O control block (CCB/IORB) is contained in register 1. SVC 15 (X'0F') can be used by system tasks only.

Processing of the SVC 15 (X'0F') is similar to SVC 0 processing (refer to channel and device scheduling later in this chapter) except that the SVC 15 (X'0F') does provide the ability to make use of reserved channel queue entries.

An internal I/O scheduling priority is assigned to each System task to determine the proper place within the I/O request chain queued to the requested I/O device. This scheduling priority differs from the dispatching priority and is determined by the head queue request priority table (HQUPRI). The system task ID is used to index a 1-byte field in this table which contains the I/O scheduling priority. From its nature, an SVC 15 (X'0F') always supersedes an SVC 0 regardless of its associated priority.

SVC 16 (X'10' - STXIT PC)

Provides support for the STXIT PC macro. It establishes linkage from the supervisor to a Program Check Exit routine in a problem program. It stores the address of the Exit routine, the caller's PSW key and the address of the save area (see mapping macro MAPSAVAR for layout) in the PC-exit control block (macro EXITBDY). The anchor to the PC-exit control block is in field TCBPCEX in the tasks TCB.

SVC 17 (X'11' - EXIT PC)

Provides supervisor support for the EXIT PC macro. Returns from the user's PC routine to the next sequential instruction in the program that was interrupted due to a program check. This is accomplished by copying the contents of the user-supplied save area to the problem program save area.

SVC 18 (X'12' - STXIT IT)

Provides support for the STXIT IT macro. It establishes linkage from the supervisor to a Timer Exit routine in a problem program which is to be entered, when a specified time interval is elapsed (the time interval has to be set with SVC 10). It stores the address of the Exit routine, the caller's PSW key and the address of the save area in the IT-exit control block (macro EXITBDY). The address to the IT-exit control block is contained in TCBXTAVS in the TCB-extension.

SVC 19 (X'13' - EXIT IT)

Supervisor support for the EXIT IT macro. Returns from the user's IT routine to the next sequential instruction in the program that was interrupted due to the clock comparator interrupt. This is accomplished by copying the contents of the user-supplied save area to the problem program save area.

SVC 20 (X'14' - STXIT OC)

Provides support for the STXIT OC macro. It establishes linkage from the supervisor to a Operator Communication Exit routine in a problem program. It stores the address of the Exit routine, the caller's PSW key and the address of the save area (macro MAPSAVAR) in the OC-exit control block (macro EXITBDY). The address to the OC-exit control block is contained in field PCBOCPTR in the partitions PCB. Only the maintask can process the interruption.

SVC 21 (X'15' - EXIT OC)

Supervisor support for the EXIT OC macro. Returns from the user's OC routine to the next sequential instruction in the program that was interrupted by the attention routine MSG command. This is accomplished by copying the contents of the user-supplied save area to the problem program save area.

SVC 22 (X'16')

Indicates to the system, that the issuing task does not allow another task to issue a SVC 22 (X'16' - SEIZE) until this same task has issued another SVC 22 (X'16' - RELEASE). This SVC is intended for system components only. The PSW protection key must be zero, otherwise the issuing program is canceled (ERR21). Any SEIZE request from a task while another task has already got the SEIZE state will result in setting the requesting task "SEIZEBND (X'73')".

If byte 3 of general register 0 is zero, the system mask is set to disable I/O and external interrupts; if the byte is not zero, the system mask is set to enable I/O and external interrupts.

If general register 0 is negative, the user protection key is set in the user's PSW.

SVC 23 (X'17')

Retrieves the load address for a specified phase from the directory entry for the phase. The program issuing an SVC 23 (X'17') is canceled if the PSW protection key is not zero (only job control and B-transient programs can issue an SVC 23 (X'17')).

Register 1 contains the storage address of the phase name and register 0 contains the address where the load address is to be stored (R0 must point to a storage location below 16MB). If the phase is relocatable, the load address returned is the relocated load address. The high order byte of the storage area is not changed. There SVC 23 can only be used for phases loaded below 16 MB (otherwise the request is cancelled with ERR21).

The fetch routine scans the SDL, the IJSYSRS.SYSLIB sublibrary (SYSLIB) and the active chain of private sublibraries (PSUBLIB) for the directory entry of the phase.

SVC 24 (X'18' - SETIME)

Provides the supervisor service for the SETIME macro. The address of the user's Timer Event Control Block (TECB) is contained in general register 0 and the time interval that the user wants to be set is contained in general register 1. The TECB address and the Expiration Time, computed from the time interval in register 1 are saved into the IT-exit control block (macro EXITBDY). The pointer to the control block is contained in TCBXTAVS. Save area address in the IT-control block is set to zero to indicate an SVC24 request. This service resets the event bit (byte 2 bit 0) in the TECB and inserts the task in the proper location in the ITREQ-chain (see SVC10 for more detail).

The event bit is set when the clock comparator interrupt occurs.
So, if the issuing task wants to wait for this event, it has to issue a WAIT or WAITM (SVC 7 or SVC 29 (X'1D')) macro.

SVC 25 (X'19' - HALTIO)

Provides the supervisor service for the HALTIO macro. Halts an I/O operation on a specified device. The address of an I/O control block (CCB / IORB) is contained in general register 1.
If the SVC 25 (X'19') is used by a program other than OLTEP, an HDV instruction is issued to the device provided that it is a teleprocessing device, it is not a BTAM controlled 270x device, and an I/O interrupt is pending for this device. If the specified device is not a teleprocessing device or if no interrupt is pending, this routine directly returns to the issuing program.
In case the SVC 25 (X'19') is for a BTAM-controlled 270x device the SVC 25 (X'19') causes the system translated channel program (polling loop) to be modified such that the polling loop is discontinued.

If OLTEP is the issuing program, a HDV instruction is issued to the device provided that an I/O interrupt is pending for the device.
In case the I/O operation has not yet been initiated or was already completed, due to device sharing, SVC 25 (X'19') just forces the channel queue entry to be dequeued from the channel queue.

SVC 26 (X'1A')

Validate address limits. The program issuing an SVC 26 (X'1A') is canceled if the PSW protection key is not zero.
The lower address must be specified in general register 1, and the upper address must be specified in general register 2.

If either address is outside the requester's partition, the task is canceled.

SVC 27 (X'1B')

Provides exactly the same service as SVC 25 (X'19') does, except that SVC 27 (X'1B') will ensure that due to a HDV instruction a CHANQ entry will *not* be dequeued from the channel queue.

SVC 28 (X'1C')

Reserved

SVC 29 (X'1D' - WAITM)

Provides supervisor support for the WAITM macro. On entry, general register 1 contains the address of an ECB list. The ECBs are all checked for the traffic bit (byte 2 bit 0). When an ECB is found with the traffic bit already posted, the SVC 29 (X'1D') routine returns with the address of the posted ECB in general register 1. If none of the specified ECBs has the traffic bit yet posted, the task is set "I/O-BOUND" and it's PSW is modified, so that the SVC will be reissued when the task is selected again (RESVC).

SVC 30 (X'1E')

Allows privileged programs to submit console input to VSE. The input is processed as if it had been entered by a console that controls the job, from which the SVC 30 was issued. This is either any master console, or a specific console when an ECHOU option is effective for that job.

General register 0 must be set to zero and general register 1 must contain the address of a 6-byte parameter list. The first two bytes (byte 0 and 1) of this parameter list must contain the length (number of bytes) of the command which is to be submitted and byte 2 through 5 must contain the address of the users area which does contain the command itself. The length of the command must be in the range from 1 to 72. On return from the SVC, general register 15 will contain a return code.

Return codes:

Reg.15 X'00' Command submitted successfully
X'04' Interface is busy or not yet operational, retry later
X'08' Input is accepted, but truncated to 72 bytes
X'0C' Input is rejected
X'10' Invalid parameter list

SVC 31 (X'1F' - REIPL)

This supervisor call provides supervisor service to perform software Re-IPL. Input registers are register 0 and register 1. Register 0 contains a function code describing the IPL request. Register 1 points to an eight byte parameter list.

SVC 32 (X'20' - WTO/WTOR)

Builds console buffer entries for the VSE/ESA Version 1 level of the WTO/WTOR macros. The pointer to the WTO/WTOR parameter list is supplied in general register 1.

For more details on WTO/WTOR processing refer to the Console Functions DRM.

SVC 33 (X'21' - COMRG)

The COMRG macro (SVC 33 - X'21') returns the address of the partition communication region.
It is often used to immediately enter the task selection.

SVC 34 (X'22' - GETIME)

Provides the supervisor service for the GETIME macro; updates the date field in the communication region of the issuing partition. Upon return, general register 1 contains the time of day in timer units (1/300 sec.)

SVC 35 (X'23')

Protects a track (or block if an FBA device) from use by more than one task at a time. A task requesting a held track must wait until the track is free. If more than sixteen holds on a track are attempted, the requesting task is canceled.

Exits are to execute the I/O operation, or to repeat the SVC (label RESVC) if the track is already held. At RESVC, the program old PSW is set to execute the SVC 35 (X'23') again, and a branch is taken to task selection.

SVC 36 (X'24' - FREE)

FREEs a track (or block if an FBA device) that is held by the task issuing the FREE macro. An attempt to free a track not owned by the requester causes the issuing task to be canceled.

SVC 37 (X'25' - STXIT AB)

SVC 37 (X'25') provides supervisor support for the STXIT AB macro. It establishes linkage from the supervisor to a Abnormal Termination Exit routine in a problem program. It stores the address of the Exit routine, the caller's PSW key and the address of the save area in the AB-exit control block (anchor in TCBABEX in the tasks TCB).

Notes:

1. The format of the user save area is shown in MACRO MAPSAVAR (refer to *z/VSE System Macros Reference*, SC33-8230).
2. After the invocation of an AB exit routine, register 0 contains the abnormal termination code and register 1 the pointer to the AB save area.
3. The save area address will be validated (ERR25) before both addresses and the caller's PSW key are saved in the AB-exit control block.

Subsystem Support via OPTION=EARLY:
OPTION=EARLY indicates that the AB exit routine has to be invoked for any type of termination (normal and abnormal) and, for a maintask, before propagating the termination to its subtasks.

Restrictions:

- An EARLY exit can only be set once during the whole lifetime of a task.
- Reset is not allowed.
- OPTION=EARLY is supported only for subsystems, depending on their identification via the SUBSID macro.

If the user violates these rules, the setting will be ignored and a return code will be put in register 15:

0 (X'00')	Exit successfully set.
4 (X'04')	Exit already set.
8 (X'08')	Reset not allowed.
12 (X'0C')	No subsystem request.

If a second AB exit routine with an OPTION other than EARLY is specified, the related AB-exit control block is anchored in the AB-exit control block of the EARLY-AB exit control block. In this case, only the EARLY AB exit will be invoked for any kind of termination.

The maintask and subtasks may have the same or different AB routines. When a subtask is attached after an STXIT AB macro has been issued by the maintask, the subtask gets the AB routine address (if an AB-exit is specified with an OPTION other than EARLY) specified by the maintask only if the ATTACH macro for that subtask has the ABSAVE parameter specified. The subtask can override this by issuing its own STXIT AB macro.

SVC 38 (X'26' - ATTACH)

A subtask is to be established via the ATTACH macro. If no further subtask is available, if the maximum number of subtasks attachable to a partition is exhausted or if no SVA space for task control blocks is available, supervisor will supply an ECB pointer, indicated by the high order bit set to 1 which is stored into the user's R1 before control is returned to the user. The user may now issue a WAIT for this ECB.

If a set of task information, task control blocks and access register save areas is available (otherwise SVA space will be allocated), a subtask is attached by initializing the control blocks and by inserting the task ID and status byte into the Task Identifier String (TIDSTR) and Task Selection String (TSS), respectively. The newly attached subtask gets a processing priority just above that of the main task.

The issuing task's save area is copied to the subtask's save area. The subtask is set to 'ready-to-run'. Bit 0 of the issuing task's register 1 is set to 0 to indicate a successful attach. Control is then returned to task selection.

SVC 39 (X'27' - DETACH)

Performs normal termination of a subtask. DETACH may be issued by the subtask being terminated, by the maintask, or by the task which attached this subtask. If DETACH is issued by a problem program, the cancel code 16 (X'10' - normal End-Of-Job) is set in the subtask's TIB and the Terminator is entered. At the end of the termination process, DETACH is issued by the EOJ SVA-resident routine setting the subtask inactive and posting its ECB for termination (for layout of ECB see z/VSE Supervisor Diagnosis Reference, Appendix A). The subtask status byte and identifier are removed from the Task Identifier String (TIDSTR) and Task Selection

String (TSS). The task is freed; any waiting attach requests and associated supervisor ECBs are posted.

SVC 40 (X'28' - POST)

Used for inter-task communication. POST may be issued by either a maintask or a subtask. It is issued so that a task is aware of the termination of an event. Normal completion of the specified event is posted in the ECB (byte 2, bit 0 = 1). If the SAVE=parameter is present, only the task, owning the save area and waiting for this ECB, is taken out of the wait state; otherwise, all tasks waiting for this ECB are removed from the wait state.

Only tasks running in the issuing partition are affected.

SVC 41 (X'29' - DEQ)

Informs the system that a resource (shared data area) is now available for use by another task. A task may issue the DEQ macro only to a resource that it currently owns. If it attempts to issue the DEQ macro to some other resource, the task is canceled.

If any other tasks are waiting for the resource, the highest priority task ready to run is removed from the wait state and gains control. If no other task is waiting for the resource, control returns to the task that issued the DEQ macro.

If a task terminates without dequeueing all of its enqueued resources, either in its normal coding or in its abnormal termination exit routine, any task subsequently attempting to enqueue the resources is canceled.

SVC 42 (X'2A' - ENQ)

ENQ prevents tasks from simultaneous manipulation of a resource (shared data area). This is accomplished by setting all bits of byte 0 of the specified Resource Control Block (RCB) to 1. Then the Event Control Block (ECB) address is placed in bytes 4 through 7 of the RCB. A task attempting to enqueue a resource that is already enqueued by another task is placed in a queue and put in a waiting condition. The old PSW is set to re-execute the SVC 42 (X'2A') and task selection is performed.

A task is canceled if it attempts to nest ENQ(s) of a resource or if it attempts to ENQ a resource that is still owned by a terminated task.

When a task is finished with a resource, it should inform the system by issuing the DEQ macro. Tasks subsequently requesting the resource are canceled, if the tasks owning that resource have been terminated in between.

SVC 43 (X'2B' - Tasking Services)

The tasking services (DYNCLASS) allow to manipulate or retrieve dynamic partition table information and to call service routines, e.g. for preparation and clean-up.

DYNCLASS ID=	Description	Authorization
CLEANUP	process Dynamic Partition clean up	VSE/AF JC
CHECK	validates a given Dynamic Class	—
DISABLE	disable the given Dynamic Class	VSE/POWER
DEFAULT	retrieve system defaults	—
ENABLE	enable one or all Dynamic Classes	VSE/POWER
FREE	free a partition that was held	key 0
GET	get one or all Dynamic Class entries from library	VSE/POWER, II
HOLD	hold a partition during clean up	key 0
INITIAL	initialize system	IPL
JACCT	retrieve partition/class job account. information and characteristics	—
LOAD	set given Dynamic Class Table active	VSE/POWER
PREPARE	process Dynamic partition preparation	VSE/AF JC
RETURN	return from IJBDCTL phase	IJBDCTL
SAVE	store given Dynamic Class Table into VSE/AF Library	II

Figure 28. Available Services

Register 15 contains the function code.

The services DYNCLASS ID=GET and ID=SAVE will call the SVA routine **IJBDCTL**, where the routine's address is located in the SVASVDL (supervisor sub-directory), when loaded into the SVA.

IJBDCTL has its own save area. The DYNCLASS SVC code stores the input registers (0, 1, 15) into the save area and updates the PSW with the IJBDCTL start address. This address is also moved into save area's register 12. The IJBDCTL routine will be called via the dispatcher. IJBDCTL returns to the DYNCLASS SVC code with DYNCLASS ID=RETURN.

The start address of IJBDCTL can be obtained as follows:

```
IJBDCTL address in SVASVDL + length of save area
```

Save area's register 2 will contain the base register for internal service calls.

SVC 44 (X'2C')

Provides the supervisor service to write a SD record (statistical data) onto the recorder file.

General register 1 contains the address of the SD record that the user wants to be recorded onto the recorder file. The specified area is first validated (ERR25) and subsequently all virtual pages containing the statistical data are TFIXed, if they are not already PFIXed.

If the error entry in the ERBLOC area is not available, all pages just TFIXed are TFREEd and the requesting task is set "ERQBND (X'62')" until the ERP task completes its current processing. If the ERROR ENTRY is available, the information passed by the user is saved into the error entry (refer to ALTERNATE ERROR ENTRY description) and the phase name of the first physical transient to be fetched is set ($$ABERA6). The transient ERP is activated to asynchronously do the recording.

SVC 45 (X'2D')
Reserved.

SVC 46 (X'2E')
This SVC can be used by OLTEP only and allows OLTEP to operate in supervisor state. General register 1 contains the address of an OLTEP appendage routine that is immediately to be entered via a BALR interface and which will return via the link register.

SVC 47 (X'2F')
Reserved.

SVC 48 (X'30')
Reserved.

SVC 49 (X'31')
Provides I/O services for ACF/VTAM only. Any other user will be canceled (ERR21) when using SVC X'31'. This SVC makes all the provisions necessary to HALT an ongoing teleprocessing I/O operation, or, to START a teleprocessing I/o operation. In case the device is to be halted, this service routine directly returns to task selection. In case a new I/O operation is to be started, this routine passes control to the supervisor Start I/O routine.

SVC 50 (X'32')
This SVC forces the issuing task to be canceled ERR09 and is intended to be used by LIOCS (LOGICAL INPUT/OUTPUT CONTROL SYSTEM) for error diagnostics.

SVC 51 (X'33' - HIPROG)
Provides the ability to determine the length of a phase without loading it. On entry to the SVC 51 (X'33') routine, register 1 must point to the storage address of the phase name. This 8-byte area must be followed by an area large enough to hold further directory entry information. The length of the area must be specified in byte 3 of the area itself as the number of halfwords. The rest of the area must be set to X'00'. The FETCH/LOAD service returns the selected part of the directory entry. The area is not altered if no directory entry for the particular phase has been found.

FETCH scans the SDL, the IJSYSRS.SYSLIB sublibrary (SYSLIB) and the active chain of private sublibraries (PSUBLIB) for the directory entry of the phase.

If job control provides a parameter list and sets the flag option to X'01' or X'02', the HIPROG value for this partition will be calculated.

In case of X'01', only the length of the specified phase is considered.

In case of X'02', the HIPROG value is the address of the uppermost byte of the phase with the highest ending address for the corresponding partition. All phases starting with the same four characters as the phase name given on the EXEC statement are considered.

The calculated value is stored in bytes 40 through 43 of the partition communication region. If the phase searched for is in the SVA, the partition start address plus page size will be used.

SVC 52 (X'34' - TTIMER)

Provides the TTIMER macro support.
The remaining time interval (in 1/100 seconds) to elapse before the clock comparator interrupt occurs is returned as an unsigned 32-bit binary number in general register 0.
All zeros are returned if no timer interval was set by the task issuing the TTIMER macro.

If the task issuing the TTIMER macro has an entry in the IT request (ITREQ) chain, the value returned is the difference between the values of the Expiration Time in the IT-exit control block (addressed through TCBXTAVS) and the TOD clock.

If TTIMER CANCEL is specified, and the task owns an IT-exit control block with an Expiration Time value other than 0, then that entry is deleted (i.e Expiration Time is set to zero).

SVC 53 (X'35')

Used by ACF/VTAM (and ACF/VTAME) to perform a number of functions, such as enqueuing or dequeueing ACF/VTAM resources or posting ACF/VTAM ECBs. After an SVC 53 (X'35') is issued, the supervisor passes control to ACF/VTAM to perform the required function. Control is passed back to the calling module via the supervisor.

SVC 54 (X'36')

SVC 54 (X'36') provides the supervisor support for the FREEREAL function to release page frames to the page pool. These page frames may be released from the partition's REAL address area, or the SDAID area. The task issuing this SVC is canceled (ERR21), if it does not run with a protection key of zero and the phase name is other than SDAID, $$BATTN or $$BVSEPT.

A zero value in general register 2 indicates that the request is issued by SDAID. In this case, the lower and upper limit of the area to be released are obtained from the internal page manager address fields. Control is passed immediately to task selection if no SDAID area exists.

The page frames are freed, one after the other, by updating the corresponding Page Frame Table Entries (PFTEs). The partition PFIX counter in the Storage Management Control Block (SMCB) part of the Partition Control Block (PCB) is decremented by one for each page that is freed.

The released page frames are enqueued at the beginning of the invalid page frame queue.

The SVC 54 (X'36') posts as "READY TO RUN" all tasks waiting for page frames, if more than the minimum number of page frames is available in the Page Selection Queue (PSQ).

SVC 55 (X'37')

SVC 55 (X'37') provides supervisor support for the GETREAL function to request pages from the page pool for the SDAID area. Control is passed immediately to task selection if such a request is already in progress or if the SDAID area already exists. If the requester does not have protection key zero, and is neither identified as SDAID nor as $$BATTN, the issuing task is canceled (ERR21).

The number of requested page frames is passed in register 0. This value is replaced by the number of page frames that are available for GETREAL if this number is less than the requested number. After handling the request, the number of page frames taken from the main page pool and the address of the SDAID area are passed to the user in registers 0 and 1. Register 0 contains zero and register 1 remains unchanged if no page frames are available.

The SVC 55 (X'37') routine passes the begin and end addresses of the requested SDAID area as parameters to the GETREAL routine.
The begin and end address of the SDAID area are saved in the internal page manager address fields (AAAADR,AAAEND).

SVC 56 (X'38' - CPCLOSE)

Issued by VSE/POWER to close printer or punch files written by VSE/POWER and spooled by VM or VM/System Product at the End-of-Job. VSE/POWER passes a parameter list to the SVC. This list contains the HEX address of the device to be closed in low order half of the first word and the device address in EBCDIC in the second word and the job name in the third and fourth words.

SVC 57 (X'39' - GETPRTY,SETPRTY)

Allows to display and/or change the partition priorities.

GETPRTY request: Can be issued by any task. The following parameters are passed to this routine:

Register 0	Contains the length for TYPE=STATIC and the pointer to a parameter list for TYPE=ALL.
Register 1	Contains the area address for TYPE=STATIC and the pointer to a parameter list for TYPE=ALL.

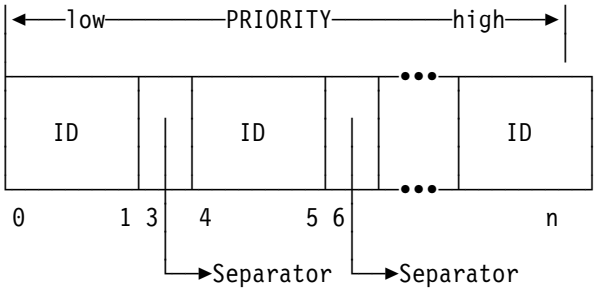
The priority list (see below) is moved to the user-supplied area.

SETPRTY request: Issued by the Attention Task following an attention PRTY command given by the system operator. Only one parameter is passed to this routine:

Register 0 address of the area containing the priority information (see below) that the system operator wants to be established.

The SVC 57 (X'39') routine scans the priority list.

The format of the priority list as it has to be made available to the SVC 57 (X'39') routine is as follows:




```
ID = BG,F1,...,FB or dynamic class character followed by X'FF'.
n  = number of static partitions + number of dynamic classes.
Separator    = comma(,) or equal sign(=)
```

Figure 29. Format of Priority List

SVC 58 (X'3A' - INVPART)

Issued by job control or end of task to initialize a partition. The task issuing this SVC is canceled if it does not run with a protection key of 0. See also the description of the INVPART macro.

R2 is the only parameter that is passed to this routine (if called by job control)

R2:

zero	Next program in the partition is to run in REAL mode.
one	Next program in the partition is to run in virtual mode.

If the routine is called by end of task the next job step will run in virtual mode. The handling is determined by the PIBTRAM flag:

- PIBTRAM = ON Ended job step ran in virtual mode
- PIBTRAM = OFF Ended job step ran in real mode

If the partition is running in virtual mode and its first page has to be invalidated, the first page is cleared except for the first 200 bytes (save area).

If the next program is to run in REAL mode, the translation mode bit in PIBFLAG0 in PIB is reset, and the entry in SYSCOM indicating the number of the active partitions running in virtual mode is decreased by one. All copy blocks used by the fast CCW translation routines (REPLICA, CCB, CCW, etc.) are released, and the pages which were FIXed by these blocks are freed.

The pages are invalidated in the range SMVPBEG, SMVPEND-1.
The invalidation is done using the INVPAGE service of page management.

If the partition is to run in REAL mode, the following actions are taken, in addition to the general actions described above:

- All page table entries belonging to the partition running in REAL mode are initialized.
- The page frame table entries (PFTEs) that correspond to the partition running in REAL mode are initialized and removed from the page selection queue (PSQ). The partition PFIX counter in the storage management control block (SMCB) and the counter for the number of page frames in the PSQ is updated.
- The storage protection key of the page frames of the partition running in REAL mode is set equal to the PIK of the partition.
- The first 200 bytes of the partition running in virtual mode are moved to the first 200 bytes of the partition running in REAL mode. The appropriate entry in the PIB and the SMCB are updated to point to the save area of the partition running in REAL mode instead of pointing to the save area of the partition running in virtual mode.
- The first page of the partition running in virtual mode is invalidated.

Page frames required for the partition which is to run in REAL mode are reserved by the GETREAL routine (refer to z/VSE Supervisor Diagnosis Reference, Chapter "GETREAL Request" which belongs to Page Management). The number of page frames to be reserved depends on the size of the real partition.

If, during execution of the GETREAL, a page frame in the partition running in REAL mode is found to be failing, the job that is to be initialized is canceled (ERR2D). The save area is not moved to the partition running in REAL mode, the save area pointer in the TCB remains unchanged, and virtual mode is posted in the PIB.

SVC 59 (X'3B')

Reserved

SVC 60 (X'3C' - GETDADR)

Calculates from the real address the virtual address of a location within the data area of an I/O request.

Before this SVC is issued, general register 8 must contain the address of the CCW. General register 0 must contain the displacement of the desired address from the start of the I/O area. Using the data address or the address of the indirect addressing list (IDAL) specified in the CCW, the supervisor calculates the virtual address and returns it in general register 15.

If the real address is invalid (in an unused page frame or beyond the end of processor storage), all zeros are returned.

SVC 61 (X'3D' - GETVIS)

Provides the supervisor support for the GETVIS macro. It reserves part of the GETVIS area which may either be part of a partition, part of a dynamic space or part of the shared virtual area (SVA).

On successful completion of the operation, X'00' is returned in general register 15, the start address of the reserved area returned is in general register 1. The length of the area, which must be specified by the user, is contained in general register 0.

SVC 62 (X'3E' - FREEVIS)

Provides the supervisor support for the FREEVIS macro. It releases a block of virtual storage. The start or subpool name address of the area to be released is contained in register 1. The length of the area to be released is in register 0.

If the return code in register 15 is not zero, no action was taken by FREEVIS.

SVC 63 (X'3F' - USE)

Allows supervisor controlled access to a system resource as requested by the internal macro USE. SVC 63 (X'3F') requests are converted by the LOCK manager to LOCK requests.

SVC 64 (X'40' - RELEASE)

Provides the supervisor service for the RELEASE macro. A resource previously locked by means of the USE macro (SVC 63 - X'3F') is now to be RELEASEd. SVC 64 (X'40') requests are converted by the LOCK manager to UNLOCK requests.

SVC 65 (X'41' - CDLOAD/CDDELETE)

The function code in register 15 indicates which macro was issued.

CDLOAD macro

Loads a phase dynamically into the partition GETVIS area when called by the macro CDLOAD.
In case the phase is found in the SVA and the requesting program is not running in REAL mode, the phase is not loaded, but the SVA load address is returned to the caller.

CDDELETE macro

Deletes a phase previously loaded in the partition GETVIS area by means of the CDLOAD macro.

SVC 66 (X'42' - RUNMODE)

Provides the supervisor service for the RUNMODE macro which returns the mode in which a partition is running. It returns 0 in register 1 if the program is running in virtual mode and returns 4 in register 1 if the program is running in REAL mode.

SVC 67 (X'43' - PFIX, SPLEVEL = 1)

Page management PFIX service for VSE/ESA releases up to 1.2 (SPLEVEL = 1). General register 1 points to the list of pages to be fixed. General register 15 is used for return code only. For more detail refer to SCV 121 function code=1 (see “SVC 121 (X'79'- Page Management Services)” on page 294).

SVC 68 (X'44' - PFREE, SPLEVEL = 1)

Page management PFREE service for VSE/ESA releases up to 1.2 (SPLEVEL = 1). General register 1 points to the list of pages to be freed. General register 15 is used for return code only. For more detail refer to SCV 121 function code=2 (see “SVC 121 (X'79'- Page Management Services)” on page 294).

SVC 69 (X'45' - REALAD)

Returns the real address for the virtual address specified in the REALAD macro. On entry to the SVC 69 (X'45') routine, the virtual address must be contained in register 1. The real address is returned in register 0.

No address is returned if:

- The virtual address in register 1 is invalid.
- The address is within a page which is not PFIXed.

SVC 70 (X'46' - VIRTAD)

Returns the virtual address for the real address specified in the VIRTAD macro. On entry to the routine, the real address must be contained in register 1, and register 0 must contain zero. The virtual address is returned in register 0.
No address is returned (register 0 contains zero) if:

- The real address refers a page frame that is not used.
- The real address is invalid.
- The virtual address is within a page which is not fixed.

SVC 71 (X'47' - SETPFA)

Page management SETPFA service if AREA parameter is not specified. (macro expansion is downward compatible).

The SVC routine is entered with general register 0 containing the address of the page fault appendage (PHO) routine if PHO processing should be enabled or zero if PHO should be disabled.

SVC 71 (X'47') establishes linkage between the supervisor and the user-written page fault appendage routine in the page fault appendage TIB located in the PCB (PHOTIB). Only one task of a partition may have an active PHO-appendage.

SVC 72 (X'48' - GETCBUF)

Gets or releases a copy block used for channel program translation. The program issuing a SVC 72 (X'48') is canceled if it is not the ERP system task.

If a request for copy blocks is made, the chain of free copy blocks is searched. If the chain is not empty, a copy block is dequeued from the chain, and the address of the copy block is passed to the ERP system task. Otherwise, a value of zero is returned. A copy block is released by enqueueing it to the chain of free blocks. Any tasks waiting for copy blocks are then posted.

SVC 73 (X'49' - SETAPP)

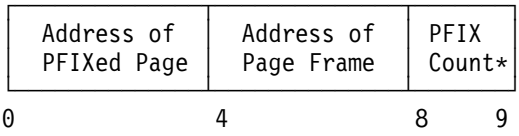
Authorizes linkage to a channel end appendage routine for authorized programs.

SVC 74 (X'4A' - PFIXCHPT/PFIXREST)

Builds a parameter list during checkpointing or when a restart occurs, fixes pages in accordance with the parameter list, and stores the correct values in the PFIX counter located in the page frame table entry and the partition PFIX counter located in the SMCB.

The PFIX counter indicates how often a page is PFIXed, the partition PFIX counter keeps tracks of how many pages of a partition have been PFIXed. If checkpointing

is requested, a parameter list is built with an entry for each PFIXed page of an affected program. The format of the parameter list is shown below.




```
* indicates how often the page is fixed
```

Figure 30. Restart-PFIX Parameter List Entry

Only tasks running with protection key 0 may issue an SVC 74 (X'4A'). Register 0 contains the length of the parameter list, and register 1 points to the parameter list. On return, register 2 contains zero, if no additional parameter list is needed, and four, if an additional parameter list is needed. A non-zero byte is placed right after the last generated entry of a parameter list.

If the restart function is requested, register 0 contains zero. Register 1 points to the parameter list built by the checkpoint function. The pages identified by the parameter list are PFIXed by calling the PFIX routine (see “SVC 67 (X'43' - PFIX, SPLEVEL = 1)” on page 267) for each page. After the PFIXing of a page, the PFIX counter is set as indicated in the parameter list, and the partition PFIX counter is increased by 1.

Since a page has to be fixed in the same page frame in which it was fixed at checkpoint time, the address of the page frame table entry belonging to the page frame in which the page should be fixed is saved in the PTFERSVD of the PCB before control is transferred to the PFIX routine.

SVC 75 (X'4B' - SECTVAL)

Calculates the sector value for a position on a track of a DASD device supporting rotational position sensing (RPS). If the supervisor was generated without RPS specified in the FOPT macro, the issuing program is canceled (ERR21) if the device in question is not an ECKD device. For ECKD devices sector calculation is always supported. The routine calculates the position for either fixed or variable length records. For a detailed description see *z/VSE System Macros Reference*, SC33-8230.

On return to the caller R0 contains the calculated sector value.

SVC 76 (X'4C')

Initiates the recording of an RMSR record on the Recorder File (SYSREC). If the system runs under VM, not all information in the record may be valid. VM gains control to perform the recording function. When not running under VM, the effect of this SVC is the same as for SVC 15 (X'0F' - SYSIO).

The address of the user's CCB must be supplied in general register 1 before this SVC is issued. The data address must be supplied in general register 0. If the recorder file is on a CKD device, register 1 must have the high-order bit on to indicate that VM must intercept this SVC. After having intercepted, VM zeros out this register so that, on return, the issuing program can check whether VM handled the I/O request.

If the recorder file is on an FBA device, the interrupt is not intercepted by VM.

SVC 77 (X'4D' - TRANSCSW)

Only system tasks are allowed to use this SVC. It is used in routines which print the CCW address of a failing I/O operation, such as the ERP message writer. The virtual address of a copied CCW is calculated.

On entry to the SVC 77 (X'4D') routine, register 0 contains the address of the copied CCW, and register 1 the address of the copied CCB.

The retranslated CCW address is returned in register 0 if the address passed in this register by the user points to a copied CCW related to the I/O operation being handled. If the user passed an invalid address, register 0 contains the value zero on return. The contents of register 1 are not changed.

SVC 78 (X'4E' - CHAP)

Provides support for the CHAP macro. The priority of the issuing subtask is made the lowest priority of all subtasks in that partition by modifying the TIDSTR and TSS fields (old dispatcher) and changing the task priority queues (TD dispatcher). The TID of a subtask is inserted immediately before the identifier of the main task. The identifiers of the subtasks with lower priority, are moved one byte higher. The use of this SVC by the main task is ignored.

SVC 79 (X'4F')

The SVC contains functions that are allowed in x-mode, e.g.

- return from ESTAEX-type exits.

SVC 80 (X'50')

Reserved.

SVC 81 (X'51')

Reserved.

SVC 82 (X'52')

This function supports the setting and resetting of monitor calls class 4 for VSE/ICCF, or transfers control immediately to that program for having services executed by VSE/ICCF.

SVC 83 (X'53' - ALLOCATE)

Allocates or reallocates real or virtual partitions. It is either called by the ALLOCATE macro to allocate a static partition or directly by VSE/POWER to allocate a dynamic partition.
Register 1 contains the address of the allocation parameter list when called by the ALLOCATE macro or the input parameters when called by VSE/POWER.
Register 15 contains a return code after completion. See also the description of the ALLOCATE macro. The allocation or reallocation process involves:

- Calling the page manager invalidation routine to flag the contents of the pages as invalid.
- Updating the partition limits in the appropriate entry of the Storage Management Control Block (SMCB). The system limits are taken from the control block SMCOM . For the layout of the SMCOM see z/VSE Supervisor Diagnosis Reference, Chapter "Storage Management".

SVC 84 (X'54' - SETLIMIT)

Changes partition sizes or sets the PFIX limits for a partition. For a description of the SETLIMIT macro, including a description of information that is passed to and returned by the supervisor see the SETLIMIT macro description.

For further information see z/VSE Supervisor Diagnosis Reference, Chapter "Storage Management".

SVC 85 (X'55' - RELPAG, SPLEVEL = 1)

Page management RELPAG service for VSE/ESA releases up to 1.2 (SPLEVEL = 1). General register 1 points to the list of pages to be released. General register 15 is used for return code only. For more detail refer to SVC 121 function code=3.

SVC 86 (X'56' - FCEPGOUT, SPLEVEL = 1)

Page management FCEPGOUT service for VSE/ESA releases up to 1.2 (SPLEVEL = 1). General register 1 points to the list of pages to be paged-out. General register 15 is used for return code only. For more detail refer to SCV 121 function code=4 (see “SVC 121 (X'79'- Page Management Services)” on page 294).

SVC 87 (X'57' - PAGEIN, SPLEVEL = 1)

Page management PAGEIN service for VSE/ESA releases up to 1.2 (SPLEVEL = 1). General register 1 points to the list of pages to be paged-in. General register 15 is used for return code only. For more detail refer to SCV 121 function code=5 (see “SVC 121 (X'79'- Page Management Services)” on page 294).

SVC 88 (X'58' - TPIN)

Provides the support for the TPIN macro. SVC 88 (X'58') is ignored and control is returned to the user in a system without Page-Data-Set. This SVC should always be used in combination with SVC 89 (X'59' - TPOUT). Both SVCs are intended for exclusive use by teleprocessing access methods such as ACF/VTAM and by data base/data communication interface programs such as CICS/VS. The SVCs are required for the supervisor to perform TP-Balancing.

SVC 88 (X'58') indicates to the supervisor that an immediate demand is to be made on system resources by the subsystem issuing the SVC 88 (X'58'). The SVC requests the supervisor to deactivate one or more static partitions or dynamic classes of lowest processing priority.

The demand is ignored in each of the following cases:

- The user has not requested TP Balancing via the TPBAL command.
- None of the partitions specified in the TPBAL command can be deactivated (no tasks running in virtual mode other than TPIN-issuing task).
- There are no page faults in the system.

Forced deactivation is obtained by indicating to the page manager that no page fault should be handled for the partitions. At the same time, reactivation and deactivation is prohibited by setting the TP-in-progress bit and the batch-deactivated bit in the SUPFLAG byte.

SVC 89 (X'59' - TPOUT)

Provides the support for the TPOUT macro. SVC 89 (X'59') is ignored and control is returned to the user in a system without Page-Data-Set. This SVC is the necessary counterpart of SVC 88 (X'58' - TPIN), it resets the TP-in-progress bit in the SUPFLAG byte and permits normal reactivation and deactivation of static partitions and dynamic classes. All partitions/classes deactivated by the previous TPIN request are reactivated immediately.

SVC 90 (X'5A' - PUTACCT)

Provides support for the PUTACCT macro. If VSE/POWER provides account support, that program's account appendage is entered. Otherwise, the user ECB is posted (byte 2, bit 0).

SVC 91 (X'5B')

Provides interface between job control and VSE/POWER. On entry of the SVC 91 (X'5B') routine, the VSE/POWER account appendage is given control.

SVC 92 (X'5C' - XECBTAB)

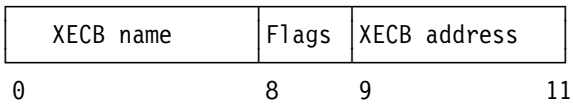
Provides supervisor support for the XECBTAB macro.
SVC 92 (X'5C') adds, removes, checks, or resets one entry in the cross-partition event control block (XECB) table or deletes all entries belonging to the issuing task. Figure 31 shows the format of an entry in the XECB table.

Bytes	Description
0 – 7	XECB name
8	ACCESS Control byte X'80' Table entry in use 40 Task that issued XPOST was canceled 20 Reserved 10 Reserved XWAIT access indicator 08 ACCESS=XPOST specified 04 ACCESS=XWAIT specified XPOST access indicator 02 ACCESS=XWAIT specified 01 ACCESS=XPOST specified
9 – 11	XECB address
12 – 13	TID of owner
14 – 15	TID of first task that posted XWAIT or XPOST for XECB
16 – 19	Forward chain pointer
20 – 23	Backward chain pointer

Figure 31. Format of XECB Table Entry

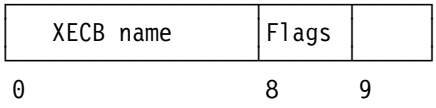
If bits 4 and 5 of byte 8 are set to 10, bytes 14 and 15 contain the TID of the first task that issued an XWAIT for this XECB. If bits 6 and 7 of byte 8 are set to 10, bytes 14 and 15 contain the TID of the first task that issued an XPOST for this XECB. Label XECBTAB identifies the first byte of the table.

Whenever SVC 92 (X'5C') is invoked, register 1 must point to a user parameter list describing the XECB. The length of the parameter list is 12 bytes for TYPE=DEFINE and 10 bytes for the other type options.



Bytes	Description
0 – 7	XECB name
8	Flag byte
	X'80' Reserved
	40 Reserved
	20 Reserved
	10 Reserved
	XWAIT access indicator
	08 ACCESS=XPOST specified
	04 ACCESS=XWAIT specified
	XPOST access indicator
	02 ACCESS=XWAIT specified
	01 ACCESS=XPOST specified
9 – 11	XECB address

Figure 32. Parameter List for TYPE=DEFINE



Bytes	Description
0 – 7	XECB name
8	Flag byte
	X'80' TYPE=CHECK
	40 TYPE=DELETE
	20 Reserved
	10 TYPE=RESET
	08 DELETALL Request
	04 Reserved
	02 Reserved
	01 Reserved
9	Reserved

Figure 33. Parameter List for TYPE=DELETE, DELETALL, RESET or CHECK

Actions for TYPE=DEFINE

The XECB table is scanned for a possible duplicate entry. If none is found, a new entry is created by means of SGETVIS and the user parameters are moved to it.

On successful completion of the operation, the XECB address is returned in general register 1, and the address of the table entry being handled is returned in general register 14.

One of the following codes is returned in register 15 after an XECBTAB TYPE=DEFINE has been issued:

```
X'00'  DEFINE function completed
       successfully.
X'04'  XECB already defined in the
       XECB table.
X'08'  No GETVIS storage available.
```

Actions for TYPE=RESET

The table is scanned for the specified XECB name. If the name is found, a check is made to see whether the issuing task owns the XECB. If it does, the waiting task (if any) is posted ready-to-run, the XECB table entry is cleared and is free now for use by another communicator. On successful completion of the operation, registers 1 and 14 are set to zero.

One of the following codes is returned in register 15 after an XECBTAB TYPE=RESET has been issued:

```
X'00'  RESET function completed
       successfully.
X'04'  XECB not found in the XECB table.
X'08'  Issuing task does not own the XECB.
```

Actions for TYPE=DELETE

The table is scanned for the XECB name specified. If the name is found, a check is made to see if the issuing task owns the XECB. If it does, the waiting task (if any) is posted ready-to-run, the XECB table entry is removed from the XECB TABLE and SFREEVIS is issued. On successful completion of the operation, registers 1 and 14 are set to zero.

One of the following codes is returned in register 15 after an XECBTAB TYPE=DELETE has been issued:

```
X'00' DELETE function completed
      successfully.
X'04' XECB not found in the XECB table.
X'08' Issuing task does not own the XECB.
```

Actions for TYPE=CHECK

The table is scanned for the XECB name specified. Depending on the result of the scan, one of the following codes is returned in general register 15:

```
X'00'  XECB name found in the table.
       General register 1 contains the
       address of the XECB; general
       register 14 contains the
       address of the table entry.
X'04'  XECB name not found in table.
       General registers 1 and 14 are
       set to zero.
```

Actions for TYPE=DELETALL

The XECB table is scanned for all entries which are owned by the issuing task or which communicate with the issuing task.

Any XECB table entry which is owned by the issuing task is deleted from the XECB table and SFREEVIS is issued. The waiting task (if any) is posted ready-to-run.

An XECB table entry which contains the issuing task's ID in its communication field has the ID field (bytes 12 and 13) cleared to zero. When the owning task is a waiting task (entry defined with ACCESS=XWAIT), it is posted ready-to-run, and an abnormal termination flag is set in the XECB table entry and in the XECB (byte 2, bit 1).

On return from SVC X'5C' (with TYPE=DELETALL) the contents of the general registers 14 and 15 are not changed.

A common subroutine, FNDLOOP, is used to scan the XECB table for a specified XECB name.

SVC 93 (X'5D' - XPOST)

Provides supervisor support for the XPOST macro. The routine posts a specified XECB and marks the task (if any) waiting for this XECB as ready.

Every time SVC 93 (X'5D') is invoked, register 1 must point to a field in the issuing partition that contains the XECB name;
register 14 must contain the table entry address. The XECB table (see Figure 31 on page 272) is scanned for the specified name by the subroutine FNDLOOP. If the entry is found, a check is made to see if the task is authorized to XPOST this XECB, as indicated in the table entry. Subroutine XECCHK is used to make this check.

If the task is found to be authorized, the traffic bit in the XECB is set and the entry is examined to see if a task is waiting on this XECB. If there is a task, it is posted ready-to-run again.

Return codes in register 15:

0 (X'00')	Successful completion.
4 (X'04')	XECB name not found in XECB table.
13 (X'0D')	Task not authorized to issue XPOST.
and	The return code is made up of
14 (X'0E')	12 (X'0C') plus, in the rightmost two bits, the XPOST access code that was contained in the table entry (see SVC 92 (X'5C') “SVC 92 (X'5C' - XECBTAB)” on page 272 for an explanation of the access code).

SVC 94 (X'5E' - XWAIT)

Provides supervisor support for the XWAIT macro. SVC X'5E' checks to see if the specified XECB has been posted. If not, the issuing task is marked waiting and its TID is entered into the table entry.

Whenever SVC 94 (X'5E') is invoked, register 1 must point to a field that contains the XECB name. Register 14 must contain the address of the table entry.

If register 14 does not point to the correct table entry, the correct entry is found through the XECB name pointed to by register 1. For this purpose, subroutine FNDLOOP is used. Authorization of the task is tested by means of subroutine XECCHK.

When the entry is found, the traffic bit in the XECB is tested. If it is off, the task status is set to "WAITBND (X'82')", and the waiting task's TID is entered into the table entry. (The task's continuation address is lowered by two bytes for re-SVC when the task gets selected.)

If the XECB was already posted, an immediate exit is taken to task selection and register 15 contains a return code of X'00', while registers 1 and 14 are set to zero. If the abnormal termination flag is posted in the XECB table entry, that is, the communicating task has broken communication without XPOSTing the waiting task, the communicating task gets control with a return code of X'08'.

Return codes in register 15:

0 (X'00')	Successful completion, the XECB has been posted
4 (X'04')	XECB name not found in table
8 (X'08')	Communication with the other task using this XECB was broken. The other task issued an SVC 92 (X'5C') with TYPE=DELETALL.
13 (X'0D')	Task not authorized to issue XWAIT. The return code is made up of 12 (X'0C') plus, in the rightmost two bits, the XPOST access code that is contained in the table entry (see the description of SVC 92 (X'5C') for an explanation of the access code).

SVC 95 (X'5F' - EXIT AB)

This supervisor call is invoked by an EXIT AB macro; it provides for a return from the user's abnormal termination routine to the supervisor to reset the cancel condition and ABEND indication in the TIB control block after the error condition has been cleared up by the Exit routine. Control is then returned to the user program and processing continues with the instruction following the EXIT AB macro.

Before the abnormal termination routine is entered, the logical transient area (LTA) is released. A request for an OC exit while an AB exit is active is ignored, a request for an IT exit is delayed till macro EXIT AB is issued. Any program check in the AB exit routine forces immediate termination.

Note: If an interrupt is delayed, the TTIMER macro returns a value of zero in register 0. Therefore, a contents of zero indicates that either no time interval has been set or a time interval has elapsed during abnormal termination processing and handling of the timer interrupt is suspended until processing is completed.

SVC 96 (X'60')

Reserved

SVC 97 (X'61')

Reserved

SVC 98 (X'62' - EXTRACT/MODCTB)

Provides supervisor support for the macros EXTRACT and MODCTB. In contrast to the FAST SVC (x'6B') services GETFLD and MODFLD the caller is redispatched on exit from EXTRACT or MODCTB.
The EXTRACT macro can retrieve and supply

- Partition-related information like boundaries, PIKs, SYSLOG IDs and GETVIS information.
- Device-related information from physical unit blocks or as retrieved by a 'SENSE-ID' command.
- Machine-related information like CPU ID or control registers
- System-related information like boundaries, GETVIS values and allocations of partitions in specified spaces.

The MODCTB macro is used to change control block entries.
For more detailed information about the macros, parameters and meanings refer to the macro description.

SVC 99 (X'63' - GETVCE)

Provides the supervisor service for the GETVCE macro. It passes device specific information back to the user. The following table describes the layout of the parameter list the address of which is contained in general register 1.

Bytes	Description
0 – 3	Address of output area
4 – 7	Address of device identifier
8	Length of output area – 1
9	Device identification code:
	Bits 0-3 Device class code 0000 Class = DASD 1000 Class = TAPE 0100 Reserved 0010 Reserved 0001 Reserved 1111 Class = ANY Bits 4-7 Device addressing code 0000 VOLID is given a 6-byte volume serial number 0001 LNO code is given 2-byte field in CCBCLS format 0010 CUU code is given 2-byte cuu address is given 0011 DEVTYPE code is given
10	Device type code or zero
11	Macro version identifier
12 – 13	Physical unit number (cuu) or zero
14 – 15	Data length
16	Key length
17	Record number
18	Reserved (must be zero)
19	Processing flags
20 – 21	Remaining track balance or zero if not known

Figure 34. Format of the SVC 99 (X'63) Parameter List

For the output format and the return codes refer to the GETVCE macro (“Macro Descriptions”).

SVC 100 (X'64')

Reserved.

SVC 101 (X'65' - MODVCE)

The device specified by the logical unit or device address in the MODVCE macro is interrogated for status, volume and device characteristics. The volume characteristics table (VCT) is updated accordingly. It must be used whenever a program changes the VOLID of a device or whenever an update of the device characteristic as stored in the AVR-table is required or seems to be necessary. See also the description of the MODVCE macro.

SVC 102 (X'66' - GETJA)

Provides supervisor support for the macro GETJA. For the format of the GETJA macro, refer to the GETJA macro description.

The supervisor service differentiates between a GETJA issued by by job control and a GETJA issued by any other task. The GETJA macro if issued by job control

requires general register 0 to contain the Function code, which has been defined as described below:

Function code:

0 UPDATE Update all account information, maintained by the supervisor

1 CLRTIME Reset JOB related information to zero.

2 RESET Reset JOB-STEP related information.

The service, if requested by any user other than job control, always forces the account information, maintained by the supervisor to be updated regardless of the contents of register 0.

The time counters (ACCTCPUT, ACCTOVHT and ACCTBNDT) in the job accounting interface partition tables (ACCTxx) are replaced with the most current CPU time, OVERHEAD time and ALLBOUND time.
The OVERHEAD time is distributed in proportion to the CPU time accumulated for each of the active partitions.
The ALLBOUND time is distributed in equal parts among the active partitions.

SVC 103 (X'67')

Performs input/output operations for SYSFIL on FBA devices. Register 1 contains the address of the CCB/IORB. The code consists of:

- A resident part, within the supervisor
- A pageable part, residing in the SVA (Module $IJBFBA).

The resident part contains the following supervisor functions:

- Check device type
- Validate I/O area address
- Gate SVC 103 (X'67') against simultaneous use of a disk information block (DIB).
- Establish an interface to the SVC 0 routine, the I/O interrupt handler, the dispatcher, and their appropriate supervisor subroutines.

The pageable part contains the following data management functions:

- Check CCB/IORB contents
- Initialize data buffers (CIDF, RDF)
- Perform blocking and deblocking between user's I/O area and data buffer
- Supply CCB/IORB return information

SVC 104 (X'68' - EXTENT)

Serves as DASD file protect interface for adding, returning or deleting extent information. For more details, see the macro description of the EXTENT macro.

SVC 105 (X'69' - SUBSID)

Provides supervisor support for an execution time subsystem identification. The support consists of three functions, defined by a value passed in register 0. For more details, refer to the macro description of the SUBSID macro.

SVC 106 (X'6A')

The area specified by registers 1 and 2 is invalidated and the storage protection key of this area is set to the value provided in register 0. The registers contain the following operands:

R0 The storage protection key to be used by the SSK instruction.
R1 Begin address (in first page) of the area to be invalidated. The high order byte must be zero.
R2 End address (in last page) of the area to be invalidated.

This function can only be used by OCCF or a task running in an ICCF partition and having a storage protection key of zero.

SVC 107 (X'6B' - Fast SVC)

The "FASTSVC = SVC 107 (X'6B')" has been established to provide retrieval and modifications of fields which are only known to the supervisor. This SVC covers a lot of services and whenever possible, runs over a "Fast path" (mainly bypasses GENERAL ENTRY [GENENT] and GENERAL EXIT Routines) and returns control to the caller without redispatching. Special services can therefore be invoked within other SVC routines and within supervisor appendages.

The FASTSVC functions can be included in reentrant code without special provisions, since all parameters are passed in registers. If any input parameter is invalid, the request owner is canceled.

These functions are provided by various macros which set up different Function codes (see below).

Macro	Function
GETFLD	Get a task/partition-related information
MODFLD	Modify a task/partition-related field
TREADY	Post or cancel a task/partition
TSTOP	Deactivate the current task or partition
RLOCK	Obtain access to a specified resource or wait for it
SRCHFLD	Retrieve PUB index
DEVUSE	Force a device to be set "in use"
SENTER	Enter a Subsystem
SLEAVE	Leave a Subsystem
DEVREL	Release a device that was "in use"
VIOPOINT	Point to VIO control block (VIORB)
VALID	Validate a specified area
GETJA	Update Job Accounting information

Note: All services are accessed via function codes in register 15. See the following table for a description of the function codes.

SVC 107 (X'6B') Function Codes

For a more detailed description of the services see the description of the GETFLD services. Any of the FASTSVC services belongs to one of the following three classes:

Class	Usage
A	Unrestricted usage. The function can be invoked in any variation from any type of code. The fast path is always taken.
B	Task related usage. A request other than from problem program code can refer in the TASK parameter only to the current task. Requests from appendages related to asynchronous events like I/O interrupts are not allowed, since no current task is defined in this case. The fast path is taken only for requests referring to the current task.
C	Problem program usage. The function can only be invoked from problem program code. The fast path is never taken. **Note:** System task code is considered as problem program code.
D	Service is fast path service as such, but does not give control back to user. SVC 107 code is left to some other phase. This phase may or may not be interruptable.

The class is given for each function (see SVC X'6B'). If an invalid request for a class B or C function is issued, the request owner (i.e. the current task or the owner of the asynchronous event) is canceled.

MACRO	OPTION	FUNCTION CODE DEC	FUNCTION CODE HEX	SERVICE CLASS	AUTHORIZED USER	BRANCH ENTRY
TREADY	LQ	00	00	A	LOG-TASK	
TREADY	NO	01	01	A	IPL	
TREADY	IO	02	02	A		
TREADY	VTAM	03	03	A	VTAM	
TREADY	CANCEL	04	04	A	VTAM,POWER,ICCF	
TREADY	VCANCEL	05	05	A	VTAM	
GETFLD	SAVAR	06	06	A		YES
				B	SYSTEM	
GETFLD	PPSAVAR	07	07	A		
GETFLD	LTAPTR	08	08	A	POWER,VTAM	
GETFLD	CNCLCODE	09	09	A	EOT,TERM.,POWER,VTAM	YES
GETFLD	PIK TASK	10	0A	A		YES
GETFLD	MAINTASK	11	0B	A		
GETFLD	VTAMOPEN	12	0C	A	VTAM	
GETFLD	VTAMDISP	13	0D	A	VTAM	
GETFLD	AOTPTR	14	0E	A	VTAM	
MODFLD	SYSRESW	15	0F	A	KEY 0	
MODFLD	CNCLCODE	16	10	A	VTAM,POWER,EOJ	
MODFLD	VTAMOPEN	17	11	A	VTAM,SYSTEM	
MODFLD	VTAMDISP	18	12	A	VTAM,SYSTEM	
TREADY	START	19	13	A	JCL,POWER	
TREADY	OC	20	14	A	JCL,POWER	
TREADY	CANCEL	21	15	A	POWER	
TSTOP	SYSBND,NO	22	16	C	SYSTEM	
TSTOP	SYSBND,YES	23	17	A	SYSTEM	
TSTOP	STOP	24	18	C	JCL	
TSTOP	UNBATCH	25	19	C	JCL	
GETFLD	CNCLALL	26	1A	A	TERMINATOR	
GETFLD	ICCFPP	27	1B	A	ICCF	YES
MODFLD	SAVAR	28	1C	A	IPL	
MODFLD	CNCLALL	29	1D	A	TERMINATOR	
GETFLD	SYSRESW	30	1E	A		

Figure 35 (Part 1 of 5). SVC 107 (X'6B') Function Codes

MACRO	OPTION	FUNCTION CODE DEC	FUNCTION CODE HEX	SERVICE CLASS	AUTHORIZED USER	BRANCH ENTRY
GETFLD	ICCFRO	31	1F	A	ICCF	
GETFLD	ACLOSE	32	20	A		
GETFLD	STATUS	33	21	A	ICCF	
MODFLD	ICCFPP	34	22	A	ICCF	
MODFLD	ICCFRO	35	23	A	ICCF	
MODFLD	ACLOSE	36	24	A	EOJ	
GETFLD	NSUB	37	25	A		YES
GETFLD	CPUTIME	38	26	A		
MODFLD	VSAMOPEN	39	27	A	OPEN/CLOSE	
GETFLD	ABINPR	40	28	B	ICCF	
TREADY	ICCF	41	29	A	ICCF	
GETFLD	LTAACT	42	2A	A	ICCF	
GETFLD	OPENSVA	43	2B	A		
MODFLD	OPENSVA	44	2C	A		
MODFLD	ICCFSVC	45	2D	B	ICCF	
GETFLD	PAGEIN	46	2E	A		
GETFLD	PAGEOUT	47	2F	A		
GETFLD	TERMACT	48	30	A	ICCF	
GETFLD	EOTACT	49	31	A	ICCF	
GETFLD	PCEXIT	50	32	A		
GETFLD	ITEXIT	51	33	A		
GETFLD	CNCLCOD2	52	34	A	EOT,TERM.,POWER,VTAM	
GETFLD	OCEXIT	53	35	A		
		54	36	A	RESERVED	
		55	37	C	RESERVED	
		56	38	C	RESERVED	
		57	39	A	RESERVED	
		58	3A	A	RESERVED	
		59	3B	A	RESERVED	
		60	3C	A	RESERVED	
GETFLD	OCCFACT	61	3D	A	OCCF	
GETFLD	BALANCE	62	3E	A	BAM	
GETFLD	SSFLAGS	63	3F	A	SYSTEM	
GETFLD	COMRGPTR	64	40	A	POWER,ICCF,OCCF	
GETFLD	ABEXIT	65	41	A		YES
SRCHFLD	CHNUNIT	66	42	A		
SRCHFLD	DEVTYPE	67	43	A		
DEVUSE	PU	68	44	A		
DEVREL	PU	69	45	A		
SENTER	LIBR	70	46	A	LIBRARIAN	
SLEAVE	LIBR	71	47	A	LIBRARIAN	
VIO	POINT	72	48	B		
GETFLD	USECNT	73	49	A		

Figure 35 (Part 2 of 5). SVC 107 (X'6B') Function Codes

MACRO	OPTION	FUNCTION CODE DEC/HEX		SERVICE CLASS	AUTHORIZED USER	BRANCH ENTRY
GETFLD	PUSECNT	74	4A	A		
GETFLD	MOUNTFLG	75	4B	A		
MODFLD	MOUNTFLG	76	4C	A	JCL	
TREADY	POWER	77	4D	A	POWER	
GETFLD	PUBXPTR	78	4E	A		
GETFLD	PCBPTR	79	4F	A	SYSTEM	
GETFLD	TCBPTR	80	50	A		YES
GETFLD	OWNER	81	51	A		
GETFLD	MSECS	82	52	A	SYSTEM,JCL	
MODFLD	MSECS	83	53	A	SYSTEM,JCL	
VALID	READ	84	54	A		
VALID	WRITE	85	55	A		
GETFLD	VSAMOPEN	86	56	A		
MODFLD	PERBIT	87	57	B	SDAID	
GETFLD	PU	88	58	A		
GETJA	PART	89	59	C		
MODFLD	RUNMODE	90	5A	A	SYSTEM	
MODFLD	SASCOPE	91	5B	A	SYSTEM,POWER	
MODFLD	PASCOPE	92	5C	A	SYSTEM,POWER,VTAM	
RLOCK	ALLOCR	93	5D	A	SYSTEM	
		94	5E		RESERVED	
TREADY	ALLOCR	95	5F	A	SYSTEM	
		96	60		RESERVED	
MODFLD	LIBRSERV	97	61	A	LIBRARIAN	
GETFLD	DEVPROP LU	98	62	A		
GETFLD	DEVPROP PU	99	63	A		
GETFLD	EOTRTN	100	64	A		
GETFLD	MODE LU	101	65	A		
GETFLD	MODE PU	102	66	A		
GETFLD	PIK LOGID	103	67	A		YES
GETFLD	PSTAT	104	68	A		YES
GETFLD	IVMCB	105	69	A	VSIC	
GETFLD	RIPLAUTH	106	6A	A	KEY 0	
TREADY	OCANCEL	107	6B	A		
------INVALID ---------		108-114		----------		

Figure 35 (Part 3 of 5). SVC 107 (X'6B') Function Codes

MACRO	OPTION	FUNCTION CODE DEC	FUNCTION CODE HEX	SERVICE CLASS	AUTHORIZED USER	BRANCH ENTRY
GETFLD	COMRGPTR LOGID	115	73	A		
GETFLD	PSTAT LOGID	116	74	A		
GETFLD	DIBPTR PART	117	75	A		
GETFLD	DIBPTR	118	76	A		
GETFLD	LUBTAB PART	119	77	A		YES
GETFLD	LUBTAB LOGID	120	78	A		YES
GETFLD	LUB PART	121	79	A		
GETFLD	LUB LOGID	122	7A	A		
GETFLD	FREELUB PART	123	7B	A		
GETFLD	FREELUB LOGID	124	7c	A		
GETFLD	NPART	125	7D	A		
GETFLD	PUB PART	126	7E	A	POWER	
GETFLD	PUB LOGID	127	7F	A	POWER	
GETFLD	PCEATAB	128	80	A		
GETFLD	PCEPTR PART	129	81	A		
GETFLD	PCEPTR LOGID	130	82	A		
GETFLD	PIB PART	131	83	A		YES
GETFLD	PIB LOGID	132	84	A		YES
GETFLD	LOGID	133	85	A		YES
GETFLD	STORKEY PART	134	86	A		YES
GETFLD	STORKEY LOGID	135	87	A		YES
GETFLD	PUBPTR	136	88	A		YES
GETFLD	PUB2	137	89	A		
GETFLD	NUMLUB PART	138	8A	A		
GETFLD	NUMLUB LOGID	139	8B	A		
GETFLD	CLASSTAB	140	8C	A	KEY 0	
GETFLD	IPLTIME	141	8D	A		
GETFLD	POWJOB PART	142	8E	A	CICS	
GETFLD	POWJOB LOGID	143	8F	A	CICS	
GETFLD	POWSYS	144	90	A		
GETFLD	ALET	145	91	C	see below	
MODFLD	COPYOWN	146	92	A	POWER,JCL	
MODFLD	RESETOWN	147	93	A	POWER,JCL	
MODFLD	VSPTFLG	148	94	A	VSE PERFORMANCE TOOL	
MODFLD	SASCOPE1	149	95	A		
MODFLD	PASCOPE1	150	96	A	SYSTEM,POWER,VTAM	
GETFLD	LOGBAL	151	97	A	BAM, ASSEMBLER	
GETFLD	ABEXIT1	152	98	A		YES
GETFLD	PCEXIT1	153	99	A		
GETFLD	OCEXIT1	154	9A	A		
GETFLD	ITEXIT1	155	9B	A		
SENTER	ALET	156	9C	D	AR, DUMP	
SLEAVE	ALET	157	9D	D	AR, DUMP	
GETFLD	SUBCHN	158	9E	A		

Figure 35 (Part 4 of 5). SVC 107 (X'6B') Function Codes

MACRO	OPTION	FUNCTION CODE DEC/HEX		SERVICE CLASS	AUTHORIZED USER	BRANCH ENTRY
GETFLD	SAVAR1	159	9F	B		YES
MODFLD	CELANCH	160	A0	B		
		161	A1		RESERVED	
		162	A2		RESERVED	
		163	A3		RESERVED	
		164	A4		RESERVED	
MODFLD	MNTRSTMP	165	A5	A		
MODFLD	MNTRSPRM	166	A6	A	JCL, AR	
MODFLD	MNTRSFRC	167	A7	C	AR	
MODFLD	MNTRLTMP	168	A8	C		
MODFLD	MNTRLPRM	169	A9	C	JCL, AR	
MODFLD	MNTRLTPA	170	AA	C	JCL	
GETFLD	PUBXPTR, CUU	171	AB	A		
GETFLD	NXTOWNER	172	AC	A		
MODFLD	MNTRLPRA	173	AD	C	JCL, AR	
GETFLD	POWCAT	174	AE	A		
GETFLD	CELANCH	175	AF	A		YES
GETFLD	MNTINFO	176	B0	A		
MODFLD	MNTINFO	177	B1	A		
GETFLD	RXEOJPTR	178	B2	A		
MODFLD	RXEOJPTR	179	B3	A		
GETFLD	REALKEY	180	B4	A		
GETFLD	DSPACE	181	B5	A		
MODFLD	RID	182	B6	A	VTAM	
GETFLD	PTRACE	183	B7	A		
MODFLD	IPWSEGM	184	B8	A		
GETFLD	UCB	185	B9	A		
MODFLD	MODE,LU	186	BA	A		
MODFLD	MODE,PU	187	BB	A		
MODFLD	PMODE,LU	188	BC	A		
MODFLD	PMODE,PU	189	BD	A		
GETFLD	PMODE,LU	190	BE	A		
GETFLD	PMODE,PU	191	BF	A		
MODFLD	ACEEPTR	192	C0	A		
GETFLD	ACEEPTR	193	C1	A		YES
MODFLD	SYSAL	194	C2	A		
GETFLD	ITACTIVE	195	C3	A		YES
GETFLD	JCLPARM	196	C4	A		
GETFLD	PU,CUU	197	C5	A		
GETFLD	EXECMODE	198	C6	A		
TREADY	IO,PART	199	C7	A		
GETFLD	TIDSTR	200	C8	A		

Figure 35 (Part 5 of 5). SVC 107 (X'6B') Function Codes

GETFLD FIELD=ALET can be invoked by CICS, POWER, OCCF, VTAM, VSE PERFORMANCE TOOL and such subsystems that announced themselves to the system with the PRODID service.

SVC 108 (X'6C' - SECHECK)

The SVC X'6C' routine offers a set of services for the access control feature. These services are:

- Access Control Authorization Checking

  The service checks whether a user is allowed to access a specified r esource.

- POWER * PUN DISP=I Processing Service

  This routine returns a pointer to the Userid from the JPL pointed to by the COMREG of the specified partition

- POWER SLI Processing - Preparation Routine

  This routine temporarily passes the JPL of the user in the serviced partition to POWER in order to be used for access checking of the members to be included.

- POWER SLI Processing - Cleanup Routine

  This routine returns the JPL, which was temporarily passed to POWER.

Access Control Authorization Checking

This service checks whether a user is allowed to access a specified resource. A pointer to the resource name is given through an access control authorization parameter list (DTSAPL). The address of this DTSAPL must be supplied in register 1. The access control authorization checking routine, loaded into the SVA during IPL, is entered from the supervisor to check whether access is allowed or not.

Return codes from the SVC:
Register 15 will pass back one of the following return codes.

0 (X'00')	Access allowed.
4 (X'04')	Access control facility not active.
8 (X'08')	Access control violation.
12 (X'0C')	Resource name not in access control resource table (DTSECTAB), which is only possible for sublibrary and member.

The access control authorization checking routine runs as an SVC appendage to the SVC 108 (X'6C') routine. The routine is re-entrant and loaded into the SVA during IPL after a pointer to that routine has been initialized in the supervisor security vector table (SCYVECTB), addressed by means of SYSCOM.

This routine performs the actual access control check for the resource referenced in the DTSAPL against the access control resource table, which is initialized by the user and contains the various resources to be protected (such as libraries, sublibraries, members and files).

Based on the DTSAPL, access control authorization checking takes place either for a library, a sublibrary, a member or a file:

- A library request originates from a VSE system component such as the librarian or JCL. This component has to determine whether a system library is protected or not. The component issues an OPEN for a DTF with a DTSAPL addressed by the DTF. The $$B-transient OPEN phase passes control to the access control OPEN appendage routine which builds the DTSAPL for the library to be validated before issuing the SVC 108 (X'6C').

- A sublibrary request originating from a VSE system component such as the librarian. This component has to determine whether a sublibrary is protected or not.
- A member request originating from a VSE system component such as the librarian. This component has to determine whether a member of a sublibrary is protected or not.
- A member request as the result of a FETCH/LOAD/CDLOAD request which is intercepted by the supervisor for access control authorization checking. A direct branch is made into the access control SVC processing routine. This routine prepares the DTSAPL for the phase to be loaded and then comes to this SVC appendage routine to perform the actual validation.
- A member request as the result of a call of the vendor interface. The call is intercepted by the supervisor for access control authorization checking. A direct branch is made into the access control SVC processing routine. This routine prepares the DTSAPL for a check to the member $IJBVEND and then comes to the SVC appendage routine to perform the actual validation.
- A file request originates from any OPEN request for a file on a DASD volume or on magnetic tape. As stated above, the $$B-transient OPEN phase transfers control to the OPEN appendage routine which completes the DTSAPL for processing by the SVC 108 (X'6C') routine.

The access control authorization checking routine uses main input parameters to do the access control validation:

DTSJPL Addressed by the

1. APL (APJPL), or if not available the
2. TCB (TCBUCNTL) or if also not available the
3. COMREG (IJBJPL)

and initialized by access control from the user profile record (macro with DSECT available).

DTSAPL Contains the resource or object to be validated against the access control resource table (DTSECTAB) for the user defined by the DTSJPL (macro with DSECT available).

DTSECTAB This is the access control resource table which contains all resources to be protected plus the corresponding access classes and logging options.

The checking routine locates the object defined by the DTSAPL in the DTSECTAB; it checks the access classes and access rights stored in the DTSJPL against those given in the matching table entry. If the user has any of the allowed access classes assigned via the profile record and at least the requested access right, (transferred to the DTSJPL by job control), then the user has passed the access control authorization check. An access class of zero means that no secured resources can be accessed.

If the request was to check if a user has the authorization to catalog a $$B-transient, then the DTSJPL field JPSA is checked for this special authority.

If the user is found to be unauthorized, the request is canceled after logging the request to the log data set (if VSE Access Control - Logging and Reporting is installed).

The logging options in the DTSECTAB are set per class. Every access to this resource is logged if specified; violations are always logged. Determination depends on the access classes of the user and those in the table entry and the corresponding logging classes. The data for the required logging record is moved directly into the logging queue by means of a queue manager. The queue entries are written to the log data set by means of the logger system task.

Return codes from the appendage are set accordingly in register 15 to be handled by the access control check SVC:

0 (X'00')	Normal return with no logging: The user has authority for the resource.
4 (X'04')	Post logging task and continue normal operation: The user has authority for the resource but the access request must be logged to the log data set.
8 (X'08')	Post logging task and cancel: The user is not allowed to access the resource. He is canceled with cancel code 11 (X'0B') and the request is logged to the log data set.
12 (X'0C')	Cancel due to inactive logger: The user is not allowed to access the resource. He is canceled with cancel code 11 (X'0B') but the request can not be logged to the log data set because VSE Access Control - Logging and Reporting is not installed.
16 (X'10')	'Log-queue-full' wait condition:
20 (X'14')	Cancel due to authorization routine processing error: Cancel code is 10 (X'0A').

Detailed information about the return code is contained in the DTSAPL fields APJCL, APERR, APOAT, APUAR and APSPR. For the description of the content of these fields see macro DTSAPL.

*POWER * $$ PUN DISP=I Support*

This service is available for POWER only. APOPT indicates APGETUSR. The partition as specified via SYSLOG ID in the lower half of the APOBJ field in the APL is searched and when found, a pointer to the User ID out of the JPL pointed to by the partition COMREG is returned in Reg 1. Note, that Reg. 1 may point to an empty string (i.e. eight blank characters) in case the partition is not running under control of a user.

POWER SLI Processing - Preparation Service

This service is available for POWER only. APOPT indicates APSLICON. Access Checking first checks for a JPL pointed to by the TCB and if no such is available, then take the JPL pointed to by the COMREG. In order for POWER to do the necessary access checking for the members to be SLI INCLUDEd under control of the user, that is active in the serviced partition, this service gets a pointer to the JPL of the partition, that the SLI INCLUDE is done for, and stores it into the appropriate TCB field. Input to this service is the APL, which contains in the lower halfword of APOBJ the SYSLOG ID of the partition to be found. The service searches all existing partitions for a matching SYSLOG ID. When it is found, the address of the JPL as being pointed to by the COMREG of the partition is stored into the caller's TCB (TCBUCNTL).

POWER SLI Processing - Cleanup Service

This service is available for POWER only. APOPT indicates APSLIDSC. The service clears the TCB field pointing to the JPL (TCBUCNTL) in order for subsequent access checks to be done for the user, that is active in the POWER partition.

SVC 109 (X'6D' - PAGESTAT, SPLEVEL = 1)

Page management PAGESTAT service for VSE/ESA releases up to 1.2 (SPLEVEL = 1). General register 0 points to the begin address and general register 1 to the end of the area to be inspected. The macro service provides the status of this area. (See “SVC 121 (X'79'- Page Management Services)” on page 294).

SVC 110 (X'6E' - LOCK/UNLOCK)

Lock Manager (LOCK and UNLOCK)

Locks a resource against simultaneous use by other tasks.
Unlocks a given resource that was previously locked.
The SVC is invoked by the LOCK and UNLOCK macros.

For more information refer to z/VSE Supervisor Diagnosis Reference, Chapter "Lock Management".

SVC 111 (X'6F')

Reserved.

SVC 112 (X'70' - MSAT)

Manipulates assignment and device ownership information. For details of the external specification, refer to the macro description of the MSAT macro.
Input is a parameter list in Reg.1, containing an identification of the required subfunction. The subfunctions can be group together as follows:

1. Retrieve assignment information for one or more logical units in a given partition (ID=INQ,CKU,RTL,RTP,RTL1).
2. Modify the assignment information for one or more logical units in a given partition
   (ID=PER,DEL,ALP,ALT,NXT,RSU,RSA,NPM,NTM,DRL).
3. Modify the status of a unit record device in a given partition relative to spooling by VSE/POWER (ID=PST,PSP).
4. Modify the status of a device relative to physical addressing access (i.e. without a logical unit) by a system function or by an authorized component. (ID=DVU,DVR).

The following data areas are accessed (see also the I/O Control Blocks in z/VSE Supervisor Diagnosis Reference, Appendix B.

LUB	To retrieve/modify the current assignment of a logical unit in a given partition.
LUB Extension	To indicate what type of assignments of a logical unit in a given partition are stored (in addition to the current) and to store the permanent assignment or set up a pointer to a chain of SAT (Stored Assignment Table) entries.
SAT Entry	To store up to 5 permanent or permanent alternate or temporary alternate assignments of a logical unit in a given partition, together with control information.
PUB	To control the status of a device (up or down) and to retrieve the device type code.

PUB Extension	To maintain ownership and usage counters of a non-sharable device (e.g. tape and TP device).
Device Usage Field	To maintain ownership and usage counters of a partition-sharable device for the system or for a given partition.
PUB Ownership Entry	To maintain system/partition ownership indicators of a device (for compatibility only).

SVC 113 (X'71' - XPCC)

Provides cross-partition communication control (XPCC), as requested by macros XPCC, XPCCB, and MAPXPCCB. For descriptions of the macros refer to the macro descriptions.

The cross-partition communication control facility enables the various VSE subsystems to communicate with each other or with their user applications. The support provides supervisor service for the following functions:

- Identify:
  The VSE subsystem or user application identifies itself to the XPCC.
- Termination control:
  Removes information about the application from XPCC. Application may no longer use XPCC services (except with a new IDENT). Or, depending on the parameter specified, only the existing connections can be used, no new connection can be built.
- Connect:
  Establishes a connection between two applications which have identified themselves to XPCC by means of IDENT. The connection is completed and data transfer can start when both sides have issued their CONNECT. The connection is related to the corresponding applications.
- Disconnect:
  Breaks one specific connection or all connections established for an application.
- Transmit data without reply:
  With SEND the other side of a connection is posted, enabling it to receive data. With RECEIVE, the data is moved into the receiver's input buffers.
- Transmit data with reply:
  Same as SEND/RECEIVE, but with REPLY, the receiver can immediately transfer data into a predefined buffer in the sender's partition.
- Transmit data immediately:
  At CONNECT-time, a receive-buffer can be specified which is used in the SENDI protocol to receive the data immediately, when the SENDI of the other side is executed. The SENDI protocol forces both sides to issue SENDI alternately.
- Clear:
  Allows the sender of a message to withdraw that message before the receiver has issued a RECEIVE for picking up the message.
- Purge:
  Allows a message to be purged, from the receiving side, indicating to the sender that it is not able to receive this message.

Two control blocks are used by XPCC to control data transmission between partitions. The anchor to the identification control blocks (IDCB) is in the XPCC code. All IDCBs are in one chain.

For each CONNECT request a CRCB is built which contains all relevant information from both sides of a connection. Each CRCB is a member of two different CRCB chains. The CRCB chain is pointed to by a field in the IDCB.
The CRCB consists of 3 parts.

- Part 1 contains information, common to both partners.
- Part 2 contains information, describing the partner which issued connect first.
- Part 3 contains the same information as Part 2, but for the other partner.

SVC 114 (X'72' - VIO)

Supports the allocation, extension and deallocation of VIO files. For details of the external specifications, refer to the description of the VIO macro. The selected function is identified by a function code supplied in register 15.

The following functions are available:

- Allocate VIO file
- Extend VIO file
- Deallocate VIO file

Allocate (Function code = 0 - VIO OPEN)

Input is the address of a parameter list in register 1 and, optionally, a size specification in register 0. If register 0 contains zero, the size specification is taken from the parameter list. The requestor is canceled if the address in register 1 is invalid (ERR25) or any specified parameter is invalid (ERR21).

Space allocation is based on a byte string. Each byte corresponds to a VIO segment and contains X'00' for a free segment and X'FF' for an allocated segment. A number of not necessarily contiguous VIO segments sufficient for the requested size is allocated. Furthermore, a VIOTAB entry (see also z/VSE Supervisor Diagnosis Reference, Appendix A) and one or more File Segment Tables are allocated in the system GETVIS area. For a successful allocation, the corresponding Block Tables entries and several fields in the VIOTAB are initialized. The VIOTAB address is returned in register 1 and the return code in register 15 is set to 0.
For an unsuccessful allocation (not enough VIO or system GETVIS space), intermediate allocations are freed up and the return code in register 15 is set to 8.

Extend (Function code = 1 - VIO EXTND)

Input is a VIOTAB address in register 1 and a size increment in register 0. The requesting task is cancelled if register 1 does not point to a VIOTAB (ERR25) or if it is not the owner of the VIO file described by the VIOTAB (ERR21).

A number of additional VIO segments sufficient for the requested size increment is allocated. If necessary, additional file segment tables are allocated in the system GETVIS area. For a successful allocation, the corresponding block tables entries are initialized, field VIORBASZ in the VIOTAB is adjusted to the new total size of the VIO file and the return code in register 15 is set to 0.
For an unsuccessful allocation (not enough VIO or system GETVIS space), intermediate allocations are freed up and the return code in register 15 is set to 8.

Deallocate (Function code = 2 - VIO CLOSE)

Input is an option indicator in register 0 and, if the indicator is zero, a VIOTAB address in register 1. The requesting task is cancelled if the indicator is non-zero, except for EOT or JC, (ERR21) or if the VIOTAB address is invalid (ERR25) or if it is not the owner of the VIO file described by the VIOTAB (ERR21).

If the option indicator is non-zero, all VIO files owned by the requesting task and having the life-time of a job-step (indicator = X'08') or of a job (indicator = X'10') are deallocated. If the indicator is zero, only the specified VIO file is deallocated. All allocated VIO segments and all associated system GETVIS space is freed.
The return code in register 15 is always set to 0.

SVC 115 (X'73' - PWROFF)

The SVC 115 (X'73') disconnects a VM guest

SVC 116 (X'74' - NPGR)

Allocates or reallocates the programmer LUBs of the specified partition(s) when called by the macro NPGR (static partitions only).

Register 1 contains the address of the NPGR parameter list. Register 15 contains a return code after completion. See also the macro description of the NPGR macro. When called by JCL via the NPGR macro, the SVC 116 (X'74') routine takes the specified NPGR values, performs some checks (see return codes) and transfers these values into the corresponding PIB(s). When starting a partition the first time after IPL, the PIB values are taken and the corresponding LUB Table is allocated for that partition within the main LUB Table pool, which was statically reserved at supervisor generation time via the NPGR parameter.

SVC 117 (X'75')

SVC 117 (X'75') provides services to maintain the multi-processing environment, e.g. start/stop an additional cpu. It also provides tasking and multi-processing servcies for vendor products.

SVC 118 (X'76' - CPCOM)

The SVC 118 (X'76') allows authorized subsystems to submit CP commands via the DIAGNOSE X'08' interface through a supervisor service. The command is passed unchanged to CP and the completion code is returned to the caller. The retrieval of information from CP is not supported.

The function is to be considered as a generalization of the current CPCLOSE macro (SVC 56 - X'38') and is currently authorized to VSE/POWER and FTP.

For the description of the CPCOM macro format see “CPCOM” on page 6.

SVC 119 (X'77')

Service processor support.

SVC 120 (X'78' - XMOVE)

See also the description of the XMOVE macro.

The SVC 120 is used by ACF/VTAM to perform the following function:

- Set the traffic bit in a specified ECB on and post all tasks waiting on that ECB.

SVC 121 (X'79'- Page Management Services)

This SVC covers the page management services for SPLEVEL greater greater than 1. The specific service is identified by the function code passed in general register 15. The functions are:

- Function code 1 - PFIX macro
- Function code 2 - PFREE macro
- Function code 3 - RELPAG macro
- Function code 4 - FCPGOUT macro
- Function code 5 - PAGEIN macro
- Function code 6 - PAGESTAT macro
- Function code 7 - SETPFA macro

PFIX Service

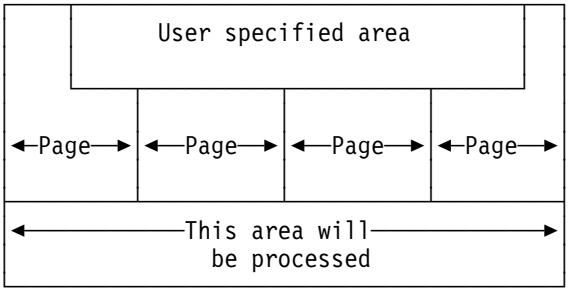
Fixes as many pages as requested by the PFIX macro. A PFIX request is ignored if it is issued by a program running in REAL mode, and return code 0 is passed in register 15.

When the SVC 121 (X'79') routine is entered, general register 15 contains the function code equal to 1 and general register 1 points to the list of pages to be fixed. The length of an entry is 8 bytes. The first four bytes contain the beginning address of the area that is to be fixed and the last four bytes contain the 370-length of this area.

Bytes	Description
0 – 3 4 – 7	Address of the area to be paged–in. length minus 1 of area

The addresses are considered to be 31 bit addresses if the SVC requestor is running in 31 bit addressing mode and uses SPLEVEL greater than 1, otherwise the addresses are treated as 24 bit addresses.

Note: The begin and end addresses of the specified area hardly ever coincide with the begin and end addresses, respectively, of a page. Therefore, the active area is extended to all pages referenced by the specified area, as shown below, before it can be processed.



The PFIX requests are gated, that is, a task is set to "PFXBND (X'92')" if it issues a PFIX request for a partition for which another PFIX request is still being processed.

Return codes are passed in general register 15:

0 (X'00')	all pages are fixed.
4 (X'04')	the maximal number of pages allowed to be pfixed for the partition is exceeded by this request alone
8 (X'08')	the maximal number of pages allowed to be pfixed for the partition is exceeded due to previous PFIX requests
12 (X'0C')	negative length or invalid address detected or begin address is greater than end address
16 (X'10')	at least one page is pfixed in PFIX area-3, but PFIX request
20 (X'14')	invalid function code or option parameter

PFREE Service

Frees as many pages as requested by the PFREE macro. A PFREE request may come from a user task or from the RSTRT command processor. A PFREE request is ignored if it is issued by a program running in REAL mode, and return code 0 is passed in register 15.

When the SVC 121 (X'79') routine is entered, general register 15 contains the function code equal to 2 and general register 1 points to the list of pages to be fixed. The format and the characteristics are same as for the PFIX macro (see above). The pages are freed sequentially until the list of requests is exhausted. If a page is not addressable or not PFIXed, the PFREE request for this page is ignored.

When a page is freed, the PFIX counter is decreased by 1. If the counter is zero, the page frame is released and enqueued at the end of the page selection queue; any task waiting for a freed page will be posted, and the next page to be freed is selected if this page is temporarily fixable in the released page frame (NFRP bit in S370FLG of the page frame table entry is OFF).

If the PFREEd page frame is reserved for another PFIX request, the address of the page frame table entry is inserted into the PFTERSVD of partition control block (PCB) and the task that issued this PFIX is posted ready to run. Return codes are passed in general register 15:

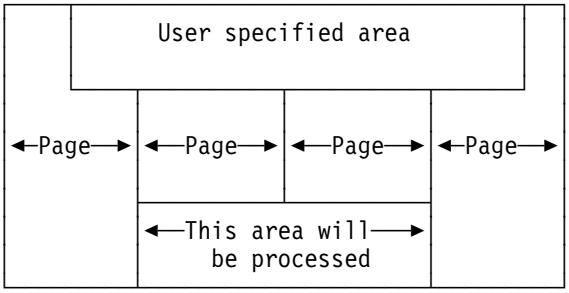
0 (X'00')	all pages are freed.
12 (X'0C')	negative length or invalid address detected
20 (X'14')	invalid function code or option parameter

RELPAG Service

Releases a list of specified pages and makes the frames available on top of PSQ. Part of the code is common for both, RELPAG (function code = 3) and FCEPGOUT (function code = 4).

When the SVC 121 (X'79') routine is entered, general register 15 contains the function code equal to 3 and general register 1 points to the list of pages to be released. The format and the characteristics are same as for the PFIX macro (see above). There is one exception:

Note: The begin and end addresses of the specified area hardly ever coincide with the begin and end addresses, respectively, of a page. In contradiction to all other page management services the handled area is restricted on those pages completely covered by the specified area, as shown below, before it is processed.



The common actions are:

- If the page is outside the address range of the requested program's partition, the return code is set to 4 (X'04').
- If the page is fixed the return code is set to 8 (X'08').
- If a negative length is detected, the return code is set to 2 (X'02').

If the list of areas that are to be handled is not completely in the requesting program's partition, the request is ignored and return code 16 (X'10') is set.

The parameter list is processed until the end of list is reached, or a page has to be handled for which none of the above three conditions are true.

The latter condition causes a return to the caller with offset 4 to allow specific actions. These actions for a RELPAG request are:

- Reset reference, change and valid copy on page data set bits.
- If page is connected are page-out is active, wait on completion of page I/O.
- If the page is not disconnected, handle POSL, delete entry from PGQO whenever feasible, clear the page frame and enqueue the associated PFTE on top of the PSQ.

FCEPGOUT Service

performs the necessary actions to page-out the pages specified in the supplied parameter list. If the system is running without Page-Data-Set, no actions are taken.

When the SVC 121 (X'79') routine is entered, general register 15 contains the function code equal to 4 and general register 1 points to the list of pages to be

paged-out. The format and the characteristics are same as for the PFIX macro (see above).

The following action is executed for the FCEPGOUT request, if the page is addressable:

- Reset the reference bit in the PTE.
- If the change bit is off, enqueue the PFTE at searching depth in the PSQ.
- If the change bit is on, enqueue the PFTE at top of the PSQ.

If the page is not addressable, no specific action is taken.

PAGEIN Service

Provides support for the PAGEIN macro by initiating the PGN system task. When the SVC 121 (X'79') routine is entered, general register 15 contains the function code equal to 4 and general register 1 points to the list of area specifications. The format and the characteristics are same as for the PFIX macro (see above).

Register 0 points to an ECB if an ECB address was specified in the PAGEIN macro, otherwise it contains zero.

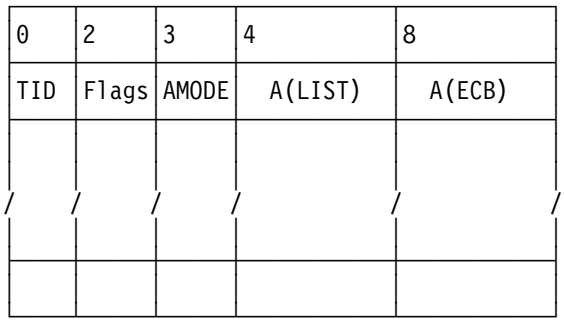
The PAGEIN request is ignored under following conditions:

- The PAGEIN macro was issued by a program running in REAL mode.
- The list of areas that are to be paged in is not completely contained in the requesting program's partition.
- The table PAGETAB (see Figure 36 on page 298) is full.
- The ECB address, if specified, is outside the requesting program's partition.

For each PAGEIN request, the SVC routine builds a 12-byte entry in a table called PAGETAB (see Figure 36 on page 298).

The entries of the table are stacked and processed (by the PGN system task) FIFO. The maximum number of table entries is specified during IPL in the PAGEIN parameter of the SYS command.

For a valid PAGEIN request, the SVC routine passes control to the PGN system task either directly or via task selection.



Bytes	Description
0 - 1	Identifier of task that issued the PAGEIN macro.
2	Flag Byte: X'80' PAGEIN request completed/ additional scan needed 40 Reserved 20 At least one page is outside partition boundary 10 At least one entry with a negative length was found 08 Reserved 04 Paging activity too high, termination was requested by LOAD LEVELER 02 Task is terminating, entry has to be deleted 01 Last scan in progress
3	Addressing mode of requestor
4 - 7	Pointer to the areas to be paged—in.
8 -11	Pointer to ECB (if used) or zero.

Figure 36. Page-in Table (PAGETAB)

If the address of an ECB was specified in the PAGEIN macro, information is returned in byte 2 of that ECB as shown below:

Bits	Meaning	Set by: SVC Routine	Set by: PGN Task
0	PAGEIN request completed (see Note below).	Y	Y
1	Page–in table (PAGETAB) is full.	Y	N
2	One or more of the requested pages are outside the address range of the requesting program's partition.	N	Y
3	At least one negative length has been detected in the processed area specifications.	N	Y
4	List of areas that are to be paged–in is not completely contained in the requesting program's partition.	Y	N
5	Paging activity too high. PAGEIN request terminated by LOAD LEVELER.	N	Y
6	Reserved.		
7	invalid function/option	Y	N

Figure 37. Return Information in Byte 2 of PAGEIN ECB

Note: Bit 0 is set by the PGN system task if that task receives control to process the pertinent PAGEIN request, otherwise the bit is set by the SVC routine.

In a system without Page-Data-Set the PAGEIN service causes only the ECB to be posted, provided an ECB was specified in the PAGEIN macro and the specified ECB address is valid.

PAGESTAT Service

When the SVC 121 (X'79') routine is entered, general register 15 contains the function code equal to 6 and general register 1 points to a parameter list of the length of 12 bytes. The layout is given as follows:

0 3	4 7	8 11
A(BEGIN	A(END)	A(ALET)

Figure 38. PAGESTAT parameter list

The first 4 bytes in the parameter list contain the begin address and the next 4 bytes the end address of the area to be inspected. The last 4 bytes contains the address of an ALET or zero.

The macro service provides the status of this area. On return, general register 0 contains the address of the first page of the area with a different status or 0 if all pages have the same status and general register 15 contains the status of the area.

0 (X'00')	address valid, page is used
4 (X'04')	address is valid, page is not used
8 (X'08')	address is invalid, that is address beyond virtual storage or PTEINVAD and PTEIBIT are on in corresponding page table entry.
12 (X'0C')	address is valid, page is connected
16 (X'10')	address belongs to VPOOL

For the format of PAGESTAT macro and status settings refer to the PAGESTAT macro description.

Invocation of the service with beginaddr higher than endaddr results in 'canceled due to invalid address' (ERR25).

SETPFA Service with AREA Parameter Specified

When the SVC 121 (X'79') routine is entered, general register 15 contains the function code equal to 7 and general register 1 points to a parameter list of the length of 12 bytes. The layout is given as follows:

0 3	4 7	8 11
A(PHO)	A(AREA)	FLAGBITS

FLAGBITS	
0	DSPACE=YES specified
1-31	re-...served

Figure 39. SETPFA Parameter List

The first 4 bytes in the parameter list contain the address of the page fault appendage (PHO) routine if PHO processing should be enabled or zero if PHO should be disabled. The next 4 bytes contain the address to the area specified by the AREA parameter. The layout of this area is described in SRL System Macro Reference.

The service establishes linkage between the supervisor and the user-written page fault appendage routine in the page fault appendage TIB located in the PCB (PHOTIB). Only one task of a partition may have an active PHO-appendage.

SVC 122 (X'7A' - SYSDEF)

The SVC X'7A' routines supply services for changing data space defaults/limits and requesting information on either system defaults/limits or the actual data space usage situation in the system.

The SYSDEF SVC is restricted to Job Control and the Attention Routine (for usage in the commands QUERY DSPACE and SYSDEF) and to authorized vendors. Input to the SYSDEF SVC is a control block, as pointed to by R1. A mapping of this control block is available under SYSDEF ID=DSECT macro.
The SVC is not gated, which means that multiple users may run this SVC concurrently. All work fields are located in a work area, that must be supplied by the user. A pointer to this work area must be given in the input control block field DSPSDDAD and its length in DSPSDDLN. The minimum length of this work area is 132 bytes. Note, however, that these fields are changed during SVC processing. To ensure a consistent output, the SVC should not be interrupted by page faults. To ensure this, the input control block as pointed to by R1 and the user supplied work area must reside in PFIXed storage. This is user responsibility. Both 24- and 31-bit callers are allowed, but for 24-bit callers, the addresses of the input control block and the work area are interpreted as 24-bit addresses. Callers must be in primary ASC mode.
The SYSDEF SVC offers the following services:

- Updating Data Space system defaults/limits('sysdef' service)
- Querying Data Space information ('query' services)
  - Data Space system defaults/limits
  - Data Space system actuals
  - Information about a definable subset of all data spaces
    - Information about all data spaces in the system
    - Information about all data spaces owned and/or accessed by a specified partition
    - Information about all data spaces with a specified name
    - Information about the data spaces with a specified name owned by a specified partition

The selected service depends on the first two bytes in the input control block. The first byte, DSPSDSRV, specifies whether 'query' or 'sysdef' services are specified. This byte is set by the SYSDEF macro. All other bytes must be set manually. The second byte specifies the subfunction for the query services. For details refer to the DSECT as being created by SYSDEF ID=DSECT.

Return codes:
Register 15 will pass back one of the following return codes.

0 (X'00')	SYSDEF SVC returns o.k.
4 (X'04')	Passed Work Area insufficient
8 (X'08')	Required input missing
12 (X'0C')	User not authorized (i.e. not AR, JCL, or Vendor)
20 (X'14')	Input Control Block/Work Area not in valid storage
24 (X'18')	Content of input control block incorrect
28 (X'1C')	unable to getvis pfixed work area

Updating Data Space System Defaults/Limits
The data space system defaults/limits are stored in a supervisor internal table (MAPDSINF). This table is pointed to from the SYSCOM. The content of this control block can be changed in a controlled manner by using the SYSDEF SVC. The values, that can be changed are:

- VSIZE reserved for data spaces
- Maximum number of data spaces in the system
- Maximum number of data spaces per partition
- Maximum number of common data spaces
- Default size of a data space

The values to be changed are placed into the input control block (refer to MAPDSSIB for a layout of this area) and indications for every selected parameter are set in DSPSDPRM.
For every accepted parameter, the bit in DSPSDPRM is cleared. In case, at least one of the specified values is not acceptable (i.e. value is out of limits), then the corresponding bit in DSPSDPRM is still set. Return code X'18' will be indicated in R15.

Querying Data Space Information
The 'query' services are split up into a set of subfunctions. The selection of a subfunction is done via setting the byte DSPSDBSF in the input control block accordingly. In the following section, every available service is described in detail.

Data Space System Defaults/Limits and Actuals
The supervisor internal control block MAPDSINF, which is pointed to by the SYSCOM (field IJBDSICB) , contains three subareas:

1. Data Space System Defaults/Limits (Mapping is MAPDSSIB),
2. Data Space System Actuals (Mapping is MAPDSSIB) and
3. Some globals for the data space support.

SYSDEF SVC offers a service to retrieve the information about the defaults/limits and a second service to retrieve the actuals. In DSPSDSRV, 'query' services are indicated and in the following byte, DSPSDBSF, an indication is set, which of the two blocks (Data Space System Defaults/Limits or Data Space System Actuals) is to be returned.
The returned information is placed into the input control block. Its content, as being mapped by MAPDSSIB, is as follows:

- VSIZE reserved/used for data spaces
- Maximum/Actual number of data spaces in the system
- Maximum number of data spaces allowed/allocated per partition
- Maximum/Actual number of common data spaces
- Default size of a data space

Information about how many Data Spaces each Partition Owns
This service returns to the caller a list of all existing partitions together with the actual number of data spaces they own. On return, the address of this list is in DSPSDDAD and its length is stored in DSPSDDLN. The list consists of a number of records. Each record is described by the DSPQRYND dsect, which is created with the MAPQRYDS macro. Together with the string, the block of data space

actuals, as being stored in MAPDSINF (see also MAPDSSIB), is returned as well. This is done to allow a consistent display with the QUERY DSPACE command.

Information about a Definable Subset of all Data Spaces
SYSDEF SVC offers support to retrieve information about a subset of all data spaces in the system. Due to changing data space usage situations in the system, the returned information is of unpredictable length. The information is made up of data records, which are collected in the user supplied work area and passed back to the user. In case, the user supplied area does not suffice, a return code of X'04' is passed to the user.
On return, the field DSPSDDAD contains the address of the records
Every returned data record has a layout as defined by DSPQRYDS. This DSECT is defined in macro MAPQRYDS. It contains the following information:

- Name of the Data Space
- SYSLOG ID of the owning partition
- SYSLOG ID of the accessing partition
- An indication for the scope of the data space
- An indication, via which access list the data space is accessed
- Maximum size of data space
- Actual size of data space

One record is created for every data space and every accessing partition of this data space as long as this data spaces matches the defined subset. Definable subsets are:

- All data spaces in the system
- All data spaces owned and/or accessed by a specified partition
- All data spaces with a specified name
- All data spaces with a specified name owned by a specified partition

For data spaces with SCOPE=COMMON, only one record is created even though every partition accesses the data space. This record contains blank characters as the SyslogId of the accessing partition.

To allow a sorting of the records before displaying them, together with the data records, a string with all existing SYSLOG IDs is passed to the caller. This string is actually used by JCL/AR for use with the QUERY DSPACE command. On return to the caller, the address of this string resides in DSPSDDA2 and its length is stored into DSPSDDL2.

Information about all Data Spaces in the System

The largest subset is defined, if ALL information is requested. An indication for this option must be set in DSPSDPRM. No additional input information is required. Note, that in the case of a large number of data spaces and/or partitions, a large number of records is created. In order to prevent a return code of X'04', you should supply a sufficient work area.

Information about all Data Spaces Owned and/or Accessed by a Specified Partition
The number of records can be reduced by restricting the search to just one partition. In this case, only these records are returned, which have the specified partition as either the owner or the accessor of a data space. To choose this option, set the corresponding bit in DSPSDPRM and store the SYSLOG ID of the partition of your choice into DSPSDDLG. Specifying a non-existing partition will lead to a return code of X'00', but of course no records will be returned.

Information about all Data Spaces with a Specified Name
If the name of the data space, that you need information about, is already known, then you can restrict the search to only the data spaces that match that name. To choose this option, set the according bit in DSPSDPRM and store the name of the data space into DSPSDDNM. Note that there may be multiple data spaces with the same name in one system. Via this option, all of these data spaces are selected.

Information about the Data Spaces with a Specified Name Owned by a Specified Partition
A further restriction can be done by specifying the name of the dataspace as described above and additionally specify the SYSLOG ID of the owning partition in DSPSDDOW. The owner indication in DSPSDPRM must be set as well.

SVC 123 (X'7B' - SDUMP(X))
The SDUMP and SDUMPX macros provide a dump of virtual storage which contains user data and system data into a dump library or to SYSLST. It dumps address ranges in the primary address space and it dumps storage ranges in address or data spaces to which addressability via an ALET or via an STOKEN exists.
On entry to the SVC, the parameter list address is loaded into register 1.

If the program is in primary ASC mode, SDUMP or SDUMPX can be used. Otherwise, wenn the program runs in access register(AR) mode, SDUMPX must be used. SDUMPX provides all of the function of SDUMP but generates code and address that are appropriate for AR mode.

The SDUMP macro cannot dump data space storage. To dump data space storage, SDUMPX is to be used instead by including either the LISTD or SUMLISTL parameter.

Note: The SDUMP macro supports only keyword parameters.
All parameters of this macro are compatible with the parameters of the MVS SDUMP macro. VSE will ignore the parameters and values which are not significant for VSE.

SVC 124 (X'7C' - PRODEXIT)
The PRODEXIT macro offers services to:

- maintain vendor exits and
- activate the specified vendor exits in the corresponding components (vendor exit hooks)

Parameter Function Requested

DEFINE PRODEXIT DEFINE defines a vendor exit of a given class and subclass. The DEFINE service returns an PRODEXIT token, that allows to use PRODEXIT ENABLE, DISABLE and DELETE services.

ENABLE This macro enables the exit defined by PRODEXIT DEFINE for exit activation. Depending on the SCOPE definition, the exit is either enabled - system wide (scope = system), - for the current partition (scope = partition), or for the current task only (scope = task). Multiple ENABLE requests can be given for scope = partition/task.

DISABLE This macro disables the exit defined by PRODEXIT DEFINE from exit activation. Depending on the SCOPE definition, the exit is either disabled - system wide (scope = system), - for the current partition (scope = partition), or for the current task only (scope = task). Multiple DISABLE requests can be given.

DELETE This macro disables the exit (if not yet done) and deletes the exit definition. PRODEXIT DELETE will wait on PRODEXIT RETURN, if the exit is active.

Note: DELETE out of an vendor exit is not allowed.

RETURN Any vendor exit has to return via PRODEXIT RETURN. If there is an other vendor exit enabled for the same subclass, then control will be given to it. If there is no more exit enabled, control will be given back to the corresponding component.

CHECK The CHECK service checks, if any vendor exit has to be activated for the given class/subclass and current task. This service can only be used in problem state programs (PSTATE). The CHECK service may be used to skip the preparation for the exit input, when no exit is available or enabled for the given class/subclass.

ACTIVATE The VSE component has to use the PRODEXIT ACTIVATE service ority to call the vendor exits for the according subclass. It receives control again after **all** vendor exits of the subclass are processed, that is after the PRODEXIT RETURN service issued by the last vendor exit.

Register Use:

Register 0 on input additional function code for ENABLE and DISABLE. On return additional return code during DEFINE if GETVIS was failing (RF=20).
Register 1 pointer to input for DEFINE, CHECK and ACTIVATE
Register 1 and 2 input for ENABLE, DISABLE and DELETE (PRODEXIT TOKEN)
Register 15 function code on input, return code

Cancel Conditions:

Cancel Code Cause

X'21' either macro expansion corrupted, i.e. an unknown or illegal function code passed on PRODEXIT, or invalid TOKEN passed on PRODEXIT ENABLE, DISABLE or DELETE

X'25' parameter list and/or area passed on PRODEXIT DEFINE, CHECK and ACTIVATE not in valid address range.

X'2E' Deadlock situation during re-occupation of LTA detected during PSTATE exit RETURN processing

X'47' Reason Code 3 - second vendor exit invocation invalid detected during CHECK and ACTIVATE processing

For the macro description see “PRODEXIT” on page 109.

SVC 124 (X'7C' - PRODID)

Provides supervisor support for an interface for vendor programs

Parameter Function Requested

DEFINE notifies the system about a product. The product description passed as input is stored in a table. A token is returned to the program in its input area. This token is used in the functions AUTH and DELETE.

AUTH=YES/NO grants or revokes authority to use certain system services by issuing task. AUTH=YES sets on bit TCBVEND in the field TCBFLAG3, AUTH=NO sets it off.

CHECK retrieves notification entry of one or more products

DELETE deletes notification entry, invalidates token, resets authority of issuing task if still on. If there are any vendor exits ENABLED and/or DEFINED for that vendor, the exits will be deleted too.

Register Use:

Register 0 on input additional function code. On return additional return code (DEFINE) or number of matching entries (CHECK)
Register 1 pointer to input for define, delete, check
Register 1 and 2 input for AUTH=
Register 15 function code on input, return code

Cancel Conditions:

Cancel Code Cause

X'0B' user starting the program issuing DEFINE has no update right for phase IJBVEND.

X'21' either macro expansion corrupted, i.e. an unknown function code passed on PRODID, or invalid token passed on PRODID AUTH or DELETE.

X'25' parameter list and/or area passed on PRODID DEFINE, DELETE or CHECK not in valid address range.

For the macro description see “PRODID” on page 165.

SVC 125 - 129 (X'7D'-X'81')

SVC 130 (X'82')

SVC 130 (X'82') is used by BAM to force that a task, which invoked BAM services in AMODE 31 is canceled and a message is issued.

SVC 131 (X'83')

SVC 131 (X'83') is used for ported OS/390 SVC based services to simulate the corresponding OS/390 SVC (see “Supervisor Call Simulation (SIMSVC)” on page 309.).

SVC 132 (X'84')

SVC 132 (X'84') is used for ported OS/390 SVC based services to simulate the corresponding OS/390 SVC.
It is used for services, that require an OS/390 emulation environment (see “Supervisor Call Simulation (SIMSVC)” on page 309.).

SVC 133 (X'85')

Interface between JCL and the supervisor.
SVC 133 (X'85') is issued by JCL to initialize and pass control to the application.

SVC 134 - 140 (X'86' - X'8C')

Reserved.

SVC 141 (X'8D' - VSIUCV)

Generic VSE interface to the VM-function IUCV (Inter User Communication Vehicle).

The SVC X'8D' is used by VSE applications to establish or end IUCV communication with other VSE or VM applications.

The SVC 141 (X'8D') performs the following functions:

SSTE Give supervisor state to an application called 'VCNA'. Is preserved for compatibility only, since the former VM/VCNA product is not supported any more. Other applications should use the MODESET macro to obtain supervisor state.

OPEN Inform VSE that the corresponding application is a potential VSE IUCV user.

CONN Establish a connection between the application and another IUCV user via VSE.

CLOS Stop usage of IUCV by a VSE application and delete all connections related to this application.

SEVR Delete a connection between a VSE application and another user of IUCV.

ACPT Accept a connection request to the issuing VSE application, issued by another IUCV user.

The handling of all IUCV related events, SVCs as well as external interrupts, are managed by means of the Application and Path ID Tables (see Figure 40 and Figure 41 on page 308).

IUCV Application ID Table Entry (DSAIDENT)

DEC	HEX	Label	Description
0– 7	0– 7	DSAIDNME	Application ID name
8–11	8– B	DSAIDEXT	Exit address for application
12–15	C– F	DSAIDTIB	TIB pointer of exit owner
16–17	10–11	DSAIDPIK	PIK of application
18–19	12–13		Reserved

Figure 40. Formats of IUCV Application ID Table Entry

IUCV/APPC/VM Path ID Table Entry (VMDSPID)

DEC	HEX	Label	Description
0– 1	0– 1	DSPIDID	Path ID
2	2		Reserved
3	3	DSPIDSW	Path ID table entry switch
		PTHINACT	X'00' Path is inactive
		PTHACTVE	X'04' Path is active
		PTHCCTIS	X'08' CONNECT issued for this path
		PTHCCTRV	X'0C' CONNECT received for this path
4– 7	4– 7	DSPIDAIP	Address of application ID table entry
8–11	8– B	DSPIDDAT	Data passed to Exit routine
12–19	C–13	DSPIDTGT	Target name

Figure 41. Format of IUCV Part of Path ID Table Entry

SVC 142 - 149 (X'8E' - X'95')

Reserved.

SVC 150 (X'96')

CICS SVC.

SVC 150 is used by CICS to pass control to a routine that executes in supervisor state (with rid=usertid). Functions, that require supervisor state are called by CICS during SVC 150 processing (e.g. cross memory services). During execution of SVC 150, CICS relies on the OS/390 control block structure. Whenever CICS issues SVC 150, the supervisor updates the OS/390 control block structure. before passing control to the CICS routine. Nested SVC 150 processing is supported. When CICS leaves one nesting level, the OS/390 control block structure is updated accordingly.

SVC 151 - 255 (X'97'-X'FF')

Reserved.

Supervisor Call Simulation (SIMSVC)

A macro, called SIMSVC, is implemented to simulate OS/390 services in the VSE/ESA environment. The macro is placed just before the SVC instruction of the OS/390 API.
The macro will

- check the execution environment (OS/390 or VSE/ESA), that is the CVT flags defining the VSE/ESA environment are used
- if the API has to be executed on a VSE/ESA system, an SVC with a SVC code (x'83' or x'84') will be executed.
- the OS/390 SVC is used as a function code

During SVC x'83' and SVC x'84' processing, a second SVC table, which contains the supported OS/390 SVCs, is used.
The SIMSVC expansion depends on the libraries used for compilation.

- when the OS/390 API library is used it expands into SVC x'84'
- without the OS/390 API library it expands into SVC x'83'

The following OS/390 SVCs are supported:

- SVC 1 (X'01' - WAIT)
- SVC 2 (X'02' - POST)
- SVC 4 (X'04' - GETMAIN)
- SVC 5 (X'05' - FREEMAIN)
- SVC 9 (X'09' - DELETE)
- SVC 10 (X'0A' - GETMAIN/FREEMAIN)
- SVC 11 (X'0B' - TIME)
- SVC 13 (X'0D' - ABEND)
- SVC 18 (X'12' - BLDL)
- SVC 34 (X'22' - MGCRE)
- SVC 35 (X'23' - WTO/WTOR)
- SVC 42 (X'2A' - ATTACHX)
- SVC 44 (X'2C' - CHAP)
- SVC 46 (X'2E' - STIMERM/TTIMER)
- SVC 47 (X'2F' - STIMER)
- SVC 48 (X'30' - DEQ)
- SVC 56 (X'38' - ENQ)
- SVC 62 (X'3E' - DETACH)
- SVC 87 (X'57' - DOM)
- SVC 107 (X'6B' - MODESET)
- SVC 109 (X'6E' - ESPIE)
- SVC 119 (X'77' - TESTAUTH)
- SVC 120 (X'78' - GETMAIN/FREEMAIN)

- SVC 122 (X'7A' - LOAD)

Appendices

This publication contains appendixes as follows:

Index

L

M

N

O

P

R

S

T

U

V

W

X