File No. S360-36
Form C28-6812-0

44PS

IBM Systems Reference Library

IBM System/360 Model 44 Programming System Supervisor Call (SVC) Functions

The IBM System/360 Model 44 Programming System provides functions for input/output operations, interruption handling, program phase loading, and inter- and intra-program communication. Source programs written in the assembler language can request these functions directly from the programming system's supervisor by using the Supervisor Call (SVC) instruction.

This publication contains detailed information for coding the assembler language sequences necessary to call these functions. To use this publication, the programmer should be familiar with the functions of the programming system and with the assembler language, as described in the following IBM publications:

<u>IBM System/360 Model 44 Programming System: Concepts and Facilities</u>, Form C28-6810

<u>IBM System/360 Model 44 Programming System: Assembler Language</u>, Form C28-6811

PREFACE

This publication supplies detailed information for writing the assembler language sequences required to use the Supervisor Call (SVC) functions provided by the System/360 Model 44 Programming System.

The publications cited on the cover describe the Model 44 Programming System and its assembler language. Machine characteristics of the Model 44 are described in the publications:

IBM System/360: System Summary, Form A22-6810

IBM System/360: Principles of Operation, Form A22-6821

IBM System/360 Model 44: Functional Characteristics, Form A22-6875

Publications on related topics (e.g., input/output devices) are listed in the publication IBM System/360 Bibliography, Form A22-6822.

Detailed specifications for using the system (e.g., coding control statements, job deck setup) will be supplied in subsequent publications.

First Edition

Significant changes or additions to the specifications contained in this publication will be reported in subsequent revisions or Technical Newsletters.

This publication was prepared for production using an IBM computer to update the text and to control the page and line format. Page impressions for photo-offset printing were obtained from an IBM 1403 Printer with a 120-character chain containing upper and lower case letters, special characters, and numerals.

Copies of this and other IBM publications can be obtained through IBM Branch Offices.

A form for readers' comments appears at the back of this publication. It may be mailed directly to IBM. Address any additional comments concerning this publication to the IBM Corporation, Programming Systems Publications, Department D39, 1271 Avenue of the Americas, New York, N.Y. 10020.

TABLES

A supervisor call is a request by a problem program for the system to provide a specific service. The Model 44 Programming System provides several different types of services that are available for use by a problem program in this manner.

The system's routines will, for example, perform all input/output services for a program. They may be used to load other programs or program segments from the phase library, store information in the user communication region, or provide special program routines that are entered only when and if certain conditions develop.

The program requests one of these services by executing a Supervisor Call (SVC) instruction. The assembler instruction SVC 4 causes the system to read a block of data from an input data set. For program clarity, this instruction may be written

```
          SVC READ
```

if, elsewhere in the program, there is the assembler instruction

```
     READ EQU 4.
```

The examples in this publication follow this format.

Execution of an SVC instruction causes an interruption. The system stops executing the problem program and branches to a supervisor routine that examines the SVC operand to determine what service is being requested. It then branches again to one of its specialized routines designed to provide the requested service.

The system retains control until the request is satisfied, except for input/output operations. In the latter case, control returns to the problem program, unless the program has specified otherwise, so that it can continue processing while the input/output operation is in progress. The system automatically regains control when an input/output interruption needs to be processed.

The program, in a sense, never knows that it was interrupted. It picks up exactly where it left off when the interruption occurred, except where specified otherwise in the discussions of the individual supervisor calls. These discussions specify what information the program must provide to enable the system to satisfy the request and they note what changes are made in the contents of the general registers and main storage as a result.

REGISTER CONVENTIONS

Certain register conventions must be observed. Register 1 is used to pass the address of a parameter list that contains data the system needs to satisfy a problem program's request. Register 14 is used for return addresses to which certain routines must branch when their work is done. Register 0 is used in a few cases to pass information.

Execution of any supervisor call destroys the contents of register 15. In most cases, when the problem program regains control, this register contains a return code indicating whether the system was able to supervisor call properly. This code is in the low order byte, and the rest of the register is set to zeros. At other times, the contents of this register are unpredictable.

NAME CONVENTIONS

The names of the supervisor calls, such as READ, UPSAND, STXITC, are used for reference only. They have no special meaning to the assembler or the system. They attain meaning only when equated to a numeric value and used in a supervisor call, as indicated previously.

GROUPINGS

The full list of supervisor calls and a brief description of each is given in Table 1. The supervisor calls are grouped for individual discussion in this publication as follows:

The input/output section covers SVC 0 - SVC 11. These are broken into subgroups of OPEN and CLOSE for label processing, READ, WRITE, CHECK, NOTE, POINT, WEF, REWIND, and UNLOAD for input/output operations at the system's read/write level, and EXCP and WAIT for input/output at the execute channel program level.

FETCH and LOAD tell how to load and enter programs and program segments from the phase library. EOJS and CANCEL tell how to end a job step or job.

INSERT, EXTRACT, UPSAND, UPSOR are used to store and retrieve information from the system's user communication region.

Timer services are available through use of GETIME, SETIME, STXITC, and RTXITC. Finally, the discussion of STXIPC and RTXIPC tells how a program may handle many of its own program check conditions.

Table 1. Supervisor Calls

Mnemonic	Code	Description
EXCP	SVC 0	EXCP (execute channel program) is used by programs that provide their own input/output routines.
WAIT	SVC 1	WAIT is used to delay further execution of a program until an EXCP operation has been completed.
OPEN	SVC 2	OPEN is used for tape label processing and repositioning of data sets before the data sets are used in a program.
CLOSE	SVC 3	CLOSE is used for label processing of data sets after they have been used in a program.
READ	SVC 4	READ is used to transmit a block of data from an input data set to an area of main storage.
WRITE	SVC 5	WRITE is used to transmit a block of data from an area of main storage to an output data set.
CHECK	SVC 6	CHECK is used to delay further execution of a program until an input/output operation, other than one initiated by EXCP, has been completed.
NOTE	SVC 7	NOTE is used to determine the current position of a data set.
POINT	SVC 8	POINT is used to reposition a data set.
WEF	SVC 9	WEF (write end-of-file mark) is used to write an end-of-file mark in an output data set on magnetic tape.
REWIND	SVC 10	REWIND is used to reposition a magnetic tape to its load point.
UNLOAD	SVC 11	UNLOAD is used to rewind and unload a magnetic tape volume.
FETCH	SVC 12	FETCH is used to load a program or program segment into main storage from the phase library and transfer control to it.
LOAD	SVC 13	LOAD is used to load a program or program segment into main storage from the phase library.
EOJS	SVC 14	EOJS (end-of-job step) is used to terminate a job step.
CANCEL	SVC 15	CANCEL is used to terminate a job.
GETIME	SVC 16	GETIME (get time) is used to obtain the time of day from the system timer.
INSERT	SVC 17	INSERT is used to store information in the system's user communication region.

(continued)

Table 1. Supervisor Calls (continued)

Mnemonic	Code	Description
EXTRACT	SVC 18	EXTRACT is used to obtain the location of the user communication region.
UPSAND	SVC 19	UPSAND (user program switch AND) is used to obtain the logical product (AND) of the user program switch byte and the low-order byte of register 1.
UPSOR	SVC 20	UPSOR (user program switch OR) is used to obtain the logical sum (OR) of the user program switch byte and the low-order byte of register 1.
STXIPC	SVC 21	STXIPC (set exit for program check interruption) is used to notify the system that the program contains a special routine to be entered when certain types of program check interruptions occur.
STXITC	SVC 22	STXITC (set exit for timer interruption) is used to notify the system that the program contains a special routine to be entered when a timer interruption occurs.
SETIME	SVC 23	SETIME (set interval timer) is used to set a value in the system's interval timer.
RTXIPC	SVC 24	RTXIPC (program check interruption return) is used at the end of a program check interruption routine to return control to the interrupted main program.
RTXITC	SVC 25	RTXITC (timer interruption return) is used at the end of a timer interruption routine to return control to the interrupted main program.

INPUT/OUTPUT SUPERVISOR CALLS

There are two levels of input/output operations. The first is the read/write level in which a problem program invokes system routines to perform all input/output operations. The system sets up the needed instructions, executes the operation, and applies standard interruption analysis and error recovery techniques. This level provides device independence in that the same problem program coding is used regardless of whether the device is a magnetic tape unit, direct access storage device, card read/punch, or printer.

The second level is the execute channel program (EXCP) level in which the problem program provides its own input/output routines. The EXCP level can be used with devices not supported by the system or to manipulate devices in a manner not provided by the read/write level routines. Device routines written and tested at the EXCP level may subsequently be incorporated into the read/write level through reassembly of the supervisor.

Both levels may be used within a single program.

This section deals in detail with the read/write functions before discussing EXCP operations. The OPEN and CLOSE supervisor calls, which are used for label processing, are used at both levels but are discussed only in the read/write section.

READ/WRITE OPERATIONS

The read/write level supervisor calls are READ, WRITE, NOTE, POINT, CHECK, WEF, REWIND, and UNLOAD. A program requiring an input/output operation issues one of these supervisor calls and provides certain information in parameters and control blocks so that system routines can execute the operation.

The system uses three types of control blocks. They are:

- Request control blocks, which contain general reference information relevant to an individual input/output request.
- File control blocks, which contain information pertaining to a particular data set and the volume in which it resides.
- Unit control blocks, which contain information about a specific input/output device.

The formats for these blocks are shown in the appendix. A problem program at the read/write level may refer to the information in any of them, but in general it is concerned only with the request control block.

The problem program defines a 40-byte request control block for each data set referred to by its input/output requests. The program also must provide one item of information within the block, the SYSUNI index value in hexadecimal for the system unit being used in the operation. This value can be obtained from Table 2. The system obtains all other information required for the block from various sources and handles the block's remaining construction and maintenance.

One of the parameters accompanying each input/output request in a program is the address of a request control block. The system then uses the information in this and other blocks to prepare channel commands, schedule and execute the requested operation, analyze the results for errors or abnormal conditions, such as end of file, and post a return code so the program can determine how the operation turned out.

A return code is saved in the request control block until the next time the program requests an input/output operation using the same block. The system examines the code to see how the previous operation terminated. If the termination was abnormal, the system places the code in register 15, resets the code area of the request control block to 00, and returns control to the problem program. The new request is ignored, but since the control block's return code area has been cleared, the request may be reissued.

It is up to the problem program to decide what to do about the abnormal termination. Such conditions are noted when the system detects a permanent transmission error, an end-of-file or end-of-extent indication, or an apparently-valid request that could not be executed, such as READ for a printer. Table 3 contains a list of the possible return codes and their meanings.

Table 2. System Unit (SYSUNI) Index Values

System Unit	Hexadecimal Code	Decimal Code
SYSAB1	01	1
SYSAB2	02	2
SYSREL	03	3
SYSLOG	04	4
SYSRDR	05	5
SYSIPT	06	6
SYSLST	07	7
SYSOPT	08	8
SYSPCH	09	9
SYSPSD	0A	10
SYSDMY	0B	11
SYSUAS	0C	12
	0D	13
reserved	0E	14
	0F	15
SYS000	10	16
SYS001	11	17
SYS002	12	18
.		
.		
.		
SYS009	19	25
SYS010	1A	26
.		
.		
.		
SYS015	1F	31
SYS016	20	32
SYS017	21	33
.		
.		
.		
SYS200	D8	216

Table 3. Input/Output Return Codes

Hexadecimal Code	Description
00	Normal return
04	Operation dependant: a. An end-of-file mark was detected during the last READ operation. b. An end-of-extent condition was detected during the last WRITE operation. The data set is positioned just prior to the block that causes the overwrite.
08	A permanent transmission error occurred during the last operation. The data set is positioned just past the block containing the error.
0C	The last operation was terminated without transmitting any data. The position of the data set is not known.
10	The last operation was terminated because of an invalid request, such as a READ request for a printer.
14	The last operation was terminated with an incorrect length condition. This applies only to a READ or WRITE that does not invoke incorrect length suppression.

A program should not attempt to determine how an operation turned out by looking into the control block to examine the code. The operation may still be in progress, and the code would not have been set. A program that must know how an operation turned out before it resumes processing can invoke the CHECK function.

When CHECK is invoked, control does not return to the problem program until the operation has been completed and a return code is posted in register 15. The control block return code area is cleared, and the next request is accepted regardless of whether the previous operation terminated abnormally.

The following paragraphs show what considerations are necessary when a program makes an input/output request.

When the supervisor call is issued, the system examines register 1 for the address of a parameter list. This list contains the address of a request control block and may also contain, depending on the type of operation, the address of an input/output buffer and the address of a count of the number of bytes to be transmitted.

The system examines the request control block to determine how the last operation associated with it terminated.

- If the last operation terminated normally (or was followed by a CHECK), the system proceeds with processing the new request.
- If the last operation still is in progress, the system waits until it terminates and has been analyzed before proceeding with the new request.
- If the last operation terminated abnormally (and this condition was not detected by a CHECK), the new request is ignored. The system places the abnormal termination code in register 15, resets the control block return code area to 00, and returns control to the problem program. It is up to the problem program to act on the abnormal termination condition and determine whether the new request should be reissued.

An abnormal condition code in one request control block does not affect operations using other request control blocks. The system treats each block independently, even if two or more blocks refer to the same data set.

If the last operation terminated normally or the request is reissued, the system determines whether the required channel and device facilities are available. It also examines the new request and its parameters for errors.

If the required facilities are free, the system continues processing the request. If they are in use, the system queues the new request and returns control to the problem program. The system will automatically resume processing the request when the facilities are available for it.

If the system discovers errors in the parameters of the new request, it posts an error code in the request control block and returns to the problem program. This code is detected by the CHECK or by the next request using the same block.

When the required facilities are free and the request has reached the top of the queue, the system constructs the required channel program and starts the operation. This channel queue method is designed to permit maximum use of all available channel facilities while providing the greatest amount of overlapped processing.

Termination of an operation causes an input/output interruption that is analyzed by the system. Standard error recovery methods are applied, when necessary, and a return code is posted in the request control block.

BLOCKING

A program must handle its own buffering, blocking, and unblocking. All blocks within a data set must be the same length. The system's read/write routines transmit one block at a time, although a program, when reading, may elect to have only a partial block put into its buffers. These capabilities are covered in more detail in the discussions of each individual supervisor call.

Programs using a 2315 disk cartridge obtain greatest efficiency by writing blocks that are either one, two, four, or eight sectors in length. Calculations of block sizes also should note that the system uses the first five bytes of each sector, when blocks are one sector or less in length, for the addresses of the track and any alternate track. When blocks are more than one sector long, these five-byte address fields are written only in the first sector for the block.

CONTROL CHARACTERS

A program may specify in its control statements that the first byte of each block contains a control character for physical manipulation of printers and unit record devices. This byte must contain the bit configuration for the American Standards Association code for the desired carriage or stacker select operation.

The proper ASA codes are as follows:

b (blank)	Space one line before printing
0	Space two lines before printing
-	Space three lines before printing
+	Suppress space before printing
1	Skip to channel 1
2	Skip to channel 2
3	Skip to channel 3
4	Skip to channel 4
5	Skip to channel 5
6	Skip to channel 6
7	Skip to channel 7
8	Skip to channel 8
9	Skip to channel 9
A	Skip to channel 10
B	Skip to channel 11
C	Skip to channel 12
V	Select card pocket 1
W	Select card pocket 2

If the device is one to which the characters do not apply, the byte is treated as part of the data record.

As an alternative to using one of the ASA codes, a program may provide the actual command code portion of the channel command. A program indicates in job control statements that it is doing this. In such cases, the first byte in the data block is used as the first byte of the channel command.

ATTENTION INTERRUPTIONS

An attention interruption occurs when the operator presses the request key on the Console Printer Keyboard. This attention condition is not recognized immediately if it would interfere with the proper functioning of an input/output operation that is in progress. The interruption is queued until it can be accepted and processed without destroying essential information.

When an attention interruption is recognized at the read/write level, the system examines word 2 of the appropriate unit control block for the address of a special attention input/output block permanently resident in main storage. This address is moved to word 3 of the unit control block, and the system branches to the device routine address given in the attention input/output block to read the message typed on the keyboard. This read operation is handled in the same manner as a regular READ request.

A program at the EXCP level must duplicate these functions when an attention interruption occurs, or it may ignore the interruption by returning control to the system in the same manner as for a device end interruption.

LABEL PROCESSING SUPERVISOR CALLS

The OPEN and CLOSE supervisor calls are used to prepare data sets for processing and to ensure their proper disposition when processing is completed.

Although the main purpose of OPEN and CLOSE is to process labels, they should be used even with unlabeled data sets. All direct access data sets, for example, contain block count and extent information that must be updated by the system.

The system routines for executing OPEN and CLOSE are present in main storage only while needed. They are loaded each time an OPEN or CLOSE supervisor call is issued. Processing time can be saved by taking advantage of their capabilities to open or close more than one data set with a single supervisor call.

The applicable codes are:

OPEN - SVC 2

CLOSE - SVC 3

OPEN - SVC 2

The OPEN supervisor call is used to verify or create magnetic tape data set header labels and reposition tape, unit record, and direct access data sets.

Each installation may specify when reassembling the system supervisor whether its standard is to use volume and data set labels on magnetic tapes. If its standard is not to use them, OPEN does not check for their presence or composition.

When the standard is to use labels, OPEN processes both the volume label and the data set header label. All volumes should be labeled prior to their use in a problem program. If a tape volume is unlabeled, any data set on it is treated as unlabeled. Direct access volumes must be labeled.

If an input data set on magnetic tape is being opened, the system checks the volume and data set header labels against data in the program's job control statements. Some of this information is entered into the file control block for the system unit containing the data set for use by the OPEN routines.

For an output data set on tape, the system checks the volume label, compares the expiration date in the data set header label with the current date, and, if requested, creates a new data set label.

Direct access volume and data set labels are examined or created by the job control processor when it processes the ACCESS or ALLOC control statements.

A data set is repositioned if this service is requested in the parameters accompanying the supervisor call.

Labeled magnetic tape data sets are repositioned to a point between the end of the label and the start of the data set.

An unlabeled magnetic tape is repositioned to its load point.

Repositioning of a direct access data set is to the origin of the data set or, in the case of a directoried data set, to the origin of the desired member.

When repositioning of a unit record data set is requested, the current block count in the system unit's file control block is reset to 0.

When OPEN is executed, register 1 must contain the address of a parameter list. This list consists of one full-word entry for each data set to be opened. Each entry is aligned on a full-word boundary and contains the address of a four-byte control information area. The first byte of the last entry in the parameter list must contain the hexadecimal code 80, indicating the end of the list.

The four bytes of control information are specified in hexadecimal as follows:

```
CONTRL   DC   XL4'uurrppcc'
```

where

uu	is the SYSUNI index value of the system unit containing the data set. This value can be obtained from Table 3.
rr	= 00 if the data set is not to be repositioned. = 01 if repositioning is wanted.
pp	= 00 if an input data set is being opened. = 01 if an output data set is being opened.
cc	is a byte reserved for use by the system if a return code is needed to notify the program of errors.

After OPEN has been executed, control returns to the instruction following the SVC. The low-order byte of register 15 contains the hexadecimal code 00 if there were no errors. If register 15 contains 04, indicating errors, the cc field for the applicable system unit contains one of the following codes:

Hexa-decimal Code	Description
01	The system cannot find the data set. For example, the uu field may contain an erroneous entry pointing to a system unit for which no file or unit control blocks exist. The data set is not opened.
02	This code indicates that either a volume label or a data set header label is missing. It appears only when the installation standard is to use labels and the program contains a LABEL control statement. Information from the LABEL statement is entered in the system unit's file control block, and the absence of labels is noted so no attempt will be made to verify a label when CLOSE is given. For an output data set, this code indicates the absence of both the volume and the header label. In either case, the data set is opened.

After returning the error code, the system takes no further notice of these conditions.

If a more serious condition is detected, such as a permanent read error, the system writes a message requesting operator intervention. The operator has the option of continuing or cancelling the job.

Following is an example of the use of the OPEN supervisor call.

This example assumes three data sets are to be opened, two input data sets on SYSIPT and SYS001 and an output data set on SYSOPT. The names of the data sets or data set members needed for the program have been specified in control statements. Repositioning of SYS001 and SYSOPT is desired.

```
OPEN     EQU  2
         .
         .
         .
         LA   1,PARAM

         SVC  OPEN

         BAL  14,ANALYS
         .
         .
         .
         DS   0F

PARAM    DC   A(CTLIPT)

         DC   A(CTL001)

         DC   X'80'

         DC   AL3(CTLOPT)

CTLIPT   DC   XL4'06000000'

CTL001   DC   XL4'11010000'

CTLOPT   DC   XL4'08010100'
```

In this example, the Branch and Link (BAL) instruction is given to branch to an analysis routine as soon as the system has finished executing OPEN. ANALYS is not a system routine. It represents a routine that should be included in a problem program. It examines the return codes, takes any necessary action, and returns to the main program by branching to the address in register 14.

CLOSE - SVC 3

The CLOSE supervisor call is used to ensure proper disposition of data sets that no longer are needed by the program.

With CLOSE, a program can

- Verify or create standard trailer labels in magnetic tape data sets.
- Reposition a tape volume, unit record, or direct access data set.
- Disconnect a symbolic unit to protect the contents of its data set.

For a labeled input tape data set, the trailer label is verified against header label and control statement data. The block count in the trailer label is compared with the count of blocks read from it by the program, and any discrepency is indicated through return codes.

For an output tape data set, an end-of-file mark is written and a trailer label is created.

For an output direct access data set, the block count in its label is updated, if necessary. This updating is not performed physically, however, until the end of the job step. If the data set is reopened before the end of the job step, it should be closed again.

Repositioning is the same as for OPEN, except that magnetic tapes are rewound to load point.

CLOSE disconnects a data set in the sense that the system treats subsequent input/output requests for the system unit as invalid. A magnetic tape volume, in addition, is rewound and unloaded.

When CLOSE is executed, register 1 must contain the address of a parameter list. This list consists of one full-word entry, aligned on a full-word boundary, for each system unit to be closed. Each entry contains the address of a four-byte control information area. The first byte of the last entry contains hexadecimal 80, signifying the end of the parameter list.

The control information for each system unit is specified in hexadecimal as follows:

```
    CONTRL    DC    XL4'uurrppcc'
```

where

uu	is the SYSUNI index of the system unit to be closed.
rr	= 00 if the data set is not to be repositioned. = 01 if the block counts of direct access and unit record data sets are to be reset to 0 or a magnetic tape is to be rewound to load point. = 02 if the system unit is to be disconnected. A magnetic tape also is rewound and unloaded.
pp	= 00 to close an input data set and to verify an input trailer label on magnetic tape. = 01 to close an output data set and update a direct access label, write an end-of-file mark, or create a magnetic tape trailer label. = 02 if no end-of-file mark is to be written, and no label processing is desired.
cc	is a byte reserved for use by the system if a code is needed to identify errors.

To facilitate device independence, the system ignores control codes that do not apply to the device being used.

After CLOSE has been executed, control returns to the instruction following the SVC. The low-order byte of register 15 contains the hexadecimal code 00 if there were no errors. This code is 04 if any of the designated system units could not be closed, and the cc field of its control word contains one of the following codes:

Hexa- decimal Code	Description
01	The system cannot find the data set. The data set is not closed.
02	Reserved for future use.
04	An input magnetic tape data set contained a standard header label but does not contain a standard trailer label. The data set is closed.
08	The block count in an input magnetic tape data set trailer label differs from the count of blocks read from it by the program. The data set is closed.

If a more serious condition, such as permanent write error is detected, a message is written, and the operator has the option of continuing or canceling the job.

READ/WRITE LEVEL SUPERVISOR CALLS

The five primary input/output supervisor calls at the read/write level are READ, WRITE, NOTE, POINT, and CHECK. These supervisor calls enable a program to read and write data, reposition a data set, and ensure that a data transmission terminated satisfactorily. A general discussion of their functioning was given at the beginning of this section.

The applicable SVC codes are:

READ	- SVC 4
WRITE	- SVC 5
CHECK	- SVC 6
NOTE	- SVC 7
POINT	- SVC 8

The parameters for each of these supervisor calls must include the address of a request control block. Each request control block referred to in a program must be defined by the program as a 40-byte area. The program also must provide the SYSUNI index for the system unit involved. This is illustrated in the examples accompanying the discussion of the individual supervisor calls.

READ - SVC 4

The READ supervisor call is used to transmit one block of data from an input data set to an area of main storage.

When READ is executed, register 1 must contain the address of a parameter list. This list consists of three full words, each aligned on a full-word boundary. The first word contains the address of a request control block. The second word contains the address of the area in main storage where the input data is to go. The third word contains the address of a four-byte count field, which also must be aligned on a full-word boundary.

The count field has the format xx00yyyy, where xx is an incorrect length suppression code and yyyy is a count of the number of bytes to be transmitted. When the read request is executed, yyyy bytes are read from the input data set into the area specified by the second word in the parameter list.

The incorrect length condition occurs when the count field specifies a byte count that differs from the block size of the input records. The system reads one block at a time. If the program specifies a count smaller than the block size, only the number of bytes specified by the program are transmitted. If the count is larger than the block size, only one block is transmitted. In either case, an incorrect length condition occurs. This condition is ignored if the xx byte of the count field is hexadecimal 20. If the first byte is 00, an incorrect length code is posted in the request block. This causes an abnormal condition return the next time an input/output request uses this block.

The following sample coding could be used to read a 360-byte block from an input data set on system unit SYS004. Any incorrect length condition is suppressed.

```
READ     EQU   4
SYS004   EQU   20
         DS    0F
RCB01    DC    AL1(SYS004)
         DC    39X'00'
IOBUFF   DS    90F
         .
         .
         .
         LA    1,PARAM
         SVC   READ
         BAL   14,ANALYS
         .
         .
         .
ANALYS   BC    15,CKCOD(15)
CKCOD    BC    15,0(14)
         BC    15,EOF
         BC    15,RDERR
         BC    15,NODTMT
         BC    15,IRQST
         .
         .
         .
         DS    0F
PARAM    DC    A(RCB01)
         DC    A(IOBUFF)
         DC    A(COUNT)
COUNT    DC    X'2000'
         DC    H'360'
```

At the opening of this example, SYS004 is equated with 20, its SYSUNI index hexadecimal value that goes into the first byte of the request control block, and the word READ is equated with its proper SVC code 4. This is done solely for ease in programming. In either case, the actual numeric value could be used at the required places in the subsequent coding.

The next steps set up a request control block and define an input buffer. The addresses of these two areas are the first two entries in the parameter list PARAM at the bottom of the example. The count field is set up to read 360 bytes and a 20 prefix to suppress any incorrect length condition.

The first of the executable instructions loads the address of the parameter list into register 1. The READ supervisor call follows. When the system returns control to the problem program, it will branch to the ANALYS routine to analyze the return code in register 15 to determine how the previous input/output operation on the same request control block turned out. The Branch and Link instruction puts a return address in register 14 to facilitate a return to this point when the ANALYS routine is finished.

ANALYS is not a system routine. It represents the sort of error-checking a problem program should do.

The ANALYS routine uses the return code in register 15 as an index for another branch. If the code is 00, the branch is to the next instruction which returns control to the main program. Otherwise, the branch is to one of the specialized routines provided elsewhere in the program to handle each of the four abnormal conditions.

When the abnormal condition routines are entered, register 1 still contains the address of the parameter list which contains the address of the request control block involved in the abnormally terminated operation. The routine may examine this and the file control block to determine what should be done.

WRITE - SVC 5

The WRITE supervisor call is used to transmit a block of data from an area of main storage to an output data set.

When WRITE is executed, register 1 must contain the address of a parameter list. This list consists of three words aligned on full-word boundaries. The first word contains the address of a request control block. The second word contains the address in main storage of the block of data to be written. The third word contains the address of a four-byte count field which must be aligned on a full-word boundary.

The count field has the format xx00yyyy, where xx represents an incorrect length suppression code, as for READ, and yyyy represents, in hexadecimal, the number of bytes to be written.

The incorrect length code applies when the program is working with a device that has set physical limits. An incorrect length condition is created, for example, by an attempt to write a block larger than 80 bytes on a card punch. This condition also can be caused by an attempt to write a block larger or smaller than other blocks in the data set.

A program may not write outside the previously defined limits of a direct access data set or data set member. An attempt to do this causes an end-of-extent abnormal return condition.

In other respects, WRITE functions and is programmed in the same general manner as a READ supervisor call.

CHECK - SVC 6

The CHECK supervisor call is used to delay further execution of a problem program until a requested read/write level operation has been completed.

CHECK may be issued at any time following any other read/write level supervisor call except OPEN and CLOSE. Its parameters include the address of the request control block used in the requested operation. Control does not return to the user until all pending operations involving the block have been completed. Before returning, the system posts the return code for the last completed operation in register 15 and resets the return code area of the request control block to 00 so that another operation request can be accepted immediately.

When CHECK is executed, register 1 must contain the address of a parameter list. This list consists of one word, aligned on a full-word boundary, containing the address of a request control block. The parameter list may be the same list used for the operation to which the CHECK applies.

Control returns to the instruction following the CHECK SVC. Register 15 contains 00 if no abnormal conditions were detected in the operation. A 00 return code also appears if the request control block has not been used or if it was examined previously by CHECK and there has been no intervening operation. If the current operation terminates abnormally, register 15 contains one of the non-zero return codes shown in Table 3.

Here is an example of how CHECK may be used.

```
SYS003   EQU  19

WRITE    EQU  5

CHECK    EQU  6

         DS   0F

RCB01    DC   AL1(SYS003)

         DC   39X'00'

IOBUFF   DS   90F
         .
         .
         .
         LA   1,PARAM

         SVC  WRITE

         SVC  CHECK

         BAL  14,ANALYS
         .
         .
         .
ANALYS   BC   15,CKCOD(15)
         .
         .
         .
         DS   0F

PARAM    DC   A(RCB01)

         DC   A(IOBUFF)

         DC   A(COUNT)

COUNT    DC   X'2000'

         DC   H'360'
```

This example calls for writing a block of 360 bytes in the data set on system unit SYS003. The incorrect length condition is suppressed. The WRITE supervisor call is followed immediately by a CHECK. This combination causes the system to delay further execution of the program until the write operation has been completed and analyzed and a return code has been posted in register 15. The abnormal return code section of the request control block has been reset to 00, and the block is ready to accept another request.

This example uses the branch to the ANALYS routine to examine the return code in register 15, as did the READ example.

A combination of operations that would result in an error code not being examined should be avoided. If, for example, the three operations, WRITE, WRITE, CHECK, all use the same request control block, and the first WRITE terminates abnormally, the second WRITE is rejected. An error code is placed in register 15, and the error section of the request control block is cleared. When CHECK is issued, the system finds no operation in progress and no error code in the request control block, so it resets register 15 to zero and returns to the main program with no indication of the abnormal termination. This condition could be avoided by using CHECK after each WRITE or by entering an analysis routine between the second WRITE and CHECK.

NOTE - SVC 7

The NOTE supervisor call is used to determine the current position of a data set.

NOTE generally is used with the POINT supervisor call, SVC 8, for nonsequential processing of data sets. NOTE causes the system to provide the number of the next block within a data set. The number given is based on the position of the block within the data set or data set member and not on the number of blocks that have been read or written by the program.

When NOTE is executed, register 1 must contain the address of a parameter list. This list consists of two full words aligned on full-word boundaries. The first word contains the address of a request control block. The second word contains the address of a four-byte area where the system is to place the block count. This area also must be aligned on a full-word boundary. The count is stored in it as a hexadecimal value. Control returns to the instruction following the SVC.

The system determines the block count after all pending input/output operations for the data set have been completed. Control returns to the instruction following the SVC.

Register 15 contains the hexadecimal code 00 if no abnormal conditions were detected in the request control block. If the previous operation terminated abnormally and was not checked, the NOTE request is ignored and register 15 contains a non-zero return code.

POINT - SVC 8

The POINT supervisor call is used to reposition a data set to a specified block.

POINT generally is used with the NOTE supervisor call, SVC 7, for nonsequential processing of data sets. POINT causes the system to reposition a data set to a place immediately following a specified block. If, for example, a block count of 3 is given, the data set is positioned so that the next block that would be read by a READ request is the fourth block.

When POINT is executed, register 1 must contain the address of a parameter list. This list consists of two full words aligned on full-word boundaries. The first word contains the address of a request control block. The second word contains the address of a four-byte area which also is aligned on a full-word boundary. This area contains the desired block count, expressed in hexadecimal.

For example, to reposition a data set so that the next block to be read is the fourth block within the data set (or data set member), the following might be used:

```
SYS003   EQU  19

POINT    EQU  8

         DS   0F

RCB02    DC   AL1(SYS003)

         DC   39X'00'
         .
         .
         .
         LA   1,PARAM

         SVC  POINT

         BAL  14,ANALYS
         .
         .
         .
PARAM    DC   A(RCB02)

         DC   A(BLKCT)

BLKCT    DC   F'3'
```

The data set is repositioned to a point between its third and fourth blocks. Control returns to the instruction following the SVC. Register 15 contains the hexadecimal code 00 if no abnormal conditions were noted in the request control block. If the previous operation terminated abnormally and was not checked, the POINT request is ignored and register 15 contains a non-zero return code.

As an example of using NOTE and POINT together, a program may issue a NOTE supervisor call to determine the current position of a data set. The program then determines the number of blocks to be skipped or backspaced, modifies the block count parameter (BLKCT in the previous example) accordingly, and issues a POINT supervisor call that uses the same parameter list as the NOTE.

EXTENDED TAPE VOLUME SUPERVISOR CALLS

The WEF, UNLOAD, and REWIND supervisor calls are used by the system's OPEN and CLOSE routines. They are available for general use, but problem programs should use the OPEN and CLOSE supervisor calls to obtain the same results. Use of these functions instead of OPEN or CLOSE may cause accidental loss of data or malfunctioning of system routines in later operations on the same data sets. They should not be used for direct access volumes.

The applicable codes are:

WEF	-	SVC 9
REWIND	-	SVC 10
UNLOAD	-	SVC 11

WEF - SVC 9

The WEF supervisor call is used to write an end-of-file mark on magnetic tape volumes.

An end-of-file mark indicates the end of a data set. When a program subsequently reads this data set and detects this mark, an abnormal return condition is recorded in the request control block.

When WEF is executed, register 1 must contain the address of a parameter list. This list consists of one word, aligned on a full word boundary, containing the address of a request control block.

Control returns to the instruction following the SVC. The low-order byte of register 15 contains the hexadecimal code 00 if no abnormal conditions were noted in the request control block. If the previous operation terminated abnormally and was not checked, register 15 contains a non-zero return code and the WEF request is ignored.

REWIND - SVC 10

The REWIND supervisor call is used to reposition a magnetic tape volume to its load point.

Parameter specifications and return codes are the same as for WEF.

UNLOAD - SVC 11

The UNLOAD supervisor call is used to rewind and unload a magnetic tape volume.

Parameter specifications and return codes are the same as for WEF.

EXECUTE CHANNEL PROGRAM

The execute channel program supervisor calls, EXCP and WAIT, are used to initiate execution of a problem program's own input/output routines.

The EXCP level permits a program to work with devices that are not supported by the system and to manipulate devices in ways not provided by the read/write level routines. For a list of supported devices, see IBM System/360 Model 44 Programming System: Concepts and Facilities, Form C28-6810. Routines written and tested at the EXCP level may subsequently be incorporated into the system's read/write level through reassembly of the supervisor.

EXCP level operations make use of both system and user routines. System routines schedule the requested operation and execute the privileged instructions required for input/output. They also handle some of the input/output interruptions that do not require the attention of an EXCP level program.

At the EXCP level, the problem program supplies the channel commands and a device-dependent routine that performs initialization, interruption, and error recovery functions. The system enters the problem program's device routines in the same way that it enters its own at the read/write level. The problem program's routines communicate with the system through control blocks and return codes.

REQUIREMENTS

To execute an operation at the EXCP level, a problem program should provide:

- A channel program consisting of one or more channel commands
- A device routine
- An input/output block

A channel program consists solely of the commands required to execute the desired operation. These commands may be constructed before the EXCP supervisor call is given, or they may be constructed or updated by the device routine which the system enters as part of its execution of the EXCP call.

The system enters the device routine first to set up the channel program. The same routine is entered again after execution of the program for interruption analysis and error recovery. This first entry is skipped if the channel program was ready before the EXCP was issued.

A device routine is able to determine why it was entered by examining byte 0 of word 3 of the unit control block for the device being used. This byte contains 00 when the routine is entered to prepare the channel program. At the time of the post-execution entry, this byte contains a non-zero value to indicate the type of interruption.

A device routine must not contain any supervisor calls. It operates in the supervisor state and cannot be interrupted. Any interruptions that would have occurred during its execution, except for machine check and certain types of program check, are queued and handled later.

An input/output block consists of the first six words of a request control block. It replaces the request control block for EXCP level operations and must be aligned on a full-word boundary.

The format of the input/output block is shown in Figure 1. Its fields are as follows:

Word	Byte	Description
0	0	SYSUNI index, supplied by programmer. (See SYSUNI index table, Table 2.)
0	1-3	Device routine address, supplied by programmer.
1	4	Post request flag indicating in hexadecimal code whether the block currently is active, supplied by the system. 00 - no operation pending 01 - operation in progress
1	5-7	Address of channel command list, supplied by the programmer.
2	8	Reserved for system use.
2,3	9-15	Last seven bytes of Channel Status Word, supplied by the system at interruption time.
4	16-19	Sense information, supplied by the system when a unit check condition occurs.
5	20-23	Four EBCDIC characters to be used by the system to locate an error recovery program in the phase library. This is an optional field supplied by the programmer.

```
                                                                   Word
┌──────────────┬─────────────────────────────────────────┐
│0             │ 1-3                                     │
│ SYSUNI       │                  Device Routine Address │   0
│              │                                         │
├──────────────┼─────────────────────────────────────────┤
│4             │ 5-7                                     │
│ Post Request │                 Channel Program Address │   1
│      Flag    │                                         │
├──────────────┼─────────────────────────────────────────┤
│8             │ 9-11                                    │
│              │                                         │
│ reserved     │                                         │   2
├──────────────┴─────Channel Status Word─────────────────┤
│12-15                                                   │
│                                                        │
│                                                        │   3
├────────────────────────────────────────────────────────┤
│16-19                                                   │
│                       Sense Data                       │   4
│                                                        │
├────────────────────────────────────────────────────────┤
│20-23                                                   │
│ Name of interruption analysis program to be loaded     │   5
│                                                        │
└────────────────────────────────────────────────────────┘
```

Figure 1. Input/Output Block Format

When a block is created, all fields that are not filled by the program should be defined as hexadecimal zeros.

EXECUTION

When the EXCP supervisor call is issued, register 1 must contain the address of an input/output block. The system examines this block to determine whether it contains a valid SYSUNI index and the addresses of a device routine and channel program.

If the SYSUNI index specifies a system unit for which no unit control block exists, control returns with a 04 hexadecimal code in register 15. If the index refers to a device that is not operational, control returns with a 08 code in register 15. In either case, the system treats the requested operation as completed.

If the SYSUNI entry is valid but the requested channel facilities are busy, the system queues the request and returns control to the problem program. If the EXCP is followed by a WAIT supervisor call, the system delays further execution of the program until the EXCP operation is completed.

When the required channel facilities are available, the system examines the input/output block for a channel program address.

If an address is present, the system starts execution of the channel program.

If there is no channel program address, the system places the address of the appropriate unit control block in register 1, the system return address in register 14, and enters the device routine. When the device routine is ready to return control to the system, it branches to the address in register 14. The addresses in registers 1 and 14 must be unchanged when the system regains control.

As indicated previously, the device routine can determine why it was entered by examining byte 0, word 3 of the unit control block. This byte at this time should contain 00, indicating that the device routine was entered to prepare a channel program.

The device routine prepares the channel program and places its address in the input/output block. It also must identify the type of operation being requested by putting a return code in register 15.

Code 04 tells the system the operation will keep the device busy but free the channel immediately. This might be a seek command on a direct access storage device.

Code 08 means the operation will tie up both the channel and the device, as a data transmission operation would do.

Code 00 means that no operation is to be performed, and the system should treat the request as completed.

The routine returns to the system by branching to the address in register 14.

The operation now is at the same point as it would have been if the channel program had been prepared in advance. The system commences execution. Unless WAIT is in effect, control returns to the problem program until there is an input/output interruption.

INTERRUPTION PROCESSING

Execution of an EXCP request may generate two or more input/output interruptions. When each occurs, the system scans the Channel Status Word to determine the cause. Some conditions, such as channel end not accompanied by device end, are handled entirely by the system.

The system re-enters the problem program's device routine if the interruption is one of the following types:

Type	UCB Request Flag
Device End	04
Program Controlled	08
Attention	0C

The UCB request flag is byte 0, word 3 of the unit control block which now contains one of the non-zero values listed above. As before, on entry to the device routine, register 1 contains the address of the unit control block, and register 14 contains the system return address.

The device routine may examine the request flag byte to determine the cause of the interruption. The three bytes following the request flag contain the address of the input/output block associated with the operation.

If a device end interruption is accompanied by a unit check condition, the system stores sense data from the channel and unit in word 4 of the input/output block. A unit check condition noted at any other time causes the system to delay entry of the problem program's device routine until a device end interruption occurs.

The Channel Status Word is stored in words 2 and 3 of the input/output block.

A device routine initiates error recovery operations, when necessary, in the same manner as a regular request. It places a 04 or 08 return code in register 15 and branches to the address in register 14. As before, a 00 return code tells the system the operation is finished.

The system does not permit other requests to disturb a volume while error recovery procedures are in progress.

The device routine may keep a specialized interruption analysis and error recovery program in the phase library for use only when needed. The device routine instructs the system to load and enter such a program by putting a 0C return code in register 15 and branching to the address in register 14. Word 5 of the input/output block must contain the name of the desired program.

The program name is specified in the input/output block as four EBCDIC characters. This name must be entered in the phase library directory in the format xxxxbbbb, where xxxx represents the four EBCDIC characters and followed by four blanks.

The program is loaded into the transient area of main storage. The system treats it as a logical extension of the device routine. It operates in the supervisor state and may not issue any supervisor calls. It uses the same return codes as a device routine and returns to the system through the address in register 14. This address and the unit control block address in register 1 must be unchanged.

A program controlled interruption differs from device end in its use of return codes. If, after a program controlled interruption, the device routine returns to the system with a non-zero return code in register 15, the system assumes the operation still is in progress. If the register contains the 00 code, the system treats the operation as completed. It resets the post request flag in the input/output block to 00, returns control to the problem program, and treats the channel and device as free.

An attention interruption is handled in the same manner as device end.

EXCP - SVC 0

When EXCP is executed, register 1 must contain the address of an input/output block, which must be aligned on a full-word boundary. Control returns to the instruction following EXCP. The low-order byte of register 15 contains hexadecimal 00 if the request was accepted. Register 15 contains 04 if no unit control block exists for the requested SYSUNI, or 08 if the requested device is not operational.

WAIT - SVC 1

WAIT instructs the system to delay further execution of the problem program until a requested EXCP operation has been completed.

Completion is indicated by a hexadecimal 00 in the post request flag byte of the input/output block.

When WAIT is executed, register 1 must contain the address of the input/output block used in the EXCP operation. When the operation is completed, control returns to the instruction following WAIT.

FLOW CONTROL SUPERVISOR CALLS

The flow control supervisor calls are FETCH and LOAD. They are used to load programs and program segments from the phase library.

The applicable codes are:

FETCH - SVC 12

LOAD - SVC 13

PHASE LIBRARY

The phase library is the source of all programs and program segments executed under system control. Each phase resident in this library has been edited into absolute form by the system's linkage editor. It represents a segment of code that is to be loaded into main storage at one time. It may be anything from an entire program to a specialized subroutine shared by several installation programs.

The IBM-supplied processors and transient supervisor functions reside permanently in this library. User-written phases (programs and program segments) may reside in it permanently or solely for the duration of a single job. All additions to the library, permanent or temporary, are made through the linkage editor.

The library includes a directory that contains an entry for each phase currently resident in the library. Each entry contains an 8-character EBCDIC name of the phase it identifies. (Names of less than eight characters are padded on the right with blanks.) Directory entries also include information on where the phase is located in the library, its main storage loading address, and its main entry point. The system uses this directory information to find, load, and enter a phase unless a programmer provides an alternate load address in his supervisor call parameters.

FETCH and LOAD are not used to load relocatable routines from the module library. These are incorporated into a program during linkage editing.

FETCH - SVC 12

The FETCH supervisor call is used to load a program or program segment from the phase library into main storage and transfer control to it.

The system uses FETCH to load and enter a user program. This user program may then invoke FETCH to load and enter phases of a program that would not all fit into main storage at one time. FETCH might also be used when the choice of subsequent phases to be loaded depends upon conditions that develop during execution of the current program phase. FETCH is designed so that the calling routine can pass one full word of information to the called routine.

A phase called by FETCH must be loaded and entered at the addresses specified in its phase library directory entry. If other load or entry points are desired, the LOAD supervisor call should be used.

When the FETCH supervisor call is executed, register 1 must contain the address of a parameter list. This list consists of either one or two full words, aligned on full-word boundaries. If there are two words, the system assumes the second word is a parameter to be passed to the new phase. Before the system transfers control to this new phase, it places the contents of this second word into register 1.

The first byte of the first word in the parameter list is a hexadecimal code, either 00 or 80. The 00 code tells the system the list consists of two words, and the second word is to be put into register 1 for use by the new phase. The 80 code indicates that the list consists of only one word and there is no information to be passed.

The remaining three bytes of this word contain the address of an 8-byte location aligned on a full-word boundary elsewhere in main storage. This location must contain the name of the desired phase in EBCDIC characters. If the name is less than eight characters, it must be padded on the right with blanks.

For example, to load and enter a phase named PROGNAME, the following coding could be used:

```
FETCH    EQU  12
              .
              .
              .
         LA   1,PARAM

         SVC  FETCH
              .
              .
              .
         DS   0F

PARAM    DC   X'80'

         DC   AL3(LOC)

LOC      DC   C'PROGNAME'
              .
              .
              .
```

The system searches the phase library for an entry named PROGNAME, loads it at the address specified in its directory entry, and enters it at the address specified in its directory entry. On entry to the phase, register 1 contains binary zeros since there was no parameter to be passed. Register 15 contains the address of the entry point. Other registers are unchanged.

If the system is unable to find and load the phase correctly, the job is cancelled.

To load and enter the same phase as in the previous example and pass to it the address of a parameter list, the following coding could be used:

```
FETCH    EQU  12
              .
              .
              .
         LA   1,PARAM

         SVC  FETCH
              .
              .
              .
         DS   0F

PARAM    DC   A(LOC)

         DC   A(LIST)

LOC      DC   C'PROGNAME'

LIST     (list of parameters or other
            information)
```

Since the first byte of the first word in this parameter list is 00, the system loads the phase, as before, but changes the contents of register 1 before entering it. On entry, register 1 contains the address of LIST, since this was the second word in the parameter list. The values in register 15 are the same as in the previous example. Other registers are unchanged.

LOAD - SVC 13

The LOAD supervisor call is used to load a program segment from the phase library into main storage.

LOAD causes the system to search the phase library directory for an entry for the desired phase. Upon finding it, the system loads the phase into main storage, places the address of its main entry point in register 1, and returns control to the calling routine. The phase is loaded either at the address specified in its directory entry or at an alternate address specified in the parameters accompanying the supervisor call. The entry point address that is placed in register 1 is obtained from the directory entry and is modified, if necessary, to reflect any change in loading address.

When the LOAD supervisor call is executed, register 1 must contain the address of a parameter list. This list consists of either one or two full words, aligned on full-word boundaries. If there is a second word, the system assumes the phase is to be loaded at an address other than the one specified in the phase's directory entry. The alternate address must be within the problem program area.

The first byte of the first word in the parameter list is a hexadecimal code, either 00 or 80. The 00 code tells the system the list consists of two words, and the second word contains the address of another full-word, aligned on a full word boundary, that contains the loading address. The 80 code indicates that the list consists of only one word and the phase is to be loaded at the load address in its directory entry.

The remaining three bytes of the first word contain the address of an 8-byte location elsewhere in main storage. This location must be aligned on a full-word boundary and must contain the name of the desired phase in EBCDIC characters. If the name is less than eight characters, it must be padded on the right with blanks.

For example, to load a phase named SBRUTINE, the following coding could be used:

```
LOAD     EQU  13
              .
              .
              .
         LA   1,PARAM

         SVC  LOAD
              .
              .
              .
         DS   0F

PARAM    DC   X'80'

         DC   AL3(LOC)

LOC      DC   C'SBRUTINE'
              .
              .
              .
```

The system finds SBRUTINE, loads it at the address specified in its directory entry, and returns control to the instruction following the SVC. On return, register 1 contains the address of the phase's main entry point. The low-order byte of register 15 contains the hexadecimal code 00, indicating no errors. If the system is unable to find the name of the desired phase in the phase library directory, register 15, on return, contains the code 04. Other registers are unchanged.

The same phase could be loaded at a different location by use of the following coding:

```
LOAD     EQU  13
              .
              .
              .
         LA   1,PARAM

         SVC  LOAD
              .
              .
              .
PARAM    DC   A(LOC)

         DC   A(ALP)

LOC      DC   C'SBRUTINE'

ALP      DC   A(loading address)
              .
              .
              .
```

This sample coding causes the system to find the desired phase and load it at the alternate address specified at location ALP. Control returns, as before, to the instruction following the SVC with the address of the phase's main entry point in register 1. The return codes in register 15 are the same as in the previous example. Other registers are unchanged.

TERMINATION SUPERVISOR CALLS

The termination supervisor calls are EOJS and CANCEL. They can be used for either normal or abnormal termination of programs and program segments.

The applicable codes are:

EOJS - SVC 14

CANCEL - SVC 15

EOJS - SVC 14

The EOJS supervisor call is used to terminate a job step.

EOJS should be the last instruction in every job step. It instructs the system to load the job control processor to read control statements and commence execution of the next job step. The next job step may be part of the same program or the first job step in a new program.

A job step is entered by a standard linkage, so it also may be terminated by a standard return sequence.

A program should check before issuing EOJS to ensure that no input/output operations are pending. Completion of an input/output operation after EOJS has been executed could cause malfunctioning of the job control processor.

CANCEL - SVC 15

The CANCEL supervisor call is used to terminate a job.

When a CANCEL supervisor call is executed, the system terminates the current job immediately. A message for the operator is written, and a dump is taken if a dump was requested in the job's control statements. The system then loads the job control processor which reads the system input unit, ignoring all statements until a /& end-of-job control statement is detected.

The communication region supervisor calls are INSERT, EXTRACT, UPSAND, and UPSOR. They are used for communication between a problem program and the system's user communication region.

The applicable codes are:

INSERT	- SVC 17
EXTRACT	- SVC 18
UPSAND	- SVC 19
UPSOR	- SVC 20

USER COMMUNICATION REGION

The user communication region is an area within the system supervisor that may be used both by system programs, such as the assembler, and problem programs. Programs may read information from this area, but must use supervisor calls to insert or alter information to avoid accidental destruction of system data.

Its contents are as follows:

Word	Byte	Description
0,1	0-4	Date, set by the operator, in the form yyddd.
2	8-11	Address of the first byte of the problem program area.
3	12-15	Address of the last byte available for use by the problem program.
4	16-19	Address of the highest byte in the problem program area filled by a phase loaded by means of any FETCH or LOAD supervisor call.
5	20-23	Address of the last byte in the problem program area filled by the most recent FETCH or LOAD supervisor call.
6,7	24-31	Job name in EBCDIC characters.
8,9	32-39	Job step name in EBCDIC characters.
10	40	User program switch byte. This byte is set to zeros whenever the system reads a JOB control statement.
10	41	Highest assembly error severity. Reset to zero by JOB statement. 00 - Normal. No errors. 04 - Warning messages listed. Execution should be successful. 08 - Error messages listed. Execution may fail. 0C - Severe errors. Execution impossible. 10 - Terminal errors. Job has been cancelled.
10	42,43	Not used.
11	44-47	User interprogram communications area. This area may be used by one job step to preserve information for use by a later job step. This area is set to zeros whenever the system reads a JOB control statement.
12,13	48-55	User intraprogram communications area. This area may be used by one job step phase to preserve information for use by another phase within the same job step. This area is set to zeros whenever the system reads an EXEC control statement initiating a new job step.
14-31	56-127	Up to six 8-byte EBCDIC option parameters from the job step EXEC statement are stored here by the job control processor. If less than six parameters are stored, the area is padded on the right with blanks. The area is reset to blanks

Word	Byte	Description
		before the beginning of the next job step. The full 72 bytes of this area and the 8-byte intraprogram communication area also are used by the system and processor programs between job steps for temporary storage of certain control statements.
32-35	128-143	Up to 16 bytes of accounting information are stored here for use by installation routines. This information, in EBCDIC form, is obtained from JOB and/or EXEC statements.
36-xx	144-xxx	Data generated by installation accounting routines may be stored here. The upper limit of the area is determined by the installation when the resident supervisor is re-assembled. This field is not included in the IBM distributed system. As distributed, the user communication region occupies 36 words, bytes 0-143.

INSERT - SVC 17

The INSERT supervisor call is used to store information in the user communication region.

The system does not require a problem program to provide any information in this area other than that contained in control statements. If a program does use the area, however, it should use the INSERT supervisor call to reduce the chances of accidental destruction of data needed by the system.

It is not necessary to know the location of the region to use INSERT. To refer to information already stored, the location can be determined by use of the EXTRACT supervisor call.

INSERT cannot be used to alter the contents of words 0 - 10, bytes 0 - 43 of the user communication region.

INSERT cannot be used to modify the user communication region permanently. The region is reinitialized when the initial program load procedure is executed.

When the INSERT supervisor call is executed, register 1 must contain the address of a parameter list. This list consists of two words aligned on full-word boundaries. The first word contains the address in the problem program area of the information to be stored in the user communication region. The second word contains the address of a 4-byte area containing control information.

The first byte of control information must be hexadecimal 00. The second byte gives the number of 4-byte words to be stored in the region. The last two bytes indicate where in the region the data is to be stored. This last location is expressed in terms of the word where the system is to start storing the information, the first word in the region being word 0.

For example, to store eight bytes of data in the intraprogram communications area, words 12 and 13, bytes 48 through 55, the following coding could be used:

```
INSERT   EQU  17
              .
              .
              .
         LA   1,PARAM

         SVC  INSERT
              .
              .
              .

PARAM    DC   A(LOC)

         DC   A(CONTRL)
              .
              .
              .

LOC      (data to be stored)

         DS   0F

CONTRL   DC   X'00'

         DC   AL1(2)

         DC   AL2(12)
```

The system stores the eight bytes of information at LOC in the intraprogram communications area and returns control to the instruction following the SVC. The low order byte of register 15 contains the hexadecimal code 00, indicating no errors.

If an attempt is made to use INSERT to store information outside the communications region or in words 0 - 10, nothing is stored, and register 15, on return, contains the hexadecimal code 04.

Other registers are unchanged.

EXTRACT - SVC 18

The EXTRACT supervisor call is used to obtain the location of the communication region.

EXTRACT causes the system to put the address of byte 0 of the communication region in register 1 and return control to the calling program. The address of any particular word or byte within the region is obtained by adding the byte count to this value.

UPSAND - SVC 19 and UPSOR - SVC 20

The UPSAND and UPSOR supervisor calls are used to set or alter the contents of the user program switch byte in the user communications region. This byte is used for communication between job steps and within a job step.

When UPSAND is used, the logical product (AND) of the user program switch byte and the low-order byte of register 1 is stored in the user program switch byte.

When UPSOR is used, the logical sum (OR) of the user program switch byte and the low-order byte of register 1 is stored in the user program switch byte.

When the two bytes are combined by either UPSAND or UPSOR, they are matched bit for bit.

With UPSAND, if each of the corresponding bits is a 1, the result is a 1. If either is 0, the result is 0.

With UPSOR, if either of the ocrresponding bits is a 1, the result is a 1. If both are 0, the result is 0.

These combinations are illustrated in the following table:

A	B	UPSAND	UPSOR
1	1	1	1
1	0	0	1
0	1	0	1
0	0	0	0

When an UPSAND or UPSOR supervisor call is executed, register 1 must contain a comparison byte in its low-order positions, bits 24 through 31. The system alters the user program switch byte accordingly and returns control to the instruction following the SVC. No error codes apply.

For example, to set the user program switch byte to all 1's, the following coding could be used:

```
UPSOR    EQU  20

         IC   1,MASK

         SVC  UPSOR
              .
              .
              .

MASK     DC   X'FF'
```

As a result of this UPSOR supervisor call, byte 40 of the communications region is set to 11111111. This byte is reset to all 0's when the system reads a JOB control statement initiating another job.

CONDITIONAL INTERRUPTION SUPERVISOR CALLS

The primary purpose of the conditional interruption supervisor calls is to notify the system that the problem program is The problem program is able, for example, to inform the system that the program should be interrupted when a specified time interval elapses. A supervisor call gives the system the address of the routine that is to be entered. When the time runs out, the system interrupts the main program and enters the special routine. Execution of another supervisor call at the end of this routine returns control to the main program at the point where it was interrupted.

In addition to timer services, special routines can be provided for certain types of program check interruptions. Program check interruptions occur when the system detects a programming error.

These supervisor calls are in effect only during the job step in which they are issued.

The applicable codes are:

GETIME - SVC 16

STXIPC - SVC 21

STXITC - SVC 22

SETIME - SVC 23

RTXIPC - SVC 24

RTXITC - SVC 25

SAVE AREAS

A program that provides its own interruption routines must also provide save areas. A save area consists of 20 words (80 bytes) where the system can store the contents of the interrupted program's 16 general registers and program status word while the special routine is being executed. This information is restored when control returns to the interrupted program through execution of another supervisor call at the end of the special routine.

The system stores the contents of the interrupted program's registers immediately after the interruption occurs. When the program's interruption routine is entered, register 15 contains the address of the interruption routine, and register 13 con- program mode and is subject to the same interruptions as the main problem program.

Words 0 and 2 of the save area are used by the system for system information. The contents of register 13 are saved in word 1. Register 14, 15, and 0 through 12 are saved, in that order, in words 3 through 17. The last two words are for the old PSW.

It is possible for a timer interruption to occur, for example, while a program check interruption routine is being executed. The program, therefore, must provide separate save areas for each type of interruption that it handles. Otherwise, the occurrence of two interruptions at approximately the same time would destroy data needed to return to the interrupted main program.

Save areas and interruption routines must be located in the problem program area of main storage. Each save area must be aligned on a double-word boundary. If the problem program specifies a save area address that is not in the problem program area, the system does not accept the operation.

PROGRAM CHECK

STXIPC - SVC 21

RTXIPC - SVC 24

The STXIPC supervisor call informs the system of the addresses of a program's program check save area and the entry point of its program check interruption routine.

The RTXIPC supervisor call is used at the end of a program check interruption routine to restore the registers and return control to the interrupted main program.

A program check interruption occurs when the system detects a programming error, such as an incorrect operand, exceptional results, such as fixed-point overflow, or a violation of system or machine restrictions. When a program check interruption occurs, the system takes one of three possible actions:

1. Cancel. The job is terminated and a message to the operator describes the reason for termination.

2. Dump and Cancel. A message to the operator is written. A listing of the contents of all general registers and the problem program area of main storage is written, and the job is terminated. This dump must have been requested in the job control statements.

3. Transfer to a User Routine. If an STXIPC supervisor call has been executed prior to the interruption, the system can, in many cases, transfer to a program check interruption routine in the problem program instead of terminating the job. This routine is able to analyze the cause of the program check condition and also has the ability to return to the interrupted program.

The system does not transfer to a user's program check routine if the interruption is caused by one of the following conditions:

- Illegal operation code

- Privileged operation in the problem program state

- Addressing

In these cases, the system always cancels the job. A dump is written if it was requested.

If the interruption was not due to one of the foregoing conditions and an STXIPC supervisor call has been executed, the system stores the contents of the 16 general registers and PSW in the user's program check save area and enters the user's program check interruption routine. This routine may examine the registers and PSW and correct the program check condition, ignore it, or cancel the job. If this routine ends with an RTXIPC supervisor call, the system returns to the interrupted program at the point of interruption.

When an STXIPC supervisor call is executed, register 13 must contain the address of the user's program check save area. Register 1 must contain the address of the entry point of his program check interruption routine. These addresses can be changed only by execution of another STXIPC.

If the STXIPC specifies a save area address that is not within the problem program area, the system does not accept the operation. Control returns to the problem program with a hexadecimal 04 return code in register 15. If there are no errors and the operation is accepted, the return code in register 15 is 00.

If a program check condition develops while a program check interruption routine is being executed, the job is cancelled.

STXIPC expires at the end of a job step.

RTXIPC should be used only in routines that are entered because of a program check interruption occurring after execution of STXIPC.

When RTXIPC is executed, the system loads the general registers with the contents of the user's program check save area. Control returns to the interrupted program at the point of interruption.

TIMER SERVICES

GETIME - SVC 16

SETIME - SVC 23

STXITC - SVC 22

RTXITC - SVC 25

The system maintains two timer services. The first is the time of day, set by the operator as part of the initial program loading procedure. The second is an interval timer that may be used by a problem program to cause interruptions when a desired time interval expires.

The time, in both cases, is expressed in units of 1/19200 second.

TIME OF DAY

The GETIME supervisor call is used to obtain the time of day. Execution of GETIME causes the system to place a value in register 0. Multiplying this value by

19,200 converts it into the number of seconds since "midnight."

Control returns to the instruction following the GETIME supervisor call.

INTERVAL TIMER

The interval timer facility is used by the problem program to cause an interruption after a designated time period expires. This timer occupies a 32-bit word at location 80 in main storage.

A program inserts a value in the interval timer with the SETIME supervisor call. The value then is reduced every 1/50 or 1/60 of a second, depending on conditions at the individual installation. When the system is equipped with the High Resolution Timer feature, the value is reduced every 13 microseconds.

The interval timer requires 15.5 hours to go from its maximum value to 0. An external interruption occurs when the value changes from positive to negative. If the proper supervisor calls have been executed, the system enters the problem program's timer interruption routine. Otherwise the interruption is ignored.

A timer interruption routine is not entered immediately if the interruption occurs while the system is executing a routine in which interruptions are masked out. The system's input/output routines, for example, mask out all interruptions except machine check and program check. Other types of interruptions occuring during such periods are processed when the masked routine has been completed.

Other factors that may delay entry into a timer routine are pauses, when the system suspends processing pending some operator action, and execution of instructions that tie up the central processing unit for extended periods.

There are three supervisor calls for use with the interval timer.

SETIME is used to insert a value into the timer.

STXITC is used to notify the system that a timer interruption routine should be entered when the inserted value expires.

RTXITC is used at the end of a timer interruption routine to return to the interrupted program.

A SETIME supervisor call is ignored unless an STXITC supervisor call has been executed previously in the job step. The value inserted by SETIME must be the number of 1/19200 second intervals desired before a timer interruption occurs.

When the SETIME supervisor call is executed, register 1 must contain a timer value. This value is inserted in the interval timer, and control returns to the instruction following the SETIME supervisor call. This value can be changed at any time by means of another SETIME supervisor call. An interruption occurs if this value changes from positive to negative within the job step.

STXITC informs the system of the addresses of a program's timer interruption save area and the entry point of its timer interruption routine.

STXITC must be executed before SETIME, or SETIME is ignored.

When an STXITC supervisor call is executed, register 1 must contain the address of the entry point of the user's timer interruption routine. Register 13 must contain the address of the user's timer interruption save area. Control returns to the instruction following the STXITC supervisor call.

Register 15 contains the hexadecimal code 00 if there were no errors and the operation was accepted. A return code of 04 indicates that the program specified a save area address that was not within the problem program area and the operation was not accepted.

RTXITC should be used only in timer interruption routines.

When RTXITC is executed, the system restores the general registers from the save area. The program status word that was stored in the save area becomes the current PSW, and control returns to the interrupted program. Control returns to the point of interruption unless the instruction address portion of the old PSW has been changed by the timer interruption routine.

No attempt should be made to return to the interrupted program by means other than this supervisor call.

This section contains the formats of the control blocks used by the system's input/output routines at the read/write level.

At the read/write level, the file and unit control blocks are constructed and maintained by the system. The problem program must reserve space for the request control block and provide the SYSUNI index for its first byte.

REQUEST CONTROL BLOCK

The fields of the request control block are defined as follows:

Word	Byte	Description
0	0	SYSUNI index number of the system unit to be used in the input/output operation. This value can be determined from Table 2.
0	1-3	Address of the device dependent routine to be used to set up the channel commands and analyze interruptions.
1	04	Post request flag indicating whether the block currently is active: 00 = No operation pending; 01 = Operation in progress
1	5-7	Address of the first channel command required to execute the operation.
2	8	Reserved for system use.
2,3	9-15	Last seven bytes of Channel Status Word, stored when an operation is started and when an interruption occurs.
4	16-19	Sense information, stored when unit check condition occurs.
5	20-23	Name of program to be loaded from the phase library when necessary for interruption analysis.
6	24	Error recovery code identifying a type of error.
6	25-27	Counters used to keep track of number of attempts made to recover an error.
7	28	Return code 00 = Operation completed 04 = End of file or end of volume 08 = Permanent transmission error 0C = No data transmitted 10 = Invalid request 14 = Incorrect length
7	29-31	Address of file control block being used for this operation.
8	32	Request code identifying the type of operation to be set up by a device routine.
8	33-35	Address of buffer to be used for transmission.
9*	36	Incorrect length control byte 20 = Suppress incorrect length indication 00 = Check for incorrect length
9*	37-39	Number of bytes to be transmitted.

*Following completion of an input/output operation, the contents of word 9 are replaced by the updated data set position block count.

FILE CONTROL BLOCKS

The file control block for disk units is set up as follows:

Word	Byte	Description
0	0	Flag Byte 01 = Open 02 = Disconnected 04 = Modification 08 = Header checked 10 = Labeled 20 = Update VTOC 40 = Fresh data set 80 = Standard unit
0	1	Flag byte 80 = Control character

Word	Byte	Description
		40 = ASA control characters 20 = Write check 10 = Ignore (dummy data set)
0	2	Reserved
0	3	Number of blocks per track
1	4-7	Total number of blocks read or written in the data set
2	8,9	Logical record length
2,3	10-15	Seek and search address for the current operation
4	16	Number of block being processed
4	17	Number of bytes in key area (2311 only)
4	18,19	Maximum number of bytes per block
5	20,21	Block number of first block of member in a directoried data set
5,6	22-27	Device address of the first record in the data set
7	28-31	Number of the last block written
8	32-35	Number of blocks reserved for this data set.

Fields in the file control block for tape units are the same as for direct access devices, except as follows:

Word	Byte	Description
0	3	Setting of modifier bits in channel command, if applicable
2,3	10-15	Expiration date of data set
4	16,17	Reserved
5	20,21	Current file count, including trailer label file marks
5,6	22-27	Volume identification in EBCDIC
7,8	28-35	Data set name in EBCDIC

Fields for the file control block for other units are the same as for tape except that bytes 28-35 are reserved.

UNIT CONTROL BLOCK

Fields in the unit control block are defined as follows:

Word	Byte	Description
0	0,1	Physical device address
0	2	Device mode 01 = Burst mode 02 = Overrunable byte mode 03 = Non-overrunable byte mode
0	3	Type of unit 10 = 1052 Console Printer-Keyboard 20 = 2501 card reader 21 = 2540 card reader 28 = 2520P card punch 29 = 2540P card punch 2A = 1442P card punch 22 = 2520 read-punch 23 = 1442 read-punch 30 = 1403 printer 31 = 1403M7 printer 32 = 1443 printer 33 = 1443S printer 40 = 2400 magnetic tape 41 = 2400H magnetic tape 42 = 2400D magnetic tape 48 = 2400T7 magnetic tape 49 = 2400T7C magnetic tape 50 = Single disk storage drive 51 = 2311 disk
1	4	Relative chain pointer (e.g., activity, event chain).
1	5	Channel Scheduler flags 01 = Device busy 02 = Event in progress 04 = Attention waiting 08 = Device not ready 10 = Retry operation
1	6	Job control flags 80 = Device down 40 = System standard assignment 20 = Job control assigned 10 = Programmer assigned 08 = Assigned this step
1	7	Read/Write flags 01 = Volume has been mounted 02 = Volume label present 04 = Volume at EOV
2	8	Reserved
2	9-11	Address of input/output block used when an attention interruption occurs.

Word	Byte	Description
		(applicable only to those devices that can signal attention.)
3	12	Request flag for device routine 00 = Setup 04 = Device end interruption 08 = Attention interruption 0C = Program controlled interruption
3	13-15	Input/output block address associated with this operation
4	16-19	Current physical position of the device in terms of the cylinder and head positions for direct access devices and block count for sequential devices.
5	20	Reserved
5	21-23	Address of channel command word area associated with the device
6,7	24-31	Permanent and temporary error counters set and used by the system's device routines.

INDEX

C28-6812-0

Printed in U.S.A. C28-6812-0

IBM®

International Business Machines Corporation
Data Processing Division
112 East Post Road, White Plains, N.Y. 10601
[USA Only]

IBM World Trade Corporation
821 United Nations Plaza, New York, New York 10017
[International]

READER'S COMMENTS

Title: IBM System/360 Model 44 Programming System Supervisor Call (SVC) Functions

Form: C28-6812-0

Is the material:

	Yes	No
Easy to read?	---	---
Well organized?	---	---
Complete?	---	---
Well illustrated?	---	---
Accurate?	---	---
Written for your technical level?	---	---

How did you use this publication?

-----As an introduction to the subject

-----For additional knowledge

Other-------------------------

Please check the items that describe your position:

-----Customer personnel	----Operator	-----Sales Representative
-----IBM personnel	----Programmer	-----Systems Engineer
-----Manager	----Customer Engineer	-----Trainee
-----Systems Analyst	----Instructor	-----Other---------------

Please check specific criticisms, give page numbers, and explain below:

-----Clarification on pages
-----Addition on pages
-----Deletion on pages
-----Error on pages

Explanation:

If you wish a reply, be sure to include your name and address.

fold

fold

FIRST CLASS
PERMIT NO. 33504
NEW YORK, N.Y.

BUSINESS REPLY MAIL
NO POSTAGE NECESSARY IF MAILED IN THE UNITED STATES

POSTAGE WILL BE PAID BY . . .

IBM CORPORATION
1271 AVENUE OF THE AMERICAS
NEW YORK, N.Y. 10020

ATTENTION: PUBLICATIONS, DEPT. D39

fold

fold

IBM®

International Business Machines Corporation
Data Processing Division
112 East Post Road, White Plains, N.Y. 10601
[USA Only]

IBM World Trade Corporation
821 United Nations Plaza, New York, New York 10017
[International]