

The IBM logo, consisting of the letters "IBM" in a bold, sans-serif font, is positioned inside a solid black square.

Systems Reference Library

## OS Advanced Checkpoint/Restart

Release 21

This publication describes advanced checkpoint/restart, a technique for recording information about a job at programmer-designated checkpoints so that, if necessary, the job can be restarted at the beginning of a step or at a checkpoint within a step.

The checkpoint/restart manual describes how to take a checkpoint and how to restart a job. It also contains systems planning information. It is intended for use by programmers and system analysts who are familiar with the information presented in:

*OS Job Control Language Reference*, GC28-6704

*OS Data Management Services Guide*, GC26-3746



*Fifth Edition* (February 1972)

This publication, a part of OS Release 21, is a revision of, and makes obsolete, the fourth edition (GC28-6708-3) of *Advanced Checkpoint/Restart*. It applies to all subsequent releases unless otherwise indicated in new editions or technical newsletters.

Changes may be made to the information in this manual at any time; before using this publication in connection with the operation of IBM systems, consult the latest *SRL Newsletter*, GN20-0360, for the editions that are applicable and current.

Forms for reader's comments are provided at the back of this publication. If the forms have been removed, comments may be addressed to IBM Corporation, Programming Publications, Department D78, San Jose, California 95114.

For copies of this or any other IBM publication, see your IBM representative or call your local IBM Branch Office.

© Copyright International Business Machines Corporation 1968, 1969, 1970, 1972

## ABOUT THIS BOOK

This publication describes advanced checkpoint/restart, a technique for recording information about a job at programmer-designated checkpoints so that, if necessary, the job can be restarted at the beginning of a step or at a checkpoint within a step.

The major parts of this publication and the information in them are as follows:

- Chapter 1 describes in general terms checkpoint/restart and its components.
- Chapter 2 describes how to establish a checkpoint.
- Chapter 3 describes the restrictions that must be observed when a checkpoint is taken or a restart performed on user data sets.
- Chapter 4 describes how to request restart.
- Chapter 5 describes what the operator must do to authorize restart.
- Chapter 6 contains storage estimates.
- Chapter 7 contains miscellaneous information about checkpoint/restart.
- Appendixes A and B list completion codes and describe how to establish checkpoint at end-of-volume.

Advanced checkpoint/restart is intended for use by programmers and system analysts. A general understanding of job control language and data management is prerequisite knowledge for understanding the information in this book. See *OS Job Control Language Reference*, GC28-6704, and *OS Data Management Services Guide*, GC26-3746, for background information on these subjects.

The following publications are referred to in this book:

- OS COBOL Language*, GC28-6380, and *USA Standard COBOL*, GC28-6396, which contain information about the COBOL RERUN clause
- OS Data Management Macro Instructions*, GC26-3793, which contains information about coding DCBs
- OS Data Management for System Programmers*, GC28-6550, which contains information on preallocated data sets
- OS MFT Guide*, GC27-6939, or *OS MVT Guide*, GC28-6720, which contains information about the RESERVE macro instruction and creating or modifying a list of resident modules
- OS PL/I (F) Programmer's Guide*, GC28-6594, or the program product publications *OS PL/I Checkout Compiler General Information*, GC33-0003, and *OS PL/I Optimizing Compiler General Information*, GC33-0001, which contain information on how PL/I users can take a checkpoint and request restart
- OS Sort/Merge*, GC28-6543, which contains information about taking a checkpoint and requesting automatic checkpoint/restart when performing a sort with the sort/merge program
- OS Supervisor Services and Macro Instructions*, GC28-6647, which contains information about the list and execute forms of the CHKPT macro instruction
- OS Tape Labels*, GC28-6680, which contains information about tape labels



# CONTENTS

iii	<b>About This Book</b>
ix	<b>Summary of Changes for Release 21</b>
1	<b>Chapter 1: Introduction</b>
1	Types of Restart
1	Components of Checkpoint/Restart
1	CHKPT Macro Instruction
2	End-of-Volume Exit Routine
2	RD (Restart Definition) Parameter
2	RESTART Parameter
2	SYSCHK DD Statement
2	CKPTREST System Generation Specification
5	<b>Chapter 2: How to Establish a Checkpoint</b>
5	CHKPT Macro Instruction
6	Programming Notes on the CHKPT Macro Instruction
7	Exceptional Conditions
7	List and Execute Forms of CHKPT
7	Cautions in Taking a Checkpoint
7	Use of CHKPT with Other Macro Instructions
8	Use of CHKPT in Exit Routines
8	Explicit and Implicit Requests for ENQ
9	Use of Special Operating System Features
9	DCB for a Checkpoint Data Set
9	Required DCB Parameters
10	DCB Options
10	DD Statement for a Checkpoint Data Set
11	Use of Checkpoint Data Sets
11	How Checkpoint Entries Are Written
12	How to Ensure Restart
14	How Checkpoint Entries Are Identified
15	How to Use the CANCEL Option
17	<b>Chapter 3: User Data Sets</b>
17	What to Consider for Checkpoint/Restart
17	Cautions
19	Repositioning User Data Sets
20	Preserving Data Set Contents
21	Nonstandard Tape Labels
21	Input/Output Errors
22	What to Consider for Checkpoint or Step Restart
22	Generation Data Sets
23	Preallocated Data Sets
23	SYSIN Data Sets
23	SYSOUT Data Sets
24	SYSABEND Data Sets
25	<b>Chapter 4: How to Request Restart</b>
25	RD (Restart Definition) Parameter
26	RESTART Parameter
26	SYSCHK DD Statement

27	Automatic Restarts
27	Requirements for Automatic Restart to Occur
27	How to Request Automatic Step Restart
28	How to Request Automatic Checkpoint/Restart
28	JCL Requirements and Restrictions
29	Resource Variations Allowed in Automatic Restart
29	How the System Works at Automatic Restart
31	How MOD Data Sets Are Handled During Automatic Step Restart
31	Deferred Step Restart
31	How to Request Deferred Step Restart
32	JCL Requirements and Restrictions
33	Resource Variations Allowed in Deferred Step Restart
33	Deferred Checkpoint/Restart
33	How to Request Deferred Checkpoint Restart
34	JCL Requirements and Restrictions
36	Resource Variations Allowed in Deferred Checkpoint/Restart
37	How the System Works During Deferred Checkpoint/Restart
<b>39</b>	<b>Chapter 5: What the Operator Must Consider</b>
39	MVT Environment
39	Automatic Restart Message Sequence
40	Operator Options During Automatic Restart
42	Deferred Restart Message Sequence
42	Operator Considerations During Deferred Checkpoint/Restart
43	MFT Environment
43	Automatic Restart Message Sequence
44	Operator Options During Automatic Restart
46	Deferred Restart Message Sequence
46	Operator Considerations During Deferred Checkpoint/Restart
<b>49</b>	<b>Chapter 6: Storage Estimates</b>
49	Checkpoint/Restart Work Area
49	Checkpoint Data Set Storage Requirements
48	Resident Access Methods
50	Resident Checkpoint/Restart Module for MFT
<b>51</b>	<b>Chapter 7: Miscellaneous Information</b>
51	MVT Track Stacking
51	Job and Job Step Accounting and Checkpoint/Restart
52	MVT Job Step Time Limit
52	Completion of Step or Job Termination at System Restart
53	Reader Procedure Changes
53	COBOL RERUN Clause
53	Checkpoint/Restart and the Sort/Merge Program
53	PL/I Checkpoint/Restart Capability
54	TCAM Data Set Considerations
<b>55</b>	<b>Appendix A: Completion Codes</b>
55	Return Codes Associated with the CHKPT Macro Instruction
56	Completion Codes Issued by Checkpoint/Restart
<b>58</b>	<b>Appendix B: End-of-Volume Exit Routine (Taking a Checkpoint at End-of-Volume)</b>
<b>59</b>	<b>Index</b>

## FIGURES

- 3 Figure 1. Standard Eligible System Completion Codes
- 7 Figure 2. Obtaining Updated TCB Information after Restart
- 8 Figure 3. Requesting a Resource after Restart
- 13 Figure 4. Using One Sequential Checkpoint Data Set to Ensure Restart
- 13 Figure 5. Using Two Sequential Checkpoint Data Sets to Ensure Restart
- 14 Figure 6. Recording a Checkpoint Identification Assigned  
by the Control Program
- 15 Figure 7. Canceling a Request for Automatic Restart
- 28 Figure 8. Requesting Automatic Step Restart
- 28 Figure 9. Requesting Automatic Checkpoint/Restart
- 32 Figure 10. Requesting a Deferred Step Restart
- 34 Figure 11. Requesting a Deferred Checkpoint/Restart





# **SUMMARY OF CHANGES FOR RELEASE 21**

## ***Support of DOS Tapes***

Information on taking a checkpoint with DOS tape files has been added to the manual.

## ***Size of Access Method Modules***

The approximate size of access method modules that must be resident in main storage has been updated.

## ***Miscellaneous Changes***

- “Chapter 1: Introduction” has been rewritten. The detailed information (on the checkpoint/restart components and how to request restart) previously in this chapter has been incorporated into chapters 3 and 4.
- Minor technical and editorial changes have been made throughout the manual.



# CHAPTER 1: INTRODUCTION

Advanced checkpoint/restart is a technique for recording information about a job at programmer–designated checkpoints so that, if necessary, the job can be restarted at one of these checkpoints or at the beginning of a job step.

A checkpoint is taken when a user program issues the `CHKPT` macro instruction. This macro causes the contents of the program’s main–storage area and certain system control information to be written as a series of records in a data set. These records can then be retrieved from the data set if the job terminates abnormally or produces erroneous output, and the job can be restarted. Restart can take place immediately (initiated by the operator at the console) or be deferred until the job is resubmitted. In either case, the time–consuming alternative of rerunning an entire job is eliminated.

## Types of Restart

The checkpoint/restart program allows four types of restart:

- automatic step restart
- automatic checkpoint/restart
- deferred step restart
- deferred checkpoint/restart

Automatic restarts are initiated by the operator at the console. Automatic step restart, which is restart at the beginning of a job step, is requested in the job control language. Automatic checkpoint/restart, which is restart at the last checkpoint taken before the job failed, is requested in the `CHKPT` macro instruction.

Deferred restarts take place when a job is resubmitted to be run. Deferred step restart takes place at the beginning of the job step specified in the job control language. Deferred checkpoint/restart takes place at the checkpoint specified in the job control language.

## Components of Checkpoint/Restart

### *CHKPT Macro Instruction*

The `CHKPT` macro is coded in the user’s program to cause a checkpoint to be taken. In addition, it may request automatic restart at the last checkpoint taken.

When a `CHKPT` macro is executed, the contents of the program’s main–storage area and certain system control information are written, as a series of records, in a data set. The series of records is called a checkpoint entry, and the data set in which they’re written is called a checkpoint data set. The checkpoint entry, which has a unique programmer–specified or system–generated identification called a checkid, is retrieved from the data set when restart occurs.

Chapter 2 explains in detail how to establish a checkpoint.

## ***End-of-Volume Exit Routine***

The end-of-volume exit routine is coded in the user's program to allow execution of the CHKPT macro instruction each time the processing of a multivolume physical sequential user data set is continued on another volume. Appendix B contains more detailed information about the end-of-volume exit routine.

## ***RD (Restart Definition) Parameter***

The RD parameter is coded in the JOB or EXEC statements and is used to request automatic step restart if job failure occurs and/or to suppress, partially or totally, the action of the CHKPT macro instruction. Chapter 4 contains more detailed information about this parameter.

## ***RESTART Parameter***

The RESTART parameter, coded in the JOB statement, is used when a job is resubmitted for restart (deferred restart). It specifies either the step (for deferred step restart) or the step and the checkpoint within that step (for deferred checkpoint/restart) at which restart should begin. Chapter 4 contains more detailed information about this parameter.

## ***SYSCHK DD Statement***

The SYSCHK DD statement is used to request deferred checkpoint/restart when a job is being resubmitted. Chapter 4 contains more detailed information about the SYSCHK DD statement.

## ***CKPTREST System Generation Specification***

The CKPTREST macro instruction specifies, at system generation, which of the completion codes accompanying abnormal step termination indicates that the step is eligible to be restarted. During system generation, a standard, IBM-defined set of system completion codes (codes emitted when the system executes ABEND) is placed in a table of eligible codes. The table becomes part of the control program. CKPTREST, which is optional, can be used to delete system completion codes from the table and to add user completion codes (codes emitted when the user's program executes ABEND) to the table. The syntax of the macro instruction is:

---

```
CKPTREST [NOTELIG = (hex-code(,hex-code)...)][,ELIGBLE = (dec-code(,dec-code)...)]
```

---

The NOTELIG operand can be used to delete any number of system completion codes from the table of eligible codes; *hex-code* is specified as a three-character hexadecimal number.

The ELIGBLE operand can be used to add up to ten user completion codes to the table; *dec-code* is specified as a decimal number having a maximum value of 4095.

If multiple codes are specified in either operand, the codes can be specified in any order.

The IBM-defined set of eligible system completion codes is listed in Figure 1.

---

001	106	2F3 <sup>1</sup>	422	714	926
031	113	313	513	717	937
033	117	314	514	737	A14
03A	137	317	613	806	B14
0A3	20A	32D	614	813	B37
0B0	213	413	626	837	C13
0F3	214	414	637	906	E37
100	217	417	700	913	

<sup>1</sup>Code 2F3 indicates that a job was executing normally when MFT or MVT system failure occurred. The code is included in a console message displayed during system restart.

Figure 1. Standard Eligible System Completion Codes

---

**Note:** Whether or not the CKPTREST macro instruction is used, the SUPRVSOR macro instruction must be used to specify resident access methods. For details, refer to “Resident Access Methods” in Chapter 6.

If CALL IEHREST is used in PL/I programs, the CKPTREST macro instruction must specify 4092 as an eligible user completion code.



## CHAPTER 2: HOW TO ESTABLISH A CHECKPOINT

This chapter explains how a user may establish checkpoints at which to restart job steps. The topics discussed are:

- CHKPT macro instruction
- Cautions in taking a checkpoint
- DCB for a checkpoint data set
- DD statement for a checkpoint data set
- Use of checkpoint data sets

### CHKPT Macro Instruction

The CHKPT macro instruction is coded in the user's program. When the CHKPT macro is executed, job step information about the user's program, main-storage data areas, data set position, and supervisor control is written as a checkpoint entry in a checkpoint data set. The point at which this information is saved becomes a checkpoint from which a restart may be performed if the job terminates abnormally or the system fails. After the checkpoint entry is written, control returns to the user's program at the instruction following the CHKPT macro.

The CHKPT macro instruction refers to the data control block (DCB) for the checkpoint data set. The checkpoint data set can be opened for output before the CHKPT macro instruction is executed. If the data set is not open, the checkpoint routine opens it and then closes it after writing the checkpoint entry. If the data set is open, the checkpoint routine writes the checkpoint entry, but does not close the data set.

The checkpoint data set must be on one or more magnetic tape volumes or on one direct-access volume. A checkpoint data set can reside on a magnetic tape with IBM standard labels, nonstandard labels, or no labels. American National Standard labels cannot be used for a checkpoint data set.

The standard form of the CHKPT macro instruction is:

---

```
[symbol] CHKPT {dcb address[,checkid address [,checkid length ]] }  
                [, 'S' ]  
                {CANCEL }  
                }
```

---

The operands are defined as follows:

*dcb address*

is the address of the DCB for the checkpoint data set. The DCB must specify use of BSAM or BPAM. It must also specify RECFM=U or UT, MACRF=(W), DDNAME= *anyname*, and BLKSIZE= *nnn*, where *nnn* is at least 600 bytes but not more than 32,760 bytes for magnetic tape, and not more than the track length for direct-access devices. (If the data set is opened by the control program, blocksize need not be specified; the device-determined maximum blocksize is assumed if blocksize is not specified. For seven-track tape, the DCB must specify TRTCH=C and DEVD=TA; for direct-access devices, it must specify or

imply KEYLEN=0. To request chained scheduling, OPTCD=C and NCP=2 can be specified.

#### **CANCEL**

cancels the request for automatic checkpoint/restart. Automatic step restart can occur if RD=R was specified. If CHKPT without CANCEL is then executed before abnormal termination, a request for automatic checkpoint/restart is again in effect. Checkpoint entries written before a CHKPT with CANCEL are left intact and may be used to perform a deferred checkpoint/restart.

#### *checkid address*

specifies the address of a programmer-provided field that is to contain a unique, printable identification of the checkpoint entry. The identification is called a checkid. The checkpoint routine writes the checkid as part of the entry and prints it in a message on the operator's console when it finishes writing the entry. The programmer must subsequently use the checkid by coding it in the JOB statement RESTART parameter if he wishes to use the corresponding entry to perform a deferred restart at a checkpoint. If the *checkid address* operand is omitted, the checkid length or 'S' operand is invalid.

#### *checkid length or 'S'*

*Checkid length* is the length in bytes of the field that contains the checkid. The maximum length of this field is 16 bytes when the checkpoint data set is physical sequential, 8 bytes when it is partitioned. (For a partitioned data set, the field can be longer than the actual checkid identification if the unused low-order portion of the field contains blanks.) By coding this operand or by omitting this operand entirely (in which case a length of 8 bytes is implied), the programmer specifies that his program will form an identification and store it into the checkid field before CHKPT is executed. If the *checkid address* operand is omitted, this operand is invalid.

By coding this operand as 'S', the programmer specifies that the checkpoint routine is to generate an identification 8 bytes in length and store it in the checkid field. If the *checkid address* operand is omitted, this operand is invalid.

### ***Programming Notes on the CHKPT Macro Instruction***

If both checkid address and checkid length or 'S' are omitted, the checkpoint routine generates an identification and writes it in the checkpoint entry and on the operator's console, but does not return it to the user's program.

If the programmer provides the checkpoint identification and the checkpoint data set is sequential, the identification can be any combination of up to 16 alphanumeric, special characters, and blanks. For a partitioned data set, it must be a valid member name of up to eight alphanumeric. The identification for each checkpoint should be unique. If two identifications differ only by having a different number of trailing blanks, the control program considers them to be the same.

A checkpoint identification generated by the checkpoint routine consists of the letter C followed by a seven-digit decimal number. The number is the total number of checkpoints taken by the job; it includes the current checkpoint, checkpoints taken earlier in the job step, and checkpoints taken by any previous steps of the job.

The checkid address operand allows a user's program to select fields in the records of an input data set and use them as checkids. Alternatively, the user's program may use



the checkid address and the 'S' operands and include a system-generated checkid in the current record of an output data set.

### ***Exceptional Conditions***

The CHKPT macro instruction returns a code in register 15 to indicate whether the CHKPT macro instruction was executed successfully. Appendix A contains a list of these codes and their meanings.

### ***List and Execute Forms of CHKPT***

The CHKPT macro instruction may be coded in the list and execute forms as well as in the standard form. The dcb address, checkid address, and checkid length operands can be coded in the list and execute forms; the CANCEL operand must not be coded.

A complete description of the list and execute forms of this macro instruction appears in *OS Supervisor Services and Macro Instructions*, GC28-6647.

## **Cautions in Taking a Checkpoint**

The following discusses certain cautions that should be observed when taking a checkpoint. These cautions relate to the operation of certain macro instructions, serially-reusable resources, and special operating system features. Cautions that relate to user data sets are listed in Chapter 3.

### ***Use of CHKPT With Other Macro Instructions***

**EXTRACT:** The EXTRACT macro instruction is used to obtain information from the task control block (TCB). TCB information is subject to change when the task is terminated and the job step is restarted. If the information is needed after restart, the EXTRACT macro instruction should be reissued after the checkpoint is taken, as shown in Figure 2.

---

```
      .  
      .  
      .  
      EXTRACT ANSADDR,FIELDS=(ALL)   Obtain TCB information  
      .  
      .  
      .  
      CHKPT   CHKPTDCB                Establish checkpoint  
      CH      15,=H'4'                Is restart in progress  
      BNE     NRESTART                No, branch to NRESTART  
      EXTRACT ANSADDR,FIELDS=(ALL)   Yes, obtain new information  
NRESTART  
      .  
      .  
      .
```

Figure 2. Obtaining Updated TCB Information After Restart

---

**SETPRT:** The SETPRT macro instruction is used in data management to load the UCS buffer for a 3211 or 1403 Printer with the universal character set feature or the forms control buffer (FCB) for a 3211 Printer. The buffer contents are not saved

when a checkpoint is taken. To reload the buffer upon restart, the user must reissue the SETPRT macro instruction.

**WTOR:** The reply to a WTOR macro instruction must have been received before CHKPT is issued.

**STIMER:** A time interval established by the STIMER macro instruction must have been completed before CHKPT is issued.

**ATTACH:** If ATTACH is issued in the program using CHKPT, all subtasks created must have terminated before CHKPT is issued; that is, the job-step task must be the only task of the step.

### *Use of CHKPT in Exit Routines*

The CHKPT macro instruction must not be used in an exit routine other than the end-of-volume exit routine. The user may take a checkpoint when a BSAM or QSAM data set reaches end-of-volume.

### *Explicit and Implicit Requests for ENQ*

When a job step terminates, it loses control of serially-reusable resources. If the step is restarted, it must request all of the resources needed to continue processing. Explicit use of a serially-reusable resource is requested when the user's program issues the ENQ macro instruction. If the program issues the ENQ and takes a checkpoint, it must issue the ENQ again whenever restart occurs at the checkpoint. Figure 3 shows a program that requests a serially-reusable resource by issuing an ENQ before establishing a checkpoint. After the checkpoint, it tests for a restart. If one has occurred, it requests the same resource again. It requests the resource again because the job step has terminated abnormally, has lost control of the resource, and has then been restarted from the checkpoint.

---

```
      .
      .
      .
      ENQ   ( QADDR, RADDR )
      .
      .
      .
      CHKPT  CHKPTDCB
      CH    15, =H'4'
      BNE   NRESTRT
      ENQ   ( QADDR, RADDR )
NRESTRT
      .
      .
      .
      DEQ   ( QADDR, RADDR )
      .
      .
      .
```

Figure 3. Request for a Resource After Restart

---

Some serially-reusable resources are requested implicitly by issuing data management macro instructions. These resources may be records that the user is processing or tracks on a direct-access device. To ensure correct processing, the user must not establish checkpoints while he has control of these resources:

- If the basic direct access method (BDAM) is used, the user's program must execute the WRITE or RELEX macro instruction to release a record that has been read with exclusive control, before executing the CHKPT macro instruction.
- If BDAM is used to add a record to a data set with variable-length or undefined records, BDAM issues an ENQ macro instruction for the capacity record (R0); therefore, the user's program must execute the WAIT or CHECK macro instruction to check completion of the write operation before it executes CHKPT.
- If the basic indexed sequential access method (BISAM) is used, a checkpoint must not be taken before waiting for completion of a write operation. If a record is read for update, a checkpoint must not be taken before writing the updated record and waiting for the write operation to be checked.
- If the queued indexed sequential access method (QISAM) is used, an ESETL macro instruction must be issued before taking a checkpoint if a SETL macro instruction was issued previously. Another SETL macro instruction may be issued after the checkpoint.
- Use of the RESERVE macro instruction (see "Shared DASD" caution below).

### ***Use of Special Operating System Features***

**Shared DASD:** At some installations, a direct-access storage device is shared by two or more independent computing systems. This device is a serially-reusable resource. If it is being used when a checkpoint is taken, it must be requested after a restart from the checkpoint. This resource is requested by a special macro instruction, RESERVE, described in the *OS MVT Guide* or *OS MFT Guide*.

**Rollout/Rollin:** The rollout/rollin feature of MVT allows a job to obtain storage outside of its region by causing another job to be rolled out. The CHKPT macro instruction does not execute successfully if issued while the job step is using main storage outside of its assigned region.

## **DCB For a Checkpoint Data Set**

### ***Required DCB Parameters***

The programmer must provide a DCB for the checkpoint data set. (The publication *OS Data Management Macro Instructions* contains detailed information about coding DCBs.) The following parameters must be included in this DCB:

- Data set organization—BSAM or BPAM (DSORG=PS or PO).
- Macro instructions—WRITE (MACRF=W).
- Record format—undefined (RECFM=U).
- Device type—direct access or tape (DEVD=DA or TA).
- Key length—none.

- Buffer parameters—none.
- Magnetic tape recording technique—data conversion with odd parity (TRTCH=C). This is required only if the data set is on a 7-track magnetic tape.

The programmer must code the DSORG and MACRF operands and the DDNAME operand in the DCB macro instruction. He may code the RECFM, DEVD, and TRTCH operands in the DCB macro instruction, or he may code, in the related DD statement, the RECFM and TRTCH subparameters of the DCB parameter. Because RECFM and DEVD have default values of U and DA respectively, they need not be provided explicitly in either the DCB macro instruction or the DD statement. The LABEL parameter of the DD statement describes the labels of a data set on magnetic tape. For a checkpoint data set, the programmer can specify IBM standard labels (SL or SUL), nonstandard labels (NSL), or no labels (NL). American National Standard labels (AL or AUL) cannot be specified for a checkpoint data set. If the label type is not specified, the operating system assumes that the data set has IBM standard labels.

### ***DCB Options***

The programmer may optionally provide the following DCB parameters:

- Blocksize (BLKSIZE $\geq$ 600)—600 bytes minimum
- Write validity checking (OPTCD=W)
- Track overflow (RECFM=UT)
- Number of channel programs (NCP=2)—two
- Chained scheduling (NCP=2 and OPTCD=C)

### **Notes on DCB:**

- BLKSIZE is required if the user opens the checkpoint data set. If the checkpoint routine opens the data set and BLKSIZE is omitted, the checkpoint routine provides a BLKSIZE parameter having a value of the track size if the checkpoint data set is on a direct access volume, or 32,760 bytes if the data set is on a tape. The routine writes control records having internally specified lengths, but it writes the contents of the programmer's main-storage area in blocks of the length specified by BLKSIZE.
- Requests for two channel programs or chained scheduling apply only to the writing of main-storage records, not to the writing of control records or the reading of records for a restart. Because main-storage records are written directly from main storage without being buffered, the requests do not cause an increase in the work area used by the checkpoint routine.
- OPTCD=Q cannot be specified in the DCB.

## **DD Statement For a Checkpoint Data Set**

The DD statement for the checkpoint data set must define the data set in a normal way. (*OS Job Control Language Reference* contains detailed information on coding the DD statement.)

The only restrictions on the statement are:

- The UNIT parameter must specify a tape or direct-access device supported by BSAM or BPAM. The device can be specified by referring to a specific device, a device type, or a group of devices. DEFER should not be coded in the DD statement.
- Secondary space allocation may be requested (by the increment subparameter), but it will not be performed. (See notes.)
- The LABEL parameter must not specify ANSI tape labels.
- OPTCD=Q cannot be specified as a DCB subparameter.

#### Notes on DD Statement:

- The initial disposition of the data set (as specified in the DISP operand of the DD statement) is used in a normal way to position the checkpoint data set when it is opened, regardless of whether the user's program or the checkpoint routine executes the OPEN macro instruction. A more detailed discussion appears in the next section.
- The final and conditional dispositions of the data set have their normal meanings. However, if termination is occurring and an automatic restart at a checkpoint is to occur, the system automatically keeps all data sets that are in use by the job, including the checkpoint data set.
- If end-of-volume (no more primary space) is encountered while writing a checkpoint on a direct-access volume, two actions are possible:
  1. If the programmer requested secondary allocation, the allocation is performed, and the checkpoint routine issues return code 08. The allocated space is not used.
  2. If the programmer did not request secondary allocation, the system executes an ABEND macro instruction applying to the step. The ABEND causes emission of a D37 system completion code, which is not a code that makes the step eligible for restart. Thus, even though secondary space will not be used, secondary allocation should be specified to avoid abnormal termination.

Examples of DD statements for the checkpoint data set are:

```
//ddname DD DSNAME=dsname,UNIT=TAPE,DISP=(MOD,KEEP)
//ddname DD DSNAME=dsname,UNIT=SYSDA,DISP=(NEW,DELETE,KEEP), X
//          SPACE=(TRK,(300,1)),VOLUME=SER=CKPTDS
```

## Use of Checkpoint Data Sets

### *How Checkpoint Entries Are Written*

If the user's program did not open the checkpoint data set before it executed the CHKPT macro instruction, the checkpoint routine opens it. The checkpoint entry is then written at a position determined by whether the data set is sequential or partitioned, and by the DISP parameter on the related DD statement. If the data set is sequential and its disposition is NEW or OLD, the checkpoint entry is written at the

beginning of the data set. If the data set is sequential and its disposition is MOD, or if the data set is partitioned, the checkpoint entry is written after the last entry existing in the data set.

If the checkpoint data set is partitioned, each checkpoint entry is a member, and its checkid is its member name. After it writes a checkpoint entry, the checkpoint routine executes a STOW macro instruction to add the checkid of the entry to the directory of the data set. If an identical checkid already exists in the directory, the related address of a member is changed to be the address of the new checkpoint entry. The initial disposition specified for the checkpoint data set has no effect on the STOW operation.

If the checkpoint routine opens the checkpoint data set, it also closes it.

If the user's program opens the checkpoint data set for output, the checkpoint routine simply writes a checkpoint entry at the data set's current position and does not close the data set. If the user opens the checkpoint data set, he need not close it after taking the last checkpoint for the job step. If many checkpoints are taken, leaving the data set open will save time. All of the checkpoint entries will be saved in this case, thus providing the ability to request a deferred restart from any of the checkpoints. If the data set is partitioned, the checkpoint routine executes a STOW macro instruction as it would if it had opened the data set.

If end-of-volume is encountered during writing of a checkpoint entry on tape, the system requests mounting of a new volume. The checkpoint routine writes the entire checkpoint entry on the new volume. Only checkpoint data sets on tape may be multivolume data sets. The previous section, "DD Statement for the Checkpoint Data Set," discusses what occurs if end-of-volume (no more primary space) is encountered during writing of a checkpoint entry on a direct-access volume.

The status (open or closed) and position of a checkpoint data set remain the same at restart as they were after execution of the CHKPT macro instruction that established the checkpoint.

Note that a checkpoint data set must contain only checkpoint entries. A checkpoint entry must not be written in one of the user's data sets. Conversely, the program must not write its own data in a checkpoint data set. Note also that a checkpoint data set may not be a concatenated data set.

### ***How to Ensure Restart***

To ensure that restart at the most recent checkpoint will be possible, a checkpoint entry must not be written over a preceding checkpoint entry, because abnormal termination or system failure may occur while the new entry is being written. Three methods by which the programmer can ensure that restart will be possible are suggested below. All three methods involve the use of sequential checkpoint data sets.

Figure 4 shows the use of one sequential checkpoint data set, one data control block, and one DD statement (CHECKDD) specifying MOD disposition. The user allows the checkpoint routine to open and close the data set each time it writes a checkpoint entry. Checkpoint entries will be written sequentially in the data set. Performance would be improved if the user's program opened the data set and kept it open; the disposition could then be NEW or OLD.

---

```

Program
.
.
.
CHKPT CHKDCB
.
.
.
CHKDCB DCB DDNAME=CHECKDD,MACRF=W,DSORG=PS
.
.
.

DD Statement

//CHECKDD DD UNIT=TAPE,DISP=(MOD,KEEP)

```

Figure 4. Using One Sequential Checkpoint Data Set to Ensure Restart

---

Figure 5 shows a way to alternate data sets when all checkpoints are taken by one CHKPT macro instruction. The data sets are opened by the control program and are identified by two DD statements, CHECKDD1 and CHECKDD2. The data control block initially refers to CHECKDD1. Before the second checkpoint, it is changed to refer to CHECKDD2; before the third checkpoint, it is again changed to refer to CHECKDD1, and so forth. In this way, one data control block can be used for two data sets that are not open at the same time.

---

```

Program
.
.
.
DCBD DSORG=PS Define IHADCB (dummy section
that defines DCBDDNAM)
CSECT Resume original control section
.
.
.
LA 2,CHECKDCB Establish CHECKDCB as base
USING IHADCB,2 address for IHADCB
XC DCBDDNAM(8),DDHOLD Exchange ddname in CHECKDCB for
XC DDHOLD(8),DCBDDNAM ddname in DDHOLD
XC DCBDDNAM(8),DDHOLD
CHKPT CHECKDCB Open, checkpoint, close
.
.
.
DDHOLD DC C'CHECKDD1'
CHECKDCB DCB DSORG=PS,MACRF=(W),DDNAME=CHECKDD2

DD Statements

CHECKDD1 DD UNIT=SYSDA,DISP=NEW
CHECKDD2 DD UNIT=SYSDA,DISP=NEW

```

Figure 5. Using Two Sequential Checkpoint Data Sets to Ensure Restart

---

An alternate method of using two sequential data sets is to use two DCBs and two DD statements specifying NEW or OLD dispositions, and to execute alternately two CHKPT macro instructions, each referring to a different data set. Performance would

be improved when using direct-access data sets if the user's program opened the data sets, kept them open, and used the POINT macro instruction to reposition them.

The method illustrated in Figure 4 saves all checkpoint entries for possible use in deferred restart, while the method illustrated in Figure 5 conserves auxiliary storage. Note that none of the methods requires use of a particular device type.

### ***How Checkpoint Entries Are Identified***

Any number of checkpoint entries can be written in a checkpoint data set, and any number of checkpoint data sets can be used concurrently. In a sequential checkpoint data set, checkkids of valid or invalid checkpoint entries in one data set should be unique. In a partitioned data set, checkkids of valid entries should be unique.

When the control program assigns identifications, the identification for each checkpoint is unique. The identification is 8 bytes in length and consists of the letter C followed by a seven-digit decimal number. This number reflects the total number of checkpoints taken by the job, including the current checkpoint, checkpoints taken earlier in the job step, and checkpoints taken by any previous job steps.

If the programmer specifies checkkids instead of having the system generate them, he may erroneously specify duplicate checkkids. The system does not recognize this error. When deferred restart at a checkpoint occurs and the checkpoint data set is sequential, the system searches the data set from its beginning for the specified checkpoint entry. It uses the first entry it finds that has the specified checkkid. If the data set is partitioned, the system searches the data set's directory to find the location of the specified checkpoint entry. If two or more entries having the same checkkid were written in the data set, the most recent of those entries is the one pointed to by the directory, and restart occurs from the most recent entry.

---

```

      .
      .
      .
      CHKPT  CHKDCB, ID, 'S'           Take checkpoint
      LTR    15, 15                   Was checkpoint taken
      BNZ    PHASE2                   No, branch to PHASE2
      PUT    STEPLOG, MESSAGE        Yes, print checkpoint ID
PHASE2
      .
      .
      .
MESSAGE  DC    H'45,0'                Record length, etc.
          DC    C'SUCCESSFUL CHKPT AT PHASE2...ID='
ID        DS    CL8
STEPLOG   DCB   DSORG=PS,MACRF=(PM),RECFM=V,BLKSIZE=128,           C
          LRECL=124,DDNAME=LOGDD
CHKDCB    DCB   DSORG=PS,MACRF=(W),RECFM=U, BLKSIZE=32760,       C
          DDNAME=CHKDD

```

Figure 6. Recording a Checkpoint Identification Assigned by the Control Program

---



Checkpoint entries have two identifications. The primary identification is the programmer-generated or system-generated checkid specified or requested by the CHKPT macro instruction. The secondary identification is identical to the system-generated checkid that might have been requested by CHKPT. The primary identification is used when a search is made for a checkpoint entry. The secondary identification is then used as a base to compute the system-generated checkids of entries written after restart has occurred. This procedure prevents the system from generating checkids that are duplicates of checkids of existing useful entries.

The control program identifies each checkpoint in a message to the operator; on request, it also makes the identification available to the user's program. In Figure 6, the CHKPT macro instruction requests the control program to supply an identification and place it in the 8-byte field named ID. When the checkpoint is successfully taken, the program prints the identification as part of a message to the programmer.

### ***How to Use the CANCEL Option***

After being restarted, the job step may again terminate abnormally. If it does, it may again be restarted from the same checkpoint, subject to operator authorization. If the user wishes to avoid restarting the job step twice from the same checkpoint, the sequence shown in Figure 7 may be coded.

---

```

      .
      .
      .
      CHKPT   CHKPTDCB   Establish checkpoint
      CH      15,=H'4'   Is restart in program
      BNE     NRESTART   No, branch to NRESTART
      CHKPT   CANCEL     Yes, cancel restart request
NRESTART
      .
      .
      .

```

Figure 7. Canceling a Request for Automatic Restart

---

After the successful initiation of a checkpoint/restart, the system places a return code of 04 (hexadecimal) in general register 15 and returns control to the user's program at the instruction that follows the CHKPT macro instruction. At this time a request for another automatic restart at the same checkpoint is normally in effect. In Figure 7, the instruction that follows the CHKPT macro instruction tests the return code to determine whether control has been returned as the result of a restart. If the return code is 04, a restart has just occurred, and a second CHKPT macro instruction is executed. This macro instruction has a CANCEL operand, which cancels the existing request for an automatic restart. If the job step again terminates abnormally after a restart from the checkpoint, automatic restart can occur only at a later checkpoint. It will not occur at the checkpoint preceding the canceled checkpoint.



## CHAPTER 3: USER DATA SETS

This chapter examines considerations in the handling of the user's data sets. The first part addresses considerations concerned with jobs that will be restarted at a checkpoint, the second, those considerations that apply to both step restart and checkpoint restart.

### What to Consider for Checkpoint/Restart

The checkpoint routine records information about all data sets used by the step executing the CHKPT macro instruction. Recorded information includes:

- For all data sets, the information that can be coded on a DD statement, for example, device type and volume serial numbers. (The contents of the step's JFCBs are recorded.)
- For data sets open at the checkpoint, being processed on either magnetic tape or direct-access devices, and using the BSAM, QSAM, BISAM, QISAM, BPAM, BDAM, and EXCP access methods, the information needed to reposition the data sets if restart occurs at a checkpoint.

### *Cautions*

- Data sets are repositioned at restart only if they were open when the checkpoint was taken. The Open routine will position normally all data sets opened after the checkpoint was taken.
- Unit record data sets are not repositioned (printer, punch, or card reader) at restart.
- If the programmer uses EXCP to process a tape data set open at a checkpoint, he should ensure that the block count in the data set's data control block is correct. If the block count is incorrect, the system may position the data set incorrectly when restart occurs.
- The system does not save and restore the contents of data sets. Therefore, the programmer must ensure that input data sets and system data sets contain all necessary data when restart occurs. If a data set on a direct-access volume is open at the checkpoint, the data set's label (the DSCB in the VTOC) must have the same location and reflect the same extents upon restart as it did when the checkpoint was taken. (See Chapter 4, the section on "Deferred Checkpoint Restart," and the subsection "JCL Requirements and Restrictions" Footnote 1.)
- When data set records are processed in an update-in-place manner (records are read, changed, and then written back into their original location in the data set), restart will be successful only if all records updated after the last checkpoint was taken are restored to their original state or if the user's program keeps track of the records that are updated and avoids updating them again during restart.
- If a checkpoint is taken and then a MOD data set (tape or direct-access) or a partitioned data set is opened, another checkpoint should be taken before any records are written into the data set. If the second checkpoint is not taken and restart occurs at the first checkpoint, the Open routine will position to the current end of the data set instead of to the original end.

Upon restart and after repositioning a partitioned data set opened for output (necessarily open at the checkpoint if it is to be repositioned), the system deletes member names from the data set's directory if the corresponding members are located in the data set at positions following the data set's current position.

If the user's program writes multiple members in a partitioned data set, it should take a checkpoint not only after it opens the data set, but also after each execution of the STOW macro instruction.

- Members may be deleted from a partitioned data set during a restart. If this action may delete members written by another job (another job may have been executed between the original and restart executions of the subject job), restart at a checkpoint should not be requested.
- When a step using the UCS (universal character set) feature is restarted, the system does not determine whether the UCS buffer is properly loaded, nor does it alert the operator to the UCS requirements of the step.
- If a checkpoint is taken, and then an output data set is extended onto a second direct-access volume (because end-of-volume occurred on the first volume and there was no more space available on the volume, or the data set contained 16 extents), and restart subsequently occurs at that checkpoint, the system does not delete the extension of the data set.
- Checkpoints should not be taken before an ISAM data set is opened in load mode. A checkpoint should be taken immediately after such an OPEN. Otherwise, an ABEND with a code of 03E will result from a restart at a previous checkpoint.
- When a job step is restarted from a checkpoint, the device type for each data set must be the same as during the original execution. If a DD statement specified a user-defined collection of devices (for example, UNIT=SYSDA), a device of the same type is assigned, but the device may or may not be a member of the specific collection. If a DD statement specified a specific device (for example, UNIT=190), a device of the same type is assigned, but the device may or may not be the specific device requested.
- Tape data set repositioning during a checkpoint/restart under MFT may severely degrade system performance if module IGC0S05B is not resident.
- Checkpoints may be taken with DOS tape files opened with the bypass leading tapemark option LABEL=(,LTM) and/or the bypass embedded DOS checkpoint records option DCB=(OPTCD=H) specified. However, a checkpoint must not be taken when an opened data set:
  - resides on a DOS 7-track tape,
  - is written in translate mode, and
  - contains embedded checkpoint records
- An ISAM data set that is shared must be closed before a checkpoint is taken.
- A user who closes his SAM data set immediately after restarting his program at a checkpoint should be aware that the data set may not be restored to the same condition it was in when the checkpoint was originally taken.

## Repositioning User Data Sets

The checkpoint routine records positioning information for user data sets as follows:

- *SYSIN and SYSOUT data sets.* The checkpoint routine waits until all requested input/output operations are complete. Then it records positioning information.
- *All other user data sets.* If input/output operations were requested but were not begun (for example, if a READ macro instruction was executed, but the related channel program was not started), the checkpoint routine stops any processing associated with the I/O request, records the positioning information, and then reestablishes I/O operations.

If I/O operations have already begun, the checkpoint routine waits until they are complete before recording positioning information.

User data sets that were open at a checkpoint are repositioned upon restart to the positions existing at the checkpoint, except in the case of data sets on unit-record devices. Upon restart, writing of a data set on a printer or punch, or reading of a data set from a card reader, is simply resumed at the current position of the device.

When QSAM or QISAM is being used to process a data set, an indeterminate number of main-storage buffers may contain data when a checkpoint is taken. If restart at a checkpoint occurs, the system's action depends on whether a card reader or another type of device is being used to process the data set:

- *Card reader being used (QSAM only).* Upon restart, existing buffer contents are released. The buffers are reprimed by reading records from the current data set into them.
- *Another device being used.* Upon restart, the buffer contents are restored to main storage, and processing continues normally. Note that it is not possible to predict the time—either before or after the checkpoint—when a given record will be transferred between a buffer and the recording medium.

When a basic access method is being used to process a data set, processing resumes normally upon restart. If the user's program wants to ensure that a particular block is read or written before a checkpoint is taken, and if the data set is not SYSIN or SYSOUT, the program should complete the operation by executing the CHECK or WAIT macro instruction before it executes the CHKPT macro instruction. If the program does not complete the operation, the block may be read or written either before or after the checkpoint is taken.

When QSAM or BSAM is being used to read a data set from a card reader, the user's program can reposition the data set upon restart. If the user provides a repositioning routine, he should instruct the operator to position the data set to the beginning if a restart becomes necessary. The program might operate as follows:

- The program saves the first record read from the data set and keeps a count of the number of records read before each checkpoint.
- After a restart, the repositioning routine reads a record from the data set and compares it with the first record read before abnormal termination. If the records are identical, the data set has been positioned to the beginning. The routine then repositions it by reading (without otherwise processing) the number of records read before the checkpoint.

## ***Preserving Data Set Contents***

**Update in Place:** The control program repositions data sets but does not preserve their contents. After taking a checkpoint, the user must ensure that data set contents are not changed in a manner that will make successful restart impossible.

If the user's program reads records from a data set, updates them, and writes them back to their original locations, it may be useless to take a checkpoint before completing this processing. If a checkpoint is taken earlier, restart will be unsuccessful under these circumstances:

- The user's program updates a record before abnormal termination and repeats the update after restart, and
- The updated record contents depend on the original contents.

For example, suppose that the purpose of the update is to switch the positions of two fields in each record. If the record is updated twice, the fields are returned to their original positions, and the results are invalid. In a different application, an update might simply place a value in a record field, regardless of the field's original contents. The user could then restart the step at a checkpoint taken before or during the update procedure, because an updated record would not be changed if updated again after restart.

**Updating a PDS:** When a partitioned data set is updated, the user must be careful to preserve the contents of the directory. The directory consists of entries that point to each member of the data set.

When a member is added to a partitioned data set, an entry is also added to the directory. If one member is added, the STOW macro instruction may be used to create the entry, or the member name may be specified in the DD statement; in the latter case, the control program creates the directory entry when the data set is closed or when the job step terminates. If more than one member is added, the STOW macro instruction must be used to create an entry for each member.

When one or more members are added to a partitioned data set, a checkpoint must be taken immediately after opening the data set. After taking the checkpoint, the new member may be written and its entry added to the directory. Then, if the step is restarted from the checkpoint, the data set is repositioned; the new member and its directory entry are deleted and are recreated after restart.

To update a member of a partitioned data set, updated records may either be written back to their original locations, or the entire member (in updated form) may be rewritten as a new member of the data set. In the latter case, the directory entry must be updated to point to the rewritten member.

If a checkpoint is taken before rewriting an entire member, one must also be taken immediately after updating the directory, because the control program will delete the updated directory entry if it repositions the data set for restart from the original checkpoint. Since no entry then points to the original member, the restart will be unsuccessful.

**Work Data Sets:** Many programs use "work" data sets, which are alternately written and read, rewritten and reread. If a work data set is used, a checkpoint should be

taken each time the user has finished reading the data set and before rewriting it. For example, a program may perform the following sequence of operations to produce different versions of data set A:

1. Write and then read back A1.
2. Write and then read back A2.
3. Write and then read back A3.

A checkpoint should be taken at the very beginning of operations 2 and 3 before any rewriting of data set A takes place. If, for example, the job step is abnormally terminated while operation 2 is in progress, the job step can be restarted from the checkpoint taken at the beginning of operation 2. At this checkpoint there is no need for the data in version A1.

### ***Nonstandard Tape Labels***

If tapes with nonstandard labels are used and steps are to be restarted at a checkpoint, the user must provide a routine to process nonstandard labels at restart time. This routine need only perform input header label processing, because output tapes will contain the header labels that were written when the data sets were opened (prior to checkpoint).

At restart time, the control program checks the tape to make sure that the first record is not a standard volume label. If the first record is 80 bytes in length and contains the identifier VOL1 in the first four bytes, the tape is not accepted. The control program issues a message directing the operator to mount the correct tape.

When it is determined that the tape does not contain a standard volume label, the control program's Restart routine gives control to the user's routine for processing nonstandard labels. When this routine receives control, the tape has been positioned at the interrecord gap preceding the nonstandard label (the tape has been rewound).

If the user's routine determines that the wrong volume is mounted, a 1 must be placed in the high-order bit position of the SRTEDMCT field of the unit control block (UCB), and control is returned to the control program. The control program then issues a message directing the operator to mount the correct volume. When the new volume is mounted, the control program again checks the initial label on the tape before giving control to the user's routine.

Before returning control to the control program, the user's routine must position the tape at the interrecord gap that precedes the initial record of the appropriate data set. This applies to both forward and backward read operations. The control program then uses the block count shown in the data control block to reposition the tape at the appropriate record within the data set. This positioning is always performed in a forward direction. If the block count is zero, or a negative number, the control program does no positioning. (If the user wants the control program to reposition the tape during a restart, his normal header label routines—Open and EO—must properly initialize the block count field of the data control block during the original creation. The block count field of the data control block must not be altered at restart time.) For additional information about tape labels, refer to *OS Tape Labels*.

### ***Input/Output Errors***

The checkpoint routine issues return code 0C if it encounters a permanent input/output error during quiescing of queued access method input/output operations or during

writing of the checkpoint data set. An exception occurs when QSAM is being used and the skip or accept option is specified in the EROPT parameter of the data set's data control block. In this case, code 00 is returned.

When an access method other than QSAM or QISAM is used, the user's program can ensure that input/output operations are complete before it executes the CHKPT macro instruction, and it can thereby avoid having read or written an erroneous record while quiescing.

If a permanent error occurs when the system reads a checkpoint data set to perform a restart, the restart step is terminated abnormally with the system completion code 13F. Further automatic restart of the step is not attempted.

## What to Consider for Checkpoint or Step Restart

### *Generation Data Sets*

The control program of the operating system allows a generation data set to be created in one step of a job and then referred to in a later step by the relative generation number used to create it. For example, a data set can be created in one step as the +1 generation and read in a following step also as the +1 generation, instead of as the 0 generation. The same relative generation number can be used because the system records in an internal table, called the bias table, the number of generation data sets created in each generation data group used by the current job. When the job uses a relative generation number to refer to a generation, the system subtracts the bias value from the specified number to determine the actual number of the desired generation.

If a generation data set is to be referred to later by a relative generation number, the DD statement used to create the generation data set must cause cataloging of the data set at the end of the step creating the data set. The programmer may use a conditional disposition to prevent cataloging at the end of a step that terminates abnormally.

Before the availability of the checkpoint/restart facility described in this text, the bias table was updated at the beginning of a step creating a generation data set. With the release of this facility, the table is updated at the end of the step, whether or not the step terminates normally, and whether or not cataloging occurs. (This method of updating must be considered if a step is executed after abnormal termination and refers to a generation data set.) However, the bias table is not updated if automatic step restart or restart at a checkpoint is occurring, nor does cataloging occur in this case. Because the original bias table is used when an automatic restart occurs, generation data sets can be referred to during the restart exactly as they could be during the original execution.

If a deferred step restart is performed, the bias table contents existing during the original execution do not exist during the restart. Therefore, generation data sets created and cataloged during the original execution, in steps preceding the restart step, must be referred to during the restart execution by their actual relative generation numbers. Conditional dispositions should be used during the original execution to delete generation data sets created by the restart step.

When a checkpoint is taken, the system records in the checkpoint entry the bias table contents existing at the beginning of the current step. These contents are restored to the bias table if a deferred restart at a checkpoint is performed. If conditional dispositions are used during the original execution to keep (instead of to catalog)



generation data sets created by the restart step, those data sets, and generation data sets created and cataloged in steps preceding the restart step, can be referred to during the restart in the same way as they could be originally.

### ***Preallocated Data Sets***

In MVT, direct-access space for temporary data sets can be preallocated to save time in scheduling job steps. This facility, however, cannot be used with checkpoint/restart. Checkpoints and automatic restarts are suppressed for any job step that uses a preallocated temporary data set.

For detailed information on preallocated data sets, refer to the *OS MVT Guide* or the *OS MFT Guide*.

### ***SYSIN Data Sets***

When restart at a checkpoint occurs, a SYSIN data set (data following a DD \* or DD DATA statement) is repositioned. Unit record data sets are never repositioned. When automatic restart is occurring, the system keeps the direct-access data sets that contain the SYSIN data of the job being restarted. During the restart execution, the job can read data from the direct-access data sets as it could during the original execution.

To perform deferred restart, the programmer includes any necessary SYSIN data in the resubmitted deck. If the restart is to be a checkpoint/restart, and a SYSIN data set was open and not completely read at the checkpoint to be used, the attributes of the direct-access data set, into which the system will write the SYSIN data and from which the data will be read by the user's program, must be the same as the attributes of the direct-access data set used originally. (The extents and number of extents in the data set used during restart need not be the same as those in the data set used originally.)

Information about altering SYSIN data in a restart deck is given in Chapter 4, the sections "Automatic Restarts" and "Deferred Checkpoint Restart," and the subsection, "JCL Requirements and Restrictions." Information about repositioning data sets during checkpoint/restart is given earlier in this chapter under "Repositioning User Data Sets."

### ***SYSOUT Data Sets***

The following discussion is about how SYSOUT data sets (data sets having the SYSOUT parameter coded on their DD statements) are handled during the various types of restart.

#### **Automatic Restart**

1. With direct system output (DSO), the user's program writes SYSOUT data directly onto a printer, card punch, or magnetic tape unit. None of these devices is repositioned during restart; therefore, data written during the restart execution does not overlay any of the data written during the original execution. All data written during the original and restart executions is printed or punched and made available to the programmer.
2. Without direct system output (DSO), the user's program writes SYSOUT data into one or more direct-access data sets. If step restart is occurring, the direct-access data sets used during the original execution are deleted. New direct-access data sets are allocated when the restart step is reinitiated.

If checkpoint/restart is occurring, the data sets used during the original execution are kept. Then if a SYSOUT data set was open at the time that the last checkpoint was taken, the data set is repositioned to its position at that time. Data written during the restart execution overlays only the data written between the time the last checkpoint was taken and the time the job step terminated abnormally. If a SYSOUT data set was closed at the checkpoint, the data set is not repositioned. If the restart step opens the same data set again, the data written during the restart follows the data written originally. (The data set has implied MOD disposition.)

### **Deferred Checkpoint/Restart**

1. When checkpoint/restart occurs, and a SYSOUT data set is open at the checkpoint, the data set written into during the restart is different from the data set used originally. The system writes data set header labels and job separators at the beginning of the data set used during the restart. Header labels are written only for direct system output (DSO) on tape. Data written by the restart step follows the job separators.
2. To perform a deferred checkpoint/restart of a step in which a SYSOUT data set was open at the checkpoint, direct system output (DSO) must be used for each data set for which it was used originally, and the device type must be the same. When DSO is not used, the attributes (device type and blocking factor) of the direct-access data set allocated to the restart step must be the same as the attributes of the data set allocated originally. (The extents and number of extents in the data set used during restart need not be the same as those in the data set used originally.)

Information about repositioning data sets during restart at a checkpoint is given earlier in this chapter in “Repositioning User Data Sets.”

### ***SYSABEND Data Sets***

Whether or not the checkpoint/restart facility is used, abnormal termination will cause the system to write a SYSABEND (or SYSUDUMP) data set if the programmer provides a SYSABEND (or SYSUDUMP) DD statement. The system uses its own data control block to write the data set, and it opens the data set during abnormal termination processing. The programmer may either code or omit the SYSOUT parameter on the SYSABEND DD statement.

When the SYSOUT parameter is coded and automatic restart occurs after abnormal termination, the SYSABEND or SYSUDUMP data set will not be printed for step restart without direct system output (DSO). Because the SYSABEND or SYSUDUMP data set was created by the job step, it is deleted during restart.

In all other cases, the SYSABEND or SYSUDUMP data set is printed, whether or not the restart is successful. If a second abnormal termination occurs, a second SYSABEND or SYSUDUMP data set is written. The second data set is always printed, assuming that a second restart does not occur. If a second restart does occur, the second data set is printed except as described above.

## CHAPTER 4: HOW TO REQUEST RESTART

This chapter explains how a user may request restart. The topics discussed are:

- RD (restart definition) parameter
- Restart parameter
- SYSCHK DD statement
- Automatic restart
- Deferred step restart
- Deferred checkpoint/restart

### RD (Restart Definition) Parameter

The RD parameter is coded in the JOB or EXEC statements and is used to request that an automatic step restart be performed if failure occurs and/or to suppress, partially or totally, the action of the CHKPT macro instruction. If the RD parameter is used simply to request that an automatic step restart be performed if failure occurs, or if the RD parameter is not coded, the action of CHKPT is normal. (CHKPT writes a checkpoint entry and requests a checkpoint/restart to be performed if failure occurs.)

When coded on an EXEC statement, the RD parameter applies to the step corresponding to the statement or to all steps of the cataloged procedure referred to by the statement. When coded on a JOB statement, the RD parameter applies to all steps of the corresponding job and overrides an RD parameter coded in any EXEC statements of the job. The parameter syntax is:

---

**RD[.procstepname] = {R/NC/NR/RNC}**

---

The possible definitions are:

**RD = R**

(Restart) Requests an automatic step restart to be performed if failure occurs. If the CHKPT macro instruction is executed in the step, the resulting request for an automatic checkpoint/restart overrides the request for an automatic step restart.

**RD = NR**

(No Automatic Restart) Does not request an automatic step restart, and suppresses the request for an automatic checkpoint/restart that would otherwise be made when the CHKPT macro instruction is executed in the step. If CHKPT is executed, it writes a checkpoint entry normally. The checkpoint entry can be used to perform a deferred restart.

**RD = NC**

(No Checkpoint) Does not request an automatic step restart, and totally suppresses the action of the CHKPT macro instruction if the macro instruction is executed in the step. This allows a program containing CHKPT to be used when the action of CHKPT is not wanted.

**RD = RNC**

(Restart and No Checkpoint) Requests an automatic step restart to be performed if failure occurs, and totally suppresses the action of CHKPT if CHKPT is executed in the step.

If RD= *value* is coded on an EXEC statement that invokes a cataloged procedure, the parameter applies to all steps of the procedure and overrides all RD parameters present in the EXEC statements of the procedure. RD.procstepname= *value* can be coded instead of RD= *value*; it applies to the specified procedure step and overrides the RD parameter that may be coded on the EXEC statement of the procedure step. RD.procstepname= *value* can be coded once for each step of the procedure.

**RESTART Parameter**

The RESTART parameter is used to perform a deferred restart of a job. It is coded in the JOB statement when the job is resubmitted. If step restart is to occur, this parameter is used to specify at which step to begin. If the restart is to occur at a checkpoint that was taken during a step, both the step and the identification of the particular checkpoint entry are specified. The syntax of the parameter is:

---

```
RESTART = ( {stepname           }[,checkid])
            {stepname.procstepname }
            {*                   }
```

---

Both operands are used if restart at a checkpoint is to occur. If a step restart is to occur, checkid must be omitted; the enclosing parentheses may be omitted.

The stepname parameter is coded as stepname.procstepname if a step of a cataloged procedure is to be restarted. The parameter can be coded as \* if the first step of the job (possibly a step of a cataloged procedure) is to be restarted.

The checkid can contain up to 16 characters in any combination of alphameric characters, printable special characters, and blanks. If it contains any special characters or blanks, it must be enclosed in single apostrophes, and apostrophes within it must be represented as double apostrophes.

**SYSCHK DD Statement**

The SYSCHK DD statement is used in the resubmitted job to perform a deferred checkpoint/restart and specifies the checkpoint data set that contains the checkpoint entry to be used in the restart. The statement may not be included when a deferred step restart is to be performed. The statement is not needed when an automatic restart at the last checkpoint occurs, because in that case, the system knows the identity and location of the checkpoint data set. (Another DD statement describing the checkpoint data set is always included if the program executes the CHKPT macro instruction.)

The statement must immediately precede the first EXEC statement in the deck that is submitted to perform a deferred restart at checkpoint. It must follow the JOBLIB DD statement if the JOBLIB DD is present. The SYSCHK DD must describe the checkpoint data set that contains the checkpoint entry to be used to perform the restart. The desired checkpoint entry must be named by the checkid subparameter of the JOB statement RESTART parameter.

The following requirements and restrictions apply to the SYSCHK DD statement:

- The statement must contain or imply DISP=(OLD, KEEP).
- The statement must define the checkpoint data set in a normal way. For example, it must specify its name, device type, and volume serial number. The catalog may be used.
- If the checkpoint data set is multivolume, the SYSCHK DD must specify, as the first volume of the data set, the volume containing the desired checkpoint entry. The serial number of the volume containing a particular entry is shown in the console message that is written when the entry is written.
- If the checkpoint data set is on a 7-track magnetic tape having nonstandard labels or no labels, the SYSCHK DD must contain DCB=TRTCH=C.
- If the checkpoint data set is partitioned, the DSNAME parameter on the SYSCHK DD must not contain a member name.
- If a RESTART parameter without the checkid subparameter is included in a job, a SYSCHK DD must not appear before the first EXEC statement of the job.
- If a RESTART parameter is not included in a job, a SYSCHK DD appearing before the first EXEC statement in the job is ignored.
- A SYSCHK DD appearing in a step or procedure step of a job is treated as an ordinary DD statement; that is, the name SYSCHK has no special meaning in that case.

An example of a SYSCHK DD statement is:

```
//SYSCHKDD      DSNAME=dsname,DISP=OLD,UNIT=name,          X
//              VOLUME=SER=volser
```

## Automatic Restarts

Because automatic step restart and checkpoint/restart are similar in many ways, they are discussed together, where possible, in the information that follows.

### *Requirements for Automatic Restart to Occur*

Automatic step restart or checkpoint/restart will occur if all of the following conditions are met:

- The step requests restart.
- The step is eligible for restart because it was terminated by an ABEND macro instruction that emitted an eligible completion code (specified by the CKPTREST macro instruction), or because system failure occurred.
- The operator authorizes the restart. This authority enables the operator to prevent repeated restarts of the same step or at the same checkpoint.

### *How to Request Automatic Step Restart*

If a step fails when automatic step restart is requested, restart occurs automatically at the beginning of the step that failed.

Automatic step restart is requested by coding the RD parameter (RD=R or RNC) on either the JOB or EXEC statement in the originally submitted job deck. The CHKPT macro instruction is suppressed if RD=RNC.

Figure 8 illustrates a job requesting automatic step restart.

---

```
//MYJOB   JOB      MSGLEVEL=1, RD=R   Requests automatic restart at
                                           the beginning of any step
                                           that terminates abnormally
//STEP1   EXEC
//STEP2   EXEC      RD=R1           Requests automatic restart of
                                           STEP2 if it terminates abnormally
//STEP3   EXEC
```

<sup>1</sup>Note that if RD = R appears on the JOB statement, it is not required on the EXEC statement.

Figure 8. Requesting Automatic Step Restart

---

### ***How to Request Automatic Checkpoint/Restart***

If a step fails and automatic checkpoint/restart is requested, restart occurs automatically at the last checkpoint taken.

Execution of the CHKPT macro instruction requests this type of restart and establishes the checkpoint. The user must provide an ordinary DD statement for the checkpoint data set.

RD=R may be omitted or included. If it is included and the step fails before or during the time when the first checkpoint is taken, an automatic step restart will occur. Automatic step restart will also occur when RD=R is coded if the last execution of the CHKPT macro instruction specified that a request for checkpoint/restart should be cancelled.

Figure 9 illustrates a job requesting automatic restart at a checkpoint.

---

```
//MYJOB   JOB      MSGLEVEL=1
//STEP1   EXEC
.
.
.
//STEP2   EXEC      PGM=MYPROG   MYPROG issues the CHKPT macro
//NAME1   DD        DSNAME=NAME2 Describes the data set into which
                                           checkpoint entries are to be written
```

Figure 9. Requesting Automatic Checkpoint/Restart

---

### ***JCL Requirements and Restrictions***

To allow occurrence of an automatic step restart or automatic checkpoint/restart, the programmer must observe the following rules when he prepares the job deck used in the original execution:

- If a step restart is desired, the RD parameter must be coded to request the restart.
- If a checkpoint/restart is desired, a DD statement for the checkpoint data set must be included in the step that executes the CHKPT macro instruction.

- The EXEC statements in the job deck must have unique names. (Upon restart, the system searches for a named step.)
- MSGLEVEL=1 must be coded on the job's JOB statement. (MSGLEVEL=1 produces internal records that are reinterpreted when restart occurs.)
- If commands are included in the original deck, the commands are not reexecuted when restart occurs.
- If a procedure used in the restarting step is in a private library other than SYS1.PROCLIB, the Restart Reader Procedure (IEFREINT) must be modified to indicate the private library.
- If an automatic restart is being performed, the default values for SYSOUT data sets will be: UNIT=SYSDA,SPACE=(TRK,(50,100)).

### ***Resource Variations Allowed in Automatic Restart***

The system's device and volume configuration during a restart execution of a job can be different from what it was during the original execution of the job.

The ability to use a different volume usually exists only in the case of a new data set on a nonspecific volume. Furthermore, if a checkpoint/restart is to be performed, the data set must not have been open at the checkpoint. The ability to use a different device does not apply to the device or devices containing the SYSRES volume and the SYSJOBQE and LINKLIB data sets. Also, if a checkpoint/restart is to be performed, the same type of device must be allocated to the data set during both the original and restart executions.

### ***How the System Works at Automatic Restart***

**How Data Set Disposition is Determined:** When a step requests restart and is eligible for restart, disposition processing of the data sets used by the step or by the job does not occur until the operator has replied to the request for authorization. If the operator denies restart, disposition processing occurs normally; that is, programmer-specified final or conditional dispositions are performed and if the programmer requested that a step be executed after abnormal termination, the step is executed. If the operator authorizes automatic restart, the following special disposition processing is performed:

- If step restart is to occur, all data sets having OLD or MOD dispositions in the restart step and all data sets being passed around the restart step are kept, even if they have been declared to be temporary. Temporary data sets normally cannot be kept.
- All data sets having NEW dispositions in the restart step are deleted.
- If checkpoint/restart is to occur, all data being used by the job (data sets that were not previously disposed of) are kept.

If the operator authorizes restart, execution of the step to be executed after abnormal termination will not occur because, in effect, abnormal termination did not occur.

If the operator performs an operator-deferred restart by replying HOLD to the request for authorization, he later may issue a CANCEL command for the job instead of a RELEASE command. If he issues CANCEL, no further data set disposition processing or step executions will occur. Thus, the disposition of these data sets remains as it was when the HOLD was issued.

**How the Job Deck Is Reinterpreted and the Input Work Queue Merged:** When it has completed disposition processing for a terminated job that is to be restarted, the system begins the restart by interpreting the job deck again. The system uses its internal records of the job (System Message Blocks), and the job is not read again.

After it has reinterpreted the job deck, the system merges information from the newly formed input work queue entry for the job (on SYS1.SYSJOBQE) into the original one, and then destroys the newly formed entry. The system inserts a special step before the restart step in the job. The special step, named IEFDSDRP, is executed first; it reads the last checkpoint entry and merges information from it into the original input work queue entry.

When the information is merged, and if a step restart is occurring, the input work queue entry is the same as it was before the original initiation of the restart step. If checkpoint/restart is occurring, the input work queue entry differs from its original form in these ways:

- Data sets specified as NEW in the restart step have had their dispositions changed to OLD, except in the case of data sets that were not opened during the original execution and for which nonspecific tapes were requested.
- In the case of data sets for which nonspecific volumes were requested in the restart step, the work queue entry describes the device type and serial numbers of the volumes assigned to the data sets during the original execution.
- In the case of multivolume data sets, the work queue entry indicates which volumes were being processed at the checkpoint. These volumes, and not the first volumes of the data sets, will be mounted (if they have not remained mounted) during the restart.

### **How Step Restart Is Initiated**

A step being restarted is initiated in the same way as it would be during a normal execution. Therefore, the devices allocated to the restart step can be different (but of the same device type) from the devices allocated originally. If the allocated devices differ, volumes must be moved from one device to another. If AVR is used, devices containing the required volumes are allocated, if the devices are available for allocation.

After devices have been allocated to the restart step, normal mounting messages request the operator to mount the required volumes on the devices, unless the volumes are already mounted. The volumes requested are those on which processing is to be resumed.

### **How Checkpoint/Restart Is Initiated**

If checkpoint/restart is occurring, the restart step must be executed in the main-storage area that was used during the original execution. If the required main storage is allocated to another step before it is reallocated to the restart step, the restart is delayed until the other step terminates.

In MFT, the partition in which a job is originally executed may be redefined before the job is restarted. The partition used for restart must be at least as large as the original partition. If the original partition included only storage from hierarchy 1, the partition used for restart must not include any storage from hierarchy 0.



After it has initiated a step being restarted at a checkpoint, the system reads the checkpoint entry again. The system uses the contents of the entry to restore main storage and to reposition data sets that were being processed at the checkpoint.

To save time during restart, data sets on magnetic tape are repositioned in parallel. The number of data sets that can be repositioned at one time is four or more, depending on the amount of available main storage. To be precise,  $N+4$  data sets can be repositioned in parallel, where  $N$  is the number of contiguous 128-byte areas that can be obtained by the control program.

### ***How MOD Data Sets Are Handled During Automatic Step Restart***

When automatic step restart has been requested for a step, the system saves, for each MOD data set that is on a direct-access volume and used by the step, the TTR (and track balance) of the end of the data set. Saving occurs when each data set is first opened. If restart occurs, the saved TTRs are used to indicate the ends of the data sets when the data sets are first opened again. Thus, if the step writes data in such a data set during the original execution, the step will write over the data during the restart. The action described here does not take place if restart at a checkpoint occurs.

If a MOD data set on tape is used in the restart step, the data set is not repositioned at the start of the restart execution. Therefore, data written into it during the restart execution follows the data written during the original execution. The programmer may wish to reposition the data set so that the data written during the restart execution overlays the data written during the original execution.

**Caution Concerning Automatic Step Restart After Checkpoint/Restart:** If a step is executing as the result of an automatic or deferred checkpoint/restart, and if you attempt an automatic step restart of this step, the attempt may be unsuccessful if the JCL of the step refers to any new data sets on direct-access volumes. When the step is initiated during the checkpoint/restart, the failure occurs because all the step's data sets that have a NEW disposition are changed to a disposition of OLD by the system. Therefore, when the special disposition processing that prepares for a step restart occurs, all data sets used by the step appear to be OLD and are kept. When the step restart occurs, the scheduler tries to obtain space for data sets specified as NEW in the JCL for the step. If the attempt for data set space is made on the volume that already contains the data set, the failure occurs because of the apparent presence of a "duplicate DSCB on direct-access volume."

## **Deferred Step Restart**

### ***How to Request Deferred Step Restart***

The programmer causes a deferred step restart of a job by coding the RESTART parameter on the JOB statement and then by resubmitting the job. The parameter specifies a job step, or a step of a cataloged procedure. The effect of the parameter is simply to restart the job at the beginning of the specified step. Steps preceding the restart step are interpreted, but not initiated.

---

		<b>Original Deck</b>	
//MYJOB	JOB	MSGLEVEL=1	No automatic restart requested
//STEP1	EXEC		
		.	
		.	
//STEP2	EXEC	PGM=MYPROG	
		.	
		.	
//STEP3	EXEC		
<b>Resubmitted Deck</b>			
//MYJOB	JOB	MSGLEVEL=1,RESTART=STEP2	Causes restart of job at STEP2
//STEP1	EXEC		
		.	
		.	
//STEP2	EXEC	PGM=MYPROG	
		.	
		.	
//STEP3	EXEC		

Figure 10. Requesting a Deferred Step Restart

---

The CHKPT macro instruction may or may not be coded in the user's program. Figure 10 illustrates a job as it is originally submitted and the same job as it is resubmitted for step restart. Assume that the results of STEP2 were unsatisfactory due to abnormal termination or incorrect data when the job was executed originally.

### ***JCL Requirements and Restrictions***

To perform a deferred step restart, the user must provide the data set environment required by the restart job. This may be accomplished by using the conditional disposition subparameter in the appropriate DD statements during the original execution of the job. Conditional dispositions in the original deck should be used to:

- Delete all NEW data sets used by the step to be restarted.
- Catalog all data sets that are passed from steps preceding the restart step to the restart step or to steps following the restart step. Abnormal termination of the restart step, when it is originally run, will then cause the passed data sets to be cataloged. Thus, the information will be available to the following steps when the deck is resubmitted.
- Keep all OLD data sets used by the restart step, other than those passed to the step.

If a MOD data set on tape is used in the restart step, the data set is not repositioned at the start of the restart execution and thus data written into it during the restart execution follows the data written during the original execution. The programmer may wish to reposition the data set so that the data written during the restart execution overlays the data written during the original execution.

The following rules apply to the restart deck:

- The RESTART parameter must be coded on the JOB statement.
- If data sets are passed from steps preceding the restart step to the restart step or to steps following the restart step, the DD statements used to receive the data sets must entirely define the data sets. They must explicitly specify volume serial numbers, device type, data set sequence number and label type, unless this information can be retrieved from the catalog. This is why it is recommended that passed data sets be conditionally cataloged during abnormal termination of the original execution. Note that label type cannot be retrieved from the catalog.
- Generation data sets created and cataloged in steps preceding the restart step must not be referred to in the restart step or in steps following the restart step by the relative generation numbers used to create them. They must be referred to by their actual relative generation numbers. For example, a data set created as the +1 data set must be referred to as the 0 data set (assuming that the +2 data set was not also created).
- The EXEC statement PGM and COND parameters and the DD statement SUBALLOC and VOLUME=REF parameters must not be used in the restart step or in steps following the restart step if they contain values of the form stepname or stepname.procstepname, referring to a step preceding the restart step.

### ***Resource Variations Allowed in Deferred Step Restart***

A deferred step restart merely allows the restarted execution of a job to begin at other than the first step of the job. Therefore, job step initiation and allocation of resources are accomplished normally. The following variations are allowed upon restart:

- Variation of device and volume configuration
- Variation in JCL and data in the resubmitted deck
- Restart on an alternate system (for example, MFT instead of MVT) or a system having a different main-storage configuration and resident contents

## **Deferred Checkpoint/Restart**

### ***How to Request Deferred Checkpoint/Restart***

The programmer causes a deferred checkpoint/restart of a job by the following procedure:

- He has the option of coding a special form of the RD parameter (RD=NR) in the original job deck. This specifies that if the CHKPT macro instruction is executed, a checkpoint entry is to be written, but an automatic checkpoint/restart is not to be requested.
- He causes execution of the CHKPT macro instruction, which writes a checkpoint entry.
- The programmer resubmits the job whether or not it terminated abnormally. For example, he might resubmit it because a volume of one of its input data sets was in error and had caused the corresponding part of an output data set to be in error.

- The programmer codes the RESTART parameter (RESTART=(stepname, checkid)) on the JOB statement of the restart deck. Thus, the parameter specifies both the step to be restarted and the checkid that identifies the checkpoint entry to be used to perform the restart.
- He places a SYSCHK DD statement immediately before the first EXEC statement in the restart deck. It specifies the checkpoint data set from which the specified checkpoint entry is to be read and is additional to any DD statements in the deck that define data sets into which checkpoint entries are to be written. Figure 11 illustrates a job when it is originally submitted and when it is resubmitted for a deferred checkpoint/restart. Assume in Figure 11 that STEP2, when originally executed, terminates abnormally at some time after CH04 has been written. Note that, in the resubmitted deck, the programmer requests that STEP2 be restarted using the checkpoint entry identified as entry CH04.

---

		<b>Original Deck</b>	
//MYJOB	JOB	RD=NR	Requests that automatic restart not occur (optional)
//STEP1	EXEC		
		.	
		.	
//STEP2	EXEC	PGM=MYPROG	MYPROG issues CHKPT macro
//NAME1	DD	DSNAME=NAME2	Describes checkpoint data set
		.	
		.	
//STEP3	EXEC		
<b>Resubmitted Deck</b>			
//MYJOB	JOB	RESTART=(STEP2,CH04)	Request restart at CH04 in STEP2
//SYSCHK	DD	DSNAME=NAME2	Describes data set which contains CH04
//STEP1	EXEC		
		.	
		.	
//STEP2	EXEC	PGM=MYPROG	
//NAME1	DD	DSNAME=NAME2	Describes data set in which new checkpoint entries will be written
		.	
		.	
//STEP3	EXEC		

Figure 11. Requesting a Deferred Checkpoint/Restart

---

### ***JCL Requirements and Restrictions***

To perform a deferred checkpoint/restart the programmer must provide the data set environment required by the restart job. He may do this by using conditional dispositions during the original execution.

Conditional dispositions should be used to:

- Keep all data sets used by the restart step.
- Catalog all data sets being passed from steps preceding the restart step to steps following the restart step. Even though the step that terminates abnormally is not using the passed data sets, its termination will cause the cataloging of the data sets if the conditional catalog parameter is used in the preceding steps.

Note that temporary data sets cannot be kept.

The following rules must be adhered to when resubmitting a job for deferred checkpoint/restart:

1. A RESTART parameter with a checkid subparameter must be coded on the JOB statement.
2. A SYSCHK DD statement must be placed in the job deck immediately before the first EXEC statement.
3. The EXEC statements in the job deck must have unique names. (The system searches for the named restart step.)
4. The JCL statements and data in steps preceding or following the restart step can be different from their original forms. However, all backward references must be resolvable.
5. The restart step must have a DD statement corresponding to each DD statement present in the step in the original deck, and the names of the statements must be the same as they were originally. However, the restart step can contain, in any position, more DD statements than it contained originally. However, the total number of volumes specified at restart must equal or exceed the number specified originally at checkpoint.
6. If a DD statement in the restart step in the original deck defined a data set that was open at the checkpoint to be used, the corresponding statement in the restart deck must refer to the same data set, and the data set must be on the same volume and have the same extents recorded in its DSCB as it did originally.<sup>1</sup> If the data set is multivolume and was being processed sequentially, only the part of the data set on the volume in use at the checkpoint need be the same as it was originally.

When there is no need to read or modify a data set after restart, the data set can be replaced by a dummy data set if the original data set was processed sequentially and the job step is not restarted from a checkpoint within the data set's end-of-volume exit routine. Of course, the data set must not be the checkpoint data set used to restart the job step. Allocation will be done for each DD statement in the job step where the checkpoint was taken, even if the data set was closed at the time of the checkpoint.

---

<sup>1</sup>The extents can differ as follows: In the DD statement, the user can request that additional space be allocated to the data set when the space currently available is exhausted. If space is allocated after a checkpoint is taken, this space is indicated in the DSCB; on restart from the checkpoint, the space is released and the DSCB contents are changed to what they were at the checkpoint.

In the DD statement, the user can request that unused space be released at the end of the job step. If the space is released, the DSCB may indicate a reduced extent for the data set when deferred restart at a checkpoint occurs; no space is allocated to replace that which was released. Note that space is not released when step termination is followed by automatic restart.

7. Data in the restart step need not be the same as it was originally. If data following a DD \* statement was present originally and is entirely omitted in the restart deck, the delimiter (/\*) statement following the data may also be omitted. The delimiter statement following a DD DATA statement may not be omitted.
8. Except for the requirements stated in rules 4 through 7, the JCL statements and data in the restart step can be different from their original forms. In particular, the DUMMY parameter can be used for any data set that was not open at the checkpoint.
9. If a SYSIN data set is open at the checkpoint to be used and is to be read during the restart, the attributes of the direct-access data set into which the SYSIN data will be written must be the same as they were originally.
10. If data sets are passed from steps preceding the restart step to steps following it, the DD statements receiving the data sets must entirely define them. They must explicitly specify volume serial numbers, device type, data set sequence number, and label type, unless this information can be retrieved from the catalog. This is why it is recommended that passed data sets be conditionally cataloged during abnormal termination of the original execution. Note that label type cannot be retrieved from the catalog.
11. The EXEC statement PGM and COND parameters and the DD statement SUBALLOC and VOLUME=REF parameters must not be used in steps following the restart step if they contain values of the form stepname or stepname.procstepname referring to a step preceding the restart step.

### ***Resource Variations Allowed in Deferred Checkpoint/Restart***

The system's device and volume configuration can be different from what it was during the original execution of the job. The allowable differences are those described earlier in this chapter under "Resource Variations Allowed in Automatic Restart."

Deferred checkpoint/restart also allows restart on an alternate system. However, the restrictions applying to use of an alternate system are such that they usually will prevent use of a given system as an alternate system. The restrictions are:

- The type (MFT or MVT) and release number of the system must be the same as those of the original system.
- The nucleus of the alternate system must be identical to that of the original system.
- If the checkpoint entry to be used was written by a CHKPT macro instruction located in an end-of-volume exit routine, SYS1.SVCLIB must be the same upon restart as it was originally.
- When MFT is used, the main-storage partition must be at least as large as that in which the job was originally executed, and must include all of the same locations. If the original partition included only storage from hierarchy 1, the partition used for restart must not include any storage from hierarchy 0. If resident access method modules or resident modules from the link library were being used at the checkpoint, they must occupy exactly the same main-storage locations as they did originally.

- When MVT is used, if link pack area modules were being used at the checkpoint, they must occupy exactly the same main-storage locations. Note that link pack area modules may include access method modules and modules made available to a program when it executes the LOAD macro instruction.
- In a system with the IBM 2361 Core Storage device, the boundary between hierarchy 0 and hierarchy 1 must be the same as it was originally.

### ***How the System Works During Deferred Checkpoint/Restart***

After the system has read and interpreted the restart deck, it reads the specified checkpoint entry and merges information from it into the input work queue entry for the job. As a result, the work queue entry differs from the entry existing during the original execution, as described earlier in this chapter. (Refer to “How the Job Deck is Reinterpreted and the Input Work Queue Merged” in the section “Automatic Restarts,” subsection “How the System Works at Automatic Restart.”)

Next, the system initiates the restart step normally. The system reads the specified checkpoint entry again and functions as in the automatic restart case. Restart is delayed until the required main-storage area is available.





## CHAPTER 5: WHAT THE OPERATOR MUST CONSIDER

This chapter describes the system messages and operator functions during various types of restarts and includes discussions on how the operator's decisions and choice of commands can cause variations in the use of system resources. The chapter is divided into two parts: one on the MVT environment, the other on the MFT environment.

### MVT Environment

#### *Automatic Restart Message Sequence*

During processing related to automatic checkpoint/restart in MVT, the system issues the following sequence of messages to the operator:

1. A message each time a checkpoint entry is written. Each message contains the checkpoint identification.
2. If the job step terminates because of an ABEND condition, an ABEND message for the job step.
3. If the ABEND code makes the job step eligible for restart, an authorization for restart message that requires a reply.
4. Assuming that restart is authorized and DISPLAY JOBNAMEs is in effect, an IEFREINT STARTED message, followed by an IEFREINT ENDED message. IEFREINT is the name of a system task called the "restart reader." The restart reader reinterprets internal system records of the job to be restarted.
5. A message indicating the main-storage requirements (beginning address and ending address) of the job step to be restarted. This allows the operator to determine that the required main storage is not currently in use by a "never ending" task.
6. A message indicating direct system output (DSO) requirements. (If this message is written, the job is placed on the HOLD queue.)
7. Normal mount messages.
8. A successful restart message.

During processing related to an automatic step restart after a job step has terminated abnormally in MVT, the sequence is the following:

1. An ABEND message for the job step.
2. If the ABEND code makes the job step eligible for restart, an authorization message that requires a reply.
3. Assuming that restart is authorized and DISPLAY JOBNAMEs is in effect, an IEFREINT STARTED message followed by an IEFREINT ENDED message.
4. Normal mount messages.

Note that the ABEND message, which is issued as:

```
IEF4501 jobname.stepname.procstepname ABEND code
```

is always displayed if a job step terminates abnormally. In addition, if the job step is being executed and the MVT system fails, this message will be displayed during the next IPL if system-supported restart is performed. The "code" part of the message has the form Shhh (S followed by a three character hexadecimal number) if the system executed the ABEND macro instruction, or Udddd (U followed by a four-digit decimal number) if the user's program executed the ABEND. It is S2F3 if MVT system failure occurred.

### ***Operator Options During Automatic Restart***

In MVT, if a step requests automatic restart and is eligible for restart, the system displays the following message to request authorization for the restart:

```
xxIEF225D SHOULD jobname.stepname.procstepname[checkid]RESTART
```

Checkid appears in the message only if restart at a checkpoint is requested. It contains from 1 to 16 characters and identifies the checkpoint entry to be used to perform the restart. The operator must reply to the request for authorization as follows:

```
      {'YES'}  
REPLY xx, {'NO'}  
      {'HOLD'}
```

YES authorizes the restart, HOLD postpones it, and NO prohibits it. During the time that the MVT system is waiting for the operator to reply to the authorization request, no other task in the system can be initiated or terminated. Therefore, the operator should reply promptly to this message.

If the advisability of allowing the restart is not readily apparent, the operator should reply HOLD to the authorization message. If he later determines that the restart should occur, he can initiate the restart by using the RELEASE command, thereby achieving the same result as with an initial YES reply. If the decision is to deny the restart authorization, the operator can cancel the job in the HOLD queue. However, he must consider that HOLD, as well as YES, causes special disposition processing to occur during the abnormal termination. This processing keeps all OLD (or MOD) data sets and deletes all NEW data sets if a step restart was requested and keeps all data sets if a checkpoint/restart was requested. If the operator subsequently decides to disallow the restart but wants to allow normal disposition (as requested on the job's DD statements) of data sets that were kept, he may release the job, wait until restart has begun, and then cancel the job.

After the authorization request and before the operator replies YES, he may, in some cases, by using the VARY and UNLOAD commands, cause the system's volume and device configuration during a restart execution of the job to be different from what it was during the original execution of the job. Thus, the operator may eliminate use of defective volumes and devices.

The ability to use a different volume usually exists only in the case of a NEW data set on a nonspecific volume. Furthermore, if a checkpoint/restart is to be performed, the data set must not have been open at the checkpoint. The ability to use a different device does not apply to the device or devices containing the SYSRES volume and the SYSJOBQE and LINKLIB data sets. Also, if a checkpoint/restart is to be performed and a data set was open at the checkpoint, the same type of device must be allocated to the data set during both the original and restart executions.

After a YES reply, the job is reinterpreted by a restart reader, named IEFREINT, that is started automatically by the system. At this time, the IEFREINT STARTED and IEFREINT ENDED messages are issued to the operator if DISPLAY JOBNAMEs is in effect. Before the restart job is reinterpreted and is ready for reinitiation, one or more initiators may select other jobs from the work queue and initiate them. The other jobs may use the main storage and devices needed by the restart job and, if they do, the restart will be delayed until the main storage and devices are available. If a delay of the restart is undesirable, the operator can hold the queue prior to the YES reply and release the queue after the IEFREINT ENDED message is displayed. This ensures that jobs with the same priority are executed in the sequence in which they were originally submitted.

If a job is to be restarted at a checkpoint, a message specifying the beginning and ending addresses of the main storage required for the job step to be restarted is issued after job reinterpretation and any IEFREINT messages. If the required main-storage area is currently unavailable because it is being used by other tasks, the restart is delayed until the area is available. If neither the mount messages nor the successful restart message is issued, it is an indication that the required area is currently unavailable. The operator can determine the status of the required area by using the DISPLAY A (Active) command. If a system task is executing in the required area, the operator can either allow the system task to continue to termination (if a reader), or issue the STOP command for the system task (if a reader or writer). If the area is occupied by another job step task, the operator can permit the job step task to continue to termination or he can cancel it.

If the required main-storage area is unavailable because the System Queue Space has expanded into that area since the original execution, a message is displayed indicating that main storage for the job step to be restarted is unavailable and the restart is terminated. In order to successfully restart the job step from the specified checkpoint, the IPL procedure must be performed again and the job resubmitted for a deferred checkpoint/restart.

Note that when an initiator has selected a job for automatic step restart and the job has been reinterpreted, no message is issued to the operator regarding main storage requirements since its execution is not location dependent.

After the required main storage has been allocated, the following message may be issued if the job is to be restarted at a checkpoint:

```
IEF390E DSO (outputclass,jobclass,devicetype)
        NEEDED TO RESTART jobname
```

The message indicates that a DSO (direct system output) data set was open at the checkpoint. The data set was part of the specified system output class, and was assigned to a printer, card punch, or magnetic tape unit, as indicated by the message. The device originally used by the data set is no longer available because:

- The operator issued a STOP command to stop DSO processing on the device.
- The operator issued a MODIFY command to assign the device to a different system output class.
- After abnormal termination, the DSO device was allocated to a step belonging to another job.

The operator can assign a device to the required system output class by issuing a START or MODIFY command. The START command starts DSO processing on a

new device; the MODIFY command changes the system output class for a device that has already been started. If the operator issues a MODIFY command for a DSO device being used by another job, the command will take effect when that job terminates.

When starting or modifying DSO processing, the operator can specify a job class for which no initiator has been started. Doing so allows the DSO device to be used by the job that is being restarted, but not by other jobs that may be initiated before restart can occur. Of course, this is not necessary if the operator has issued the HOLD command to temporarily prevent the initiation of other jobs.

Message IEF390E is issued once for each output class that requires DSO device assignment. The job is then placed on the HOLD queue. The operator must release the job for execution after assigning the required devices. If the required devices cannot be assigned, the operator should cancel the job.

### ***Deferred Restart Message Sequence***

To perform a deferred checkpoint/restart in MVT, the job to be restarted is resubmitted in an input job stream. Messages that contain checkpoint entry identifications were displayed on the console during the original execution of the job and may then be used by the programmer preparing the job for resubmission. When the resubmitted job is restarted, messages appear on the console in the following sequence:

1. A message indicating the main-storage requirements of the job
2. When a direct system output (DSO) device must be started, a message indicating DSO requirements
3. Normal mount messages
4. A successful restart message

To perform a deferred step restart in MVT, the job to be restarted is resubmitted. Normal mount messages are displayed.

### ***Operator Considerations During Deferred Checkpoint/Restart***

When a job is resubmitted to perform a deferred checkpoint/restart (the RESTART parameter is coded on the JOB statement with a checkid operand), the processing is essentially the same as during an automatic checkpoint/restart after the restart reader has reinterpreted the job. A message is issued to the operator indicating the main-storage requirements of the job. Other tasks executing in the required main-storage area can delay the restart. The operator can use the DISPLAY A command in the same manner as in an automatic checkpoint/restart.

The required main-storage area can also be unavailable for the following other reasons:

- System Queue Space has expanded into the required area.
- The REGION size parameter for the step is larger when the job is resubmitted than in the original execution, and the area used in the original execution was adjacent to (immediately below) the Master Scheduler Region. Because the area used by the step is not allowed to expand upward into the Master Scheduler Region, the request for a larger region for the step cannot be satisfied.

- A new IPL was performed and, because of different IPL options specified by the operator, the nucleus expanded upward, or the Link Pack Area expanded downward into the required area.
- The job is resubmitted on an alternate system that does not have the required main-storage area.

If these conditions exist, a message is displayed indicating that main storage for the job step to be restarted is unavailable. The restart is terminated.

When main-storage requirements can be satisfied, message IEF390E may be issued to define direct system output (DSO) requirements. Operator response is the same as in the case of automatic checkpoint/restart.

## **MFT Environment**

### ***Automatic Restart Message Sequence***

During processing related to automatic checkpoint/restart in MFT, the system issues the following sequence of messages to the operator:

1. A message each time a checkpoint entry is written. Each message contains the checkpoint identification.
2. If the job step terminates because of an ABEND condition, an ABEND message for the job step.
3. If the ABEND code makes the job step eligible for restart, an authorization for restart message that requires a reply.
4. Assuming that restart is authorized and DISPLAY JOBNAMEs is in effect, an IEFREINT STARTED message, followed by an IEFREINT ENDED message. IEFREINT is the name of a system task called the "restart reader." The restart reader reinterprets internal system records of the job to be restarted.
5. A message indicating direct system output (DSO) requirements. (If this message is written, the job is placed on the HOLD queue.)
6. Normal mount messages.
7. A successful restart message.

During processing related to an automatic step restart after a job step has terminated abnormally in MFT, the sequence is the following:

1. An ABEND message for the job step
2. If the ABEND code makes the job step eligible for restart, an authorization message that requires a reply
3. Assuming that restart is authorized and DISPLAY JOBNAMEs is in effect, an IEFREINT STARTED message followed by an IEFREINT ENDED message
4. Normal mount messages

Note that the ABEND message, which is issued as:

```
IEF4501 jobname.stepname.procstepname ABEND code
```

is always displayed if a job step terminates abnormally. In addition, if the job step is being executed and the MFT system fails, this message will be displayed during the next IPL if system-supported restart is performed. The “code” part of the message has the form Shhh (S followed by a three character hexadecimal number) if the system executed the ABEND macro instruction, or Udddd (U followed by a four-digit decimal number) if the user’s program executed the ABEND. It is S2F3 if MFT system failure occurred.

### ***Operator Options During Automatic Restart***

In MFT, if a step requests automatic restart and is eligible for restart, the system displays the following message to request authorization for the restart:

```
xxIEF225D SHOULD jobname.stepname.procstepname[checkid]RESTART
```

Checkid appears in the message only if restart at a checkpoint is requested. It contains from 1 to 16 characters and identifies the checkpoint entry to be used to perform the restart. The operator must reply to the request for authorization as follows:

```
        {'YES'}  
REPLY xx, {'NO'}  
        {'HOLD'}
```

YES authorizes the restart, HOLD postpones it, and NO prohibits it. During the time that the MFT system is waiting for the operator to reply to the authorization request, no other task in the system can be initiated or terminated. Therefore, the operator should reply promptly to this message.

If the advisability of allowing the restart is not readily apparent, the operator should reply HOLD to the authorization message. If he later determines that the restart should occur, he can initiate the restart by using the RELEASE command, thereby achieving the same result as with an initial YES reply. If the decision is to deny the restart authorization, the operator can cancel the job in the HOLD queue. However, he must consider that HOLD, as well as YES, causes special disposition processing to occur during the abnormal termination. This processing keeps all OLD (or MOD) data sets and deletes all NEW data sets if a step restart was requested and keeps all data sets if a checkpoint/restart was requested. If the operator subsequently decides to disallow the restart but wants to allow normal disposition (as requested on the job’s DD statements) of data sets that were kept, he may release the job, wait until restart has begun, and then cancel the job.

After the authorization request and before the operator replies YES, he may, in some cases, by using the VARY and UNLOAD commands, cause the system’s volume and device configuration during a restart execution of the job to be different from what it was during the original execution of the job. Thus, the operator may eliminate use of defective volumes and devices.

The ability to use a different volume usually exists only in the case of a NEW data set on a nonspecific volume. Furthermore, if a checkpoint/restart is to be performed, the data set must not have been open at the checkpoint. The ability to use a different device does not apply to the device or devices containing the SYSRES volume and the SYSJOBQE and LINKLIB data sets. Also, if a checkpoint/restart is to be performed and a data set was open at the checkpoint, the same type of device must be allocated to the data set during both the original and restart executions.

After a YES reply, the job is reinterpreted by a restart reader, named IEFREINT, that is started automatically by the system. At this time, the IEFREINT STARTED and IEFREINT ENDED messages are issued to the operator if DISPLAY JOBNAME is in effect. Before the restart job is reinterpreted and is ready for reinitiation, one or more initiators may select other jobs from the work queue and initiate them. The other jobs may use the main storage and devices needed by the restart job and, if they do, the restart will be delayed until the main storage and devices are available. If a delay of the restart is undesirable, the operator can hold the queue prior to the YES reply and release the queue after the IEFREINT ENDED message is displayed. This ensures that jobs with the same priority are executed in the sequence in which they were originally submitted.

In some cases, the partition in which a job is originally executed may be redefined before the job is restarted. For example, the operator may redefine the partition after replying HOLD to the restart authorization message, or redefinition may already be pending when the message is issued. The redefined partition may be unsuitable for use in restarting the job at a checkpoint because:

- The required main-storage area may be split between two or more partitions, or may be allocated to a resident reader or writer.
- The redefined partition may include storage from hierarchy 0 when it originally included storage only from hierarchy 1.

In either case, the system issues a message that indicates the requirements for defining a suitable partition. The operator can either define the required partition or cancel the job.

When a suitable partition has been provided, the following message may be issued if the job is to be restarted at a checkpoint:

```
IEF390E DSO (outputclass,jobclass,devicetype)
        NEEDED TO RESTART jobname Pn
```

The message indicates that a DSO (direct system output) data set was open at the checkpoint. The data set was part of the specified system output class, and was assigned to a printer, card punch, or magnetic tape unit, as indicated by the message. The device originally used by the data set is no longer available because:

- The operator issued a STOP command to stop DSO processing on the device.
- The operator issued a MODIFY command to assign the device to a different system output class.
- The operator issued a DEFINE command to redefine partitions, and the job step is not being restarted in the original partition.

The operator can assign a device to the required system output class by issuing a START or MODIFY command. The START command starts DSO processing on a new device for the restart partition. The MODIFY command changes the system output class for a device that has already been started for the partition. When necessary, a STOP command can be issued for a DSO device started for another partition, and a START command issued for the same device in the restart partition. If a STOP command is issued for a DSO device being used by another job, the command will take effect when that job terminates.

Message IEF390E is issued once for each system output class that requires DSO device assignment. The job is then placed on the HOLD queue. The operator must release

the job for execution after assigning the required devices. If the required devices cannot be assigned, the operator should cancel the job.

### ***Deferred Restart Message Sequence***

To perform a deferred checkpoint/restart in MFT, the job to be restarted is resubmitted in an input job stream. Messages that contain checkpoint entry identifications were displayed on the console during the original execution of the job and may then be used by the programmer preparing the job for resubmission. When the resubmitted job is restarted, messages appear on the console in the following sequence:

1. When required main storage is not immediately available, a message indicating the main-storage requirements of the job
2. When a direct system output (DSO) device must be started, a message indicating DSO requirements
3. Normal mount messages
4. A successful restart message

To perform a deferred step restart in MFT, the job to be restarted is resubmitted. Normal mount messages are displayed.

### ***Operator Considerations During Deferred Checkpoint/Restart***

When a job is resubmitted to perform a deferred checkpoint/restart (the RESTART parameter is coded on the JOB statement with a checkid operand), the processing is essentially the same as during an automatic checkpoint/restart after the restart reader has reinterpreted the job.

If partitions have been redefined since the job was originally executed, there may be no partition suitable for restarting the job because:

- The required main-storage area may be split between two or more partitions, or may be included in the partition for a resident reader or writer.
- The partition containing the required main-storage area may include storage from hierarchy 0 when the original partition included only storage from hierarchy 1.

In either case, the system issues a message indicating the requirements for defining a suitable partition. The operator can either define the required partition or cancel the job.

The required main-storage area may be unavailable for the following reasons:

- A new IPL was performed and, because of different IPL options specified by the operator, the nucleus expanded into the required area.
- The job is resubmitted on an alternate system that does not have the required main-storage area.

If these conditions exist, a message is displayed indicating that main storage for the job step to be restarted is unavailable. The restart is terminated.

When main-storage requirements can be satisfied, message IEF390E may be issued to define direct system output (DSO) requirements. Operator response is the same as in the case of automatic checkpoint/restart.



## CHAPTER 6: STORAGE ESTIMATES

### Checkpoint/Restart Work Area

The user must ensure that enough main storage for a special work area is available prior to execution of the CHKPT macro instruction. The algorithm for computing the size of the work area is as follows:

$$\text{Work Area} = 1108 + T + 48(N-2) \text{ bytes}$$

*where:* T = the size of the TIOT at checkpoint time

N = the number of OPEN data sets at checkpoint time

#### Notes:

- For reference purposes, the size of a TIOT in bytes is:  $28+20A+4B$ , where A is the total number of data sets of the job step (including the JOBLIB if present) and B is the sum of any devices exceeding one allocated to each data set.
- N must: (1) have a value of at least 2 and (2) include the checkpoint data set whether it is open or not.
- If checkpoint/restart opens the checkpoint data set, the user must provide space for the IOB. With MFT, the user must also provide space for the DEB and ECB.
- For MFT, the user must provide 344 bytes.
- If a user's direct-access output data set requires a new extent after a checkpoint has been taken, and then an automatic restart is attempted, the restart function requires an additional 384 bytes of main storage. This main storage is acquired from the problem program partition in MFT and from subpool 253 in MVT.
- Additional storage is required for parallel repositioning of more than four data sets on magnetic tape. The control program requests all available storage up to  $128(P-4)$  bytes, where P is the number of tape data sets that were open at the checkpoint. The number of tape data sets repositioned in parallel is  $Q + 4$ , where Q is the number of contiguous 128-byte areas obtained by the control program.

### Checkpoint Data Set Storage Requirements

The checkpoint data set may be on any direct-access device or any magnetic tape device that is supported by BSAM or BPAM. The records forming a checkpoint entry are written in undefined format (format U) in the physical order shown in the table that follows. As the table shows, each checkpoint entry contains four types of records. Some types occur more than once in a single checkpoint entry. For each type of record, the table lists the size of one record and the number of records of that type that will appear in a checkpoint entry. From this table, the space required for each checkpoint entry can be determined, and when multiplied by the expected number of entries, the space required for the checkpoint data set can be determined.

Record Type	Record Size	Number of Records
CHR (Checkpoint Header Record)	400 bytes	1
DSDR (Date Set Descriptor Record)	400 bytes	(N/2)*
CIR (Core Image Record)	B bytes	A/B
SUR (Supervisor Record)	200 bytes	1 in MFT C/70 in MVT

\*Add one record for the first generation data group (GDG) data set and a second record for each additional 4 GDG data sets. Add one record for each data set requiring 6 to 20 volumes and a record for each additional 15 volumes.

*where:*

- N = total number of data sets in the job step
- A = problem program main-storage size defined as:
  - (1) The size of the partition in MFT
  - (2) The size of the region in MVT
- B = blocksize of the checkpoint data set either as specified by user ( $\geq 600$  bytes) or as assumed by the system in the absence of user specification (32,760 bytes for magnetic tape and track capacity for direct access)
- C = number of bytes of system queue space (SQS) used for the problem program

## Resident Access Methods

The checkpoint/restart facility processes the checkpoint data set using either BSAM or BPAM. The access method modules required to process the checkpoint data set must be resident in main storage.

At system generation, access method modules are made resident by the SUPRVSOR macro instruction. For MFT, this macro instruction must specify RESIDENT=ACSMETH; for MVT, it must specify RESIDENT=RENTCODE. As a result, certain access method modules are loaded automatically at IPL.

The modules loaded automatically are those listed in SYS1.PARMLIB member IEAIGG00. These modules are selected by the installation, although a standard list is suggested by IBM. The standard list includes the modules required to process a checkpoint data set, except those required for chained scheduling or track overflow.

In defining the list IEAIGG00, the installation can include the modules required for chained scheduling and track overflow. However, it can omit other modules that are required to process the checkpoint data set. Any module that is omitted is not loaded automatically at IPL.

When processing a checkpoint data set requires modules not listed in IEAIGG00, the installation must define an alternate list that includes the required modules. This list must be a member of SYS1.PARMLIB, and must be named IEAIGGxx, where xx represents any two letters or digits.

When an alternate list is defined, OPTIONS=COMM must be specified in the SUPRVSOR macro instruction at system generation. This operand causes the following message to be printed during IPL:

```
IEA101A SPECIFY SYSTEM PARAMETERS
```

For MFT, the operator must be instructed to reply RAM=xx, where xx represents the last two characters in the name of the alternate list. If the operator replies correctly, the modules listed in IEAIGGxx are loaded and remain resident until the next IPL. If the operator does not reply RAM=xx, the modules listed in IEAIGG00 are loaded. Note that only one of the lists (IEAIGG00 or IEAIGGxx) is used during each IPL.

For MVT, the operator can reply RAM=aa[bb,cc,dd], where the aa, bb, cc, and dd parameters are appended to IEAIGG to form the name of SYS1.PARMLIB members. The members contain lists of modules to be loaded in addition to the standard required modules. From one to four members can be specified. If specified during system generation and not modified by the operator's reply, the IEAIGG00 list is used.

**Modules Required for Checkpoint Restart:** The following modules must be resident:

Module	Approximate Size	Required by
IGG019BA	400	All checkpoint data sets
IGG019BB	300	
IGG019CC	480	
IGG019BC	240	Checkpoint data sets on direct-access devices
IGG019CD	630	
IGG019CH	130	
IGG019CU*	1560	Chained scheduling
IGG019CW*	550	
IGG019CV*	790	Chained scheduling with direct-access devices
IGG019CZ*	220	
IGG019C1*	350	Track overflow
IGG019C2*	1050	
IGG019C3*	350	

*If a checkpoint data set is to reside on an RPS (2305 or 3330) device, the following additional modules must be resident:*

IGG019C0	250	Channel end (Format U)
IGG019FN*	120	Start I/O appendage—RPS
IGG019C4*	300	End-of-extent appendage for search direct
IGG019FP*	490	Channel end for search direct
IGG019EK*	570	RPS start I/O, abnormal end, and channel-end appendage

\*These modules are not included in the IBM standard list.

**Defining a Resident Module List:** A list of resident modules can be created or modified by means of the IEBUPDTE utility program. The procedure is described in the *OS MVT Guide* or the *OS MFT Guide*.

## Resident Checkpoint/Restart Module for MFT

To improve system performance during tape data set repositioning at restart, the following module should be resident:

<b>Module</b>	<b>Approximate Size</b>	<b>Description</b>
IGC0S05B	940	Repositions tape data sets at restart

## CHAPTER 7: MISCELLANEOUS INFORMATION

### MVT Track Stacking

The checkpoint/restart facility can be used with the MVT track stacking facility.

### Job and Job Step Accounting and Checkpoint/Restart

In MVT, the system accumulates CPU time used for each job step and job. An installation can provide an accounting routine that will be given control at step initiation, step termination, and job termination for the purpose of accessing these time values. Accounting routines are discussed in detail in the *OS MVT Guide* or the *OS MFT Guide*. The relationship between the checkpoint/restart facility and the step time and job time values available to the accounting routine are as follows:

- At termination (either normal or abnormal) of an original execution, the step time and job time accumulated are available to the accounting routine.
- If a job is to be restarted at a checkpoint, the system executes a special step, named IEFDSDRP, before the restart step. The accounting routine is not given control during initiation or termination of this step.
- At initiation of the restart step during an automatic restart, step time and job time accumulated for the original execution are again available to the accounting routine.
- At initiation of the restart step during a deferred restart, step time and job time are zero.
- At termination of a restart step and at all subsequent times when the accounting routine is given control during the restart execution, the step and job times reflect only the time used during the restart execution. The time used by the IEFDSDRP step is not reflected.

To illustrate these points, assume that, in an original execution, Step A uses 2 minutes of CPU time and Step B uses 3 minutes of CPU time and abnormally terminates. At step termination the step time is 3 and the job time is 5. If automatic restart is performed for Step B, a step time of 3 and a job time of 5 are again available to the accounting routine at the reinitiation of Step B. If Step B then uses 4 minutes of CPU time and terminates, a step time of 4 and a job time of 4 are available to the accounting routine at step termination.

Note that the two values available at the time the restart step is initiated are provided for information purposes only. They are not reflected in the step and job running times presented at termination time of the restarted job. Thus the user need not be charged twice for the time accumulated up to the ABEND.

Another point to be considered in a user's accounting routine is the effect of a restart on the step sequence number available to the accounting routine. The following list

indicates the sequence number presented to the accounting routine under the various restart conditions:

Condition	Step Sequence Value for Step n
Original Execution	n
Automatic Step Restart	n
Automatic Checkpoint/Restart	n+1
Deferred Step Restart	1
Deferred Checkpoint/Restart	2

Whenever an automatic restart is performed the step sequence value accurately reflects the position of the step in the job. In the case of an automatic checkpoint/restart, the IEFSDSRP step has been executed before the restarting step. This accounts for the n+1 value.

In the case of a deferred restart, the restarting step is either the first step of the restart job or, in the case of a deferred restart from a checkpoint, it is the second (having been preceded by IEFSDSRP).

## MVT Job Step Time Limit

If MVT is used, the EXEC statement TIME parameter can be used to specify a limit on the CPU time to be used by the related step. With any kind of restart, the entire value of the limit specified for the job step applies to the restart step. In the case of a deferred restart, the programmer may specify a limit different from the limit he specified originally.

If the CPU time used by a step exceeds the specified limit while a checkpoint entry is being written, the entry is invalid and abnormal termination occurs. A preceding checkpoint entry can be used to perform a deferred restart. (If it is, and if sufficient checkpoints are taken during the restart execution, the invalid checkpoint entry will be overwritten by a valid entry.)

## Completion of Step or Job Termination at System Restart

If a step or a job is terminating when system failure occurs, the termination will be completed during the system restart that the operator may perform after the failure. This will occur whether or not the step or job uses the checkpoint/restart facility. If other than the last step of a job is terminating when the failure occurs, the termination will be completed during the system restart and the next step of the job will subsequently be initiated. If the last step of a job is terminating, or if the job is terminating, all necessary terminations will be completed. If a job requests an automatic restart and then abnormally terminates, and if system failure occurs before the restart processing is complete, the processing will be completed during the system restart.

## Reader Procedure Changes

If a deferred restart at a checkpoint is to be allowed, the installation may need to change the IBM-supplied cataloged procedure used to start a reader. This may be necessary because:

- The reader procedure specifies UNIT=SYSDA for SYSIN data sets and SYSDA as a default value for SYSOUT data sets.
- SYSDA may be defined by the SYSGEN UNITNAME macro instruction to denote a set of devices including multiple device types.
- The SYSIN and SYSOUT data of a job experiencing a deferred restart are written into new data sets. The data sets are allocated devices and volumes as directed by the reader procedure.
- For each SYSIN or SYSOUT data set open at the checkpoint to be used, the device type during the restart execution must be the same as during the original execution.

Therefore, if SYSDA denotes multiple device types, the reader procedure should be changed to refer to an installation-chosen unit name that denotes only one device type.

## COBOL RERUN Clause

The COBOL RERUN clause may be used to provide the COBOL user with linkage to the checkpoint/restart facility. Cautions and restrictions on the use of the checkpoint/restart facility also apply to the use of the RERUN clause.

Further information on the RERUN clause may be found in *OS COBOL Language* or *OS USA Standard COBOL*.

## Checkpoint/Restart and the Sort/Merge Program

When performing a sort with the sort/merge program, the user can, by including the CKPT parameter in his sort control statements, cause checkpoint entries to be written and an automatic checkpoint/restart to be requested.

The job control language can be used to request automatic or deferred step restarts or a deferred restart at a checkpoint. See *OS Sort/Merge* for additional information.

## PL/I Checkpoint/Restart Capability

The PL/I user can invoke automatic and deferred step restart and can also take checkpoints and invoke automatic and deferred checkpoint/restarts. To cause a checkpoint entry to be written and request an automatic checkpoint/restart, the user codes in his program:

```
CALL IHECKPT
```

Each checkpoint entry in the checkpoint data set is identified by a system-generated checkid. A system message on the console, which includes the checkid, notifies the operator that a checkpoint entry has been written.

The organization of the checkpoint data set is always physical sequential, and the data set may be written on magnetic tape or a direct-access volume. Partitioned organization cannot be used.

A DD statement must be present in the job stream to define the checkpoint data set. The DISP parameter in this DD statement is used to specify whether single or multiple checkpoint entries are to be written. DISP=(NEW,KEEP) specifies a single checkpoint entry, while DISP=(MOD,KEEP) specifies multiple checkpoint entries.

For additional information, see *OS PL/I (F) Programmer's Guide* or the following program product publications: *OS PL/I Checkout Compiler General Information* or *OS PL/I Optimizing Compiler General Information*.

## **TCAM Data Set Considerations**

A successful restart of a telecommunications access method (TCAM) data set depends on the following conditions:

- The message control program (MCP) region must be active and have enough main storage to build the required control blocks.
- The QNAME= parameter in the DD statement of the checkpoint job must be available in the Terminal Table of the MCP region.



## APPENDIX A: COMPLETION CODES

### Return Codes Associated with the CHKPT Macro Instruction

Code (Hexadecimal)	Meaning
00	<i>Successful completion.</i> Code 00 is also returned if the RD parameter was coded as RD=NC or RD=RNC to totally suppress the function of CHKPT.
04	<i>Restart has occurred</i> at the checkpoint taken by the CHKPT macro instruction during the original execution of the job. A request for another restart of the same checkpoint is normally in effect. If a deferred restart was performed and RD=NC, NR, or RNC was specified in the resubmitted deck, a request for another restart is not in effect.
08	<i>Unsuccessful completion.</i> A checkpoint entry was not written, and a restart from this checkpoint was not requested. A request for an automatic restart from a previous checkpoint remains in effect.  One of the following conditions exists: <ul style="list-style-type: none"><li>• The parameters passed by the CHKPT macro instruction are invalid.</li><li>• The CHKPT macro instruction was executed in an exit routine other than the end-of-volume exit routine.</li><li>• A STIMER macro instruction has been issued, and the time interval has not been completed.</li><li>• A WTOR macro instruction has been issued, and the reply has not been received.</li><li>• The checkpoint data set is on a direct-access volume and is full. Secondary space allocation was requested and performed. (Secondary space allocation is performed for a checkpoint data set, but the allocated space is not used. However, had secondary allocation not been requested, the job step would have been abnormally terminated.)</li><li>• In a system with MVT, the job step (1) comprises more than one task or (2) has been allocated storage external to its region through the rollout/rollin option.</li><li>• The CHKPT macro instruction was issued for a data set on a graphics device.</li></ul>
0C	<i>Unsuccessful completion.</i> An uncorrectable error occurred in writing the checkpoint entry or in completing queued access method input/output operations that were begun before the CHKPT macro instruction was issued. A partial, invalid checkpoint entry may have been written. If the entry has a programmer-specified checkid, and the checkpoint data set is sequential, a different checkid should be

specified the next time CHKPT is executed. If the data set is partitioned, a different checkid need not be specified. This code is also returned if the checkpoint routine tries to open the checkpoint data set and the DD statement for the data set is missing.

- 10      *Successful completion with possible error condition.* The task has control, by means of an explicit or implied use of the ENQ macro instruction, of a serially reusable resource; if the task terminates abnormally, it will not have control of the resource when the job step is restarted. The user's program must, therefore, restore the enqueues.

Additional information regarding explicit and implicit use of the ENQ macro instruction may be found in the section "Cautions in Taking a Checkpoint."

When one of the errors indicated by code 08, 0C, or 10 occurs, the system prints an error message on the operator's console. The message indicating code 08 or 0C contains a code that further identifies the error. The operator should report the message contents to the programmer.

## **Completion Codes Issued by Checkpoint/Restart**

The code 13F indicates that an error occurred during performance of a checkpoint/restart. If a SYSABEND card is included in the job, a dump is produced, and the contents of the system control blocks, as shown in the dump, are unpredictable.

The code 2F3 indicates that a job was executing normally when system failure occurred.

## APPENDIX B: END-OF-VOLUME EXIT ROUTINE (TAKING A CHECKPOINT AT END-OF-VOLUME)

The user can specify, in the related data control block exit list, the address of a routine that is to be given control when end-of-volume is reached in processing a physical sequential data set (BSAM or QSAM). (See *OS Data Management Services Guide* for information about forming an exit list.) The routine is entered after a new volume has been mounted and all necessary label processing has been completed. If the volume is a reel of magnetic tape, the tape is positioned after the tape mark that precedes the beginning of the data. The end-of-volume exit routine may take a checkpoint by issuing the CHKPT macro instruction. If the job step terminates abnormally, it can be restarted from this checkpoint. When the job step is restarted, the volume is mounted and positioned as upon entry to the routine.

The end-of-volume exit routine returns control in the same manner as any other data control block exit routine. Note that restart becomes impossible if changes are subsequently made to the system SVC library (SYS1.SVCLIB). (When the step is restarted, the TTRs of end-of-volume modules must be the same as when the checkpoint was taken.)

On entry to the user's end-of-volume exit routine, the contents of the registers are:

Registers	Contents
0	Zero
1	Address of data control block
2-13	Contents before execution of the input/output macro instruction
14	Return address (must be preserved by the exit routine)
15	Address of the end-of-volume exit routine

### Notes:

1. The contents of registers 0 through 13 and 15 need not be preserved by the exit routine.
2. The exit routine must not use the save area pointed to by register 13 upon entry. If the exit routine calls another routine or executes system macro instructions, it must provide its own save area.
3. The exit routine is not provided for EXCP users, since they must explicitly execute the EOVS macro instruction.



# INDEX

Indexes to Systems Reference Library publications are consolidated in *OS Master Index to Reference Manuals*, GC28–6644. For additional information about any subject listed below, refer to other publications listed for the same subject in the *Master Index*.

/\* statement 36

## A

access method modules 36,48–49  
access methods  
  BDAM 9,19  
  BISAM 9  
  BPAM 48  
  BSAM 19,48  
  for checkpoint/restart 48  
  QISAM ( *see* QISAM)  
  QSAM 19,22  
  resident 48–49  
  TCAM 54  
accounting routine 51–52  
advanced checkpoint/restart  
  ( *see* checkpoint/restart)  
ATTACH macro instruction 3  
automatic checkpoint/restart  
  data set disposition 29–30  
  described 1  
  how initiated 30–31  
  how input work queue merged 30  
  JCL requirements 28–29  
  how job deck reinterpreted 30  
  messages issued 39,43  
  how to request 28  
  resource variations allowed 29  
automatic step restart  
  after checkpoint/restart 31  
  when checkpoint/restart requested 28  
  data set disposition 29–30  
  described 1  
  how initiated 30  
  how input work queue merged 30  
  JCL requirements 28–29  
  how job deck reinterpreted 30  
  messages issued 39–40,43–44  
  how MOD data sets handled 31  
  how to request 27–28  
  resource variations allowed 29

## B

basic direct access method 9,19  
basic indexed sequential access method 9  
basic partitioned access method 48  
basic sequential access method 19,48  
BDAM (basic direct access method) 9,19  
bias table 22

BISAM (basic indexed sequential access method) 9  
block count field 21  
BPAM 48  
BSAM (basic sequential access method) 19,48  
buffer 18,19

## C

CANCEL command  
  at automatic restart 29,40–41  
  in MFT 44  
  how to use 15  
CANCEL operand 6–7  
card reader 17,19  
cataloging generation data set 22  
cataloged procedure 25–26,53  
central processing unit time 51–52  
CHECK macro instruction 19  
checkid  
  address operand in CHKPT macro 6  
  defined 1  
  duplicate 14  
  length operand in CHKPT macro 6  
  in message at restart 40,44  
  in partitioned data set 6,14  
  primary identification 15  
  S operand in CHKPT macro 6  
  secondary identification 15  
  in sequential data set 6,14  
  how to specify 6  
  system generated 9  
checkpoint  
  how to establish 5–15  
  when to take 18  
checkpoint data set  
  closing the 12  
  contents of 12  
  DCB for 9–10  
  DD statement for 10–11  
  defined 1  
  I/O errors associated with 21–22  
  labels for 5,10  
  modules that process 48–49  
  opening the 11–13  
  in PL/I 53–54  
  resides on 5  
  storage for 47–48  
checkpoint entry  
  at checkpoint/restart 31  
  described 1  
  record format of 47–48  
  storage for 47–48  
  how written 11–12

- checkpoint identification ( *see* checkid)
- checkpoint header record 48
- checkpoint/restart
  - access methods used 48
  - completion codes issued by 3,56
  - components 1-3
  - defined 1
  - modules required for 49
  - in PL/I 53
  - system generation requirements 2-3
  - work area 47
- checkpoint routine 12,17
- CHKPT macro instruction
  - how to code 5-7
  - described 1
  - at end of volume 57
  - return codes associated with 55-56
  - with specific access methods 9
  - suppressing action of 25-26
- CHR (checkpoint header record) 48
- CIR (core image record) 48
- CKPT parameter 53
- CKPTREST macro instruction 2-3
- closing checkpoint data set 12
- COBOL RERUN clause 53
- codes
  - completion 3,56
  - return 55-56
- completion codes 3,56
- COND parameter 33,36
- control blocks
  - DCB 5,9-10,57
  - DSCB 17,35
  - TCB 7
- core image record 48
- core storage ( *see* main storage)
- CPU (central processing unit) time 51-52

## D

- DASD (direct-access storage device) 9,17
- data control block
  - address operand in CHKPT macro 5
  - exit list 57
  - parameters for checkpoint data set 9-10
- data definition statement ( *see* DD statements)
- data set control block (DSCB) 17,35
- data set descriptor record 48
- data sets
  - checkpoint ( *see* checkpoint data set)
  - disposition at automatic restart 29-30
  - disposition at deferred checkpoint/restart 34-35
  - disposition at deferred step restart 32
  - direct access 14
  - extents 35
  - GDG 48
  - generation 22-23,33
  - ISAM 18
  - MOD 31,17
  - partitioned 17-18,20
  - preallocated 23
  - repositioning at checkpoint/restart 31

- repositioning at deferred step restart 32
- repositioning of unit record 17,19
- SYSABEND 24
- SYSIN 19,23,53
- SYSOUT ( *see* SYSOUT data sets)
- SYSUDUMP 24
- tape 17-18,21,31
- TCAM 54
- user 17-24
- work 20-21
- DCB (data control block)
  - address operand in CHKPT macro 5
  - exit list 57
  - parameters for checkpoint data set 9-10
- DD statement
  - for automatic checkpoint/restart 28
  - for checkpoint data set 10-11
  - for deferred checkpoint/restart 35-36
  - for deferred step restart 32-33
  - for generation data set 22
  - QNAME operand 54
  - SUBALLOC parameter 33,36
  - VOLUME parameter 33,36
- deferred checkpoint/restart
  - data set disposition at 34-35
  - described 1
  - JCL requirements 34-36
  - messages issued 42,45-46
  - how to request 33-34
  - resource variations allowed in 36-37
  - how system works at 37
- deferred step restart
  - data set disposition at 32
  - described 1
  - JCL requirements 32-33
  - messages issued 42,45-46
  - how to request 31-32
  - resource variations allowed in 33
- DEFINE command 45
- delimiter (/\*) statement 36
- device, changing at restart
  - automatic restart 29-30
  - deferred checkpoint/restart 36
  - deferred step restart 33
  - in MFT 44
  - in MVT 40
- devices
  - for data sets at checkpoint/restart 18
  - direct-access storage 9,17
- direct-access storage devices 9,17
- direct-access volume 31
- direct system output ( *see* DSO)
- directory, preserving contents of 20
- disk operating system 18
- DISPLAY A command 41-42
- disposition
  - at automatic restart 29-30
  - of checkpoint data set 11-12
  - at deferred checkpoint/restart 34-35
  - at deferred step restart 32
- DOS (disk operating system) 18
- DSCB (data set control block) 17,35
- DSDR (data set descriptor record) 48

DSO (direct system output)  
  with SYSABEND data sets 24  
  with SYSOUT data sets 23–24  
  operator considerations 39–43,45–46  
DUMMY parameter 36

## E

ELIGBLE operand 2  
end of volume 11–12,18  
end-of-volume exit routine ( *see* exit routine)  
ENQ macro instruction 8  
errors, input/output 21–22  
exceptional conditions 7  
EXCP macro 17  
EXEC statement  
  for automatic restart 28–29  
  COND parameter 33,36  
  for deferred checkpoint/restart 35  
  for deferred step restart 34  
  PGM parameter 33,36  
  RD parameter 25–26,28  
exit routine, end-of-volume  
  checkpoint in 8,36  
  described 2  
  register contents on entry to 57  
  restarting at checkpoint in 35  
extents 35  
EXTRACT macro instruction 7

## F

FCB (forms control buffer) 7  
forms control buffer 7

## G

GDG (generation data group) data set 48  
generation data group data set 48  
generation data sets 22–23,33  
generation number, relative 22  
generation, system 2–3

## H

hierarchy 30,36,45–46  
HOLD command  
  at automatic restart 30  
  in MFT 44–45  
  in MVT 40–42

## I

IEFDSDRP 30,51  
IEFREINT  
  at automatic restart 29  
  described 39  
  in MFT 45  
  in MVT 41  
IEHREST 3  
IGCOSO5B 18  
IHECKPT 53  
initial program load 41,43–44,46  
initiators 41,45  
input/output errors 21–22  
input work queues 30,37  
IPL (initial program load) 41,43–44,46  
ISAM data sets 18

## J

JCL  
  for automatic restart 28–29  
  for deferred checkpoint/restart 34–36  
  for deferred step restart 32–33  
job deck reinterpretation 30  
JOB statement  
  for automatic restart 29  
  for deferred checkpoint/restart 35  
  for deferred step restart 31  
  RD parameter 25–26,28  
  RESTART parameter 26  
job step  
  message at abnormal termination 39–40,43–44  
  termination 52  
  time 51–52

## L

labels  
  for checkpoint data set 5,10  
  nonstandard tape 21  
  standard volume 21  
link library 36  
link pack area 43  
link pack area modules 37  
list  
  DCB exit 57  
  standard module 48–49  
LOAD macro instruction 37

## M

macro instructions  
  ATTACH 8  
  CHECK  
  CHKPT ( *see* CHKPT macro instruction)  
  CKPTREST 2–3  
  ENQ 8  
  EXCP 17  
  EXTRACT 7  
  LOAD 37  
  POINT 14  
  RESERVE 9  
  SETPRT 7–8  
  STIMER 8  
  STOW 12,18,20  
  SUPRVSOR 3,48  
  UNITNAME 53  
  WAIT 19  
  WTOR 8  
magnetic tape 17–18,21,31  
main storage  
  at automatic checkpoint/restart 30  
  at deferred checkpoint/restart 36–37  
  at restart in MFT 45–46  
  at restart in MVT 39,41–43  
master scheduler region 42  
MCP (message control program) region 54  
member of partitioned data set 18,20  
memory ( *see* main storage)  
message control program region 54  
messages  
  when checkpoint successful 15  
  error 56  
  at IPL 48  
  at restart 39–46  
MFT (multiprogramming with fixed number  
of tasks)  
  messages issued at restart 39–46  
  operator response at restart 44–46  
  partition size 30,36,48  
  tape data set repositioning 18  
MOD data sets 17,31  
MODIFY command 41,45  
modules  
  IGCOSO5B 18  
  that process checkpoint data set 48–49  
  resident access method 36,48–49  
  resident for MFT 50  
  standard list of 48–49  
MSGLEVEL 29  
MVT  
  job step time 52  
  messages issued at restart 39–43  
  operator response at restart 40–43  
  region size 48  
  track stacking 51  
multiprogramming ( *see* MFT or MVT)

## N

NO command 40,44  
nonstandard tape labels 21  
NOTELIG operand 2  
nucleus of alternate system at restart  
  in MFT 46  
  in MVT 43

## O

open routine 17–18  
opening checkpoint data set 11–12  
operator considerations 39–46  
OPTIONS parameter 48

## P

partition  
  at automatic restart 45  
  at deferred checkpoint/restart 46  
  in MFT 30,36,48  
partitioned data set 17–18,20  
PGM parameter 33,36  
PL/I programs 3,53–54  
POINT macro instruction 14  
preallocated data sets 23  
printer  
  repositioning at restart 17,19  
  1403 7  
  3211 7  
programs  
  sort/merge 53  
  message control 54  
punch 17,19

## Q

QISAM (queued indexed sequential access method)  
  input/output errors 22  
  restart at checkpoint 19  
  taking checkpoint with 9  
QNAME parameter 54  
QSAM (queued sequential access method) 19,22  
queue, input work 30,37  
queued indexed sequential access method  
( *see* QISAM)  
queued sequential access method 19,22



## R

RAM (resident access method) 36,48–49  
RD (restart definition) parameter  
  for automatic restart 28  
  how to code 25–26  
  described 2  
reader  
  for automatic restart 29  
  cataloged procedure for 53  
  restart 39,41–43,45–46  
records  
  for checkpoint entry 47–48  
  updated after checkpoint 17  
region  
  master scheduler 42  
  size of in MVT 48  
REGION parameter 42  
register contents 57  
relative generation number 22  
RELEASE command  
  at automatic restart 29  
  in MFT 44–45  
  in MVT 40–41  
repositioning  
  card reader 17,19  
  at checkpoint/restart 31  
  data sets 17–18  
  at deferred step restart 32  
  direct access data sets 14  
  MOD data sets 18,31  
  partitioned data sets 18  
  routine 28  
  SYSIN data sets 23  
  SYSOUT data sets 23–24  
  tape data sets 21  
  unit record data sets 17,19  
RERUN clause 53  
RESERVE macro instruction 9  
resident access method 36,48–49  
resource variations  
  automatic checkpoint/restart 29  
  automatic step restart 29  
  deferred checkpoint/restart 36–37  
  deferred step restart 33  
resources, serially reusable 8–9  
restart  
  automatic checkpoint ( *see* automatic  
  checkpoint/restart)  
  automatic step ( *see* automatic step restart)  
  deferred checkpoint ( *see* deferred  
  checkpoint/restart)  
  deferred step ( *see* deferred step restart)  
  at end of volume 18  
  how to ensure 12–13  
  of generation data sets 22  
  messages issued at 39–46  
  of MOD data set 31  
  in PL/I 53–54

  repositioning data sets at 17–18  
  of SYSIN data sets 23  
  of SYSOUT data sets 23–24  
restart definition parameter ( *see* RD parameter)  
RESTART parameter  
  how to code 26  
  for deferred checkpoint/restart 35  
  for deferred step restart 31–32  
  described 2  
restart reader 39,41–43,45–46  
return codes 55–56  
rollout/rollin 9  
routine  
  accounting 51–52  
  end-of-volume exit ( *see* exit routine)  
  open 17–18  
  for nonstandard tape labels 21  
  repositioning 28

## S

serially reusable resources 8–9  
SETPRT macro instruction 7–8  
sort/merge program 53  
SQS (system queue space) 41–42,48  
SRTEDMCT field 21  
standard completion codes 3,56  
standard list of modules 48–49  
standard volume label 21  
START command 41,45  
step time 51–52  
STIMER macro instruction 8  
STOP command 41,45  
storage ( *see* main storage)  
storage estimates  
  for checkpoint data set 47–48  
  for checkpoint/restart work area 47  
  for MFT checkpoint/restart module 50  
  for resident access methods 48–49  
STOW macro instruction 12,18,20  
SUBALLOC parameter 33,36  
supervisor record 48  
SUPRVSOR macro instruction 3,48  
SUR (supervisor record) 48  
SVC library 57  
SYSABEND data set 24  
SYSCHK DD statement  
  how to code 26–27  
  for deferred checkpoint/restart 35  
  described 2  
SYSGEN ( *see* system generation)  
SYSIN data sets  
  checkpoint positioning information 19  
  at restart 23,53  
SYSOUT data sets  
  checkpoint positioning information 19  
  default values for 29  
  at restart 23–24,53  
SYSOUT parameter 24  
system completion codes ( *see* completion codes)

- system failure when job or step terminating 52
- system generation
  - CKPTREST macro instruction 2-3
  - resident access method modules 48
  - UNITNAME macro instruction 53
- system message blocks 30
- system operations
  - at automatic restart 29-30
  - at deferred checkpoint/restart 37
- system queue space 41-42,48
- SYSUDUMP data set 24
- SYS1.PARMLIB 48-49
- SYS1.SVCLIB 36,57
- SYS1,SYSJOBQE 30

## T

- tables
  - bias 22
  - terminal 54
- tape data set
  - MOD 31
  - processing with EXCP 17
  - repositioning 17-18,21
- tape labels, nonstandard 21
- task control block 7
- TCAM (telecommunications access method) 54
- TCB (task control block) 7
- telecommunications access method 54
- terminal table 54
- termination of step or job
  - completed at system restart 52
  - messages issued 39-40,43-44
- trackstacking 51

## U

- UCS (universal character set) 7,18
- unit record devices, repositioning 17,19

- UNITNAME macro instruction 53
- universal character set 7,18
- UNLOAD command 40,44
- update in place 17,20
- user data set 17-24
- user repositioning routine 19

## V

- VARY command 40,44
- volume label, standard 21
- volume, changing at restart
  - at automatic restart 29
  - at deferred checkpoint 36
  - at deferred step 33
  - in MFT 44
  - in MVT 40
- VOLUME parameter 33,36

## W

- WAIT macro instruction 19
- work area, for checkpoint/restart 47
- work data sets 20-21
- WTOR macro instruction 8

## Y

- YES command 40-41,44-45

- 1403 printer 7
- 2361 core storage device 37
- 3211 printer 7



**READER'S COMMENT FORM**

OS Advanced Checkpoint/Restart

GC28-6708-4

Your comments about this publication will help us to produce better publications for your use. If you wish to comment, please use the space provided below, giving specific page and paragraph references.

Please do not use this form to ask technical questions about the system or equipment or to make requests for copies of publications. Instead, make such inquiries or requests to your IBM representative or to the IBM Branch Office serving your locality.

---

Reply requested

Yes

No

Name \_\_\_\_\_

Job Title \_\_\_\_\_

Address \_\_\_\_\_

\_\_\_\_\_ Zip \_\_\_\_\_

No postage necessary if mailed in the U S A

**YOUR COMMENTS, PLEASE . . .**

This publication is one of a series which serves as a reference source for systems analysts, programmers, and operators of IBM systems. Your answers to the questions on the back of this form, together with your comments, will help us produce better publications for your use. Each reply will be carefully reviewed by the persons responsible for writing and publishing this material. All comments and suggestions become the property of IBM.

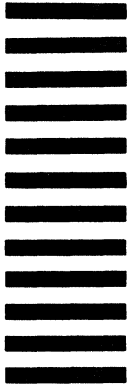
Please note: Requests for copies of publications and for assistance in utilizing your IBM system should be directed to your IBM representative or to the IBM sales office serving your locality.

fold

fold

**BUSINESS REPLY MAIL**  
NO POSTAGE STAMP NECESSARY IF MAILED IN U. S. A.

FIRST CLASS  
PERMIT NO. 2078  
SAN JOSE, CALIF.



POSTAGE WILL BE PAID BY . . .

IBM Corporation  
Monterey & Cottle Rds.  
San Jose, California  
95114

Attention: Programming Publications, Dept. D78

fold

fold



**International Business Machines Corporation**  
**Data Processing Division**  
**1133 Westchester Avenue, White Plains, New York 10604**  
**(U.S.A. only)**

**IBM World Trade Corporation**  
**821 United Nations Plaza, New York, New York 10017**  
**(International)**



**International Business Machines Corporation**  
**Data Processing Division**  
**1133 Westchester Avenue, White Plains, New York 10604**  
**(U.S.A. only)**

**IBM World Trade Corporation**  
**821 United Nations Plaza, New York, New York 10017**  
**(International)**