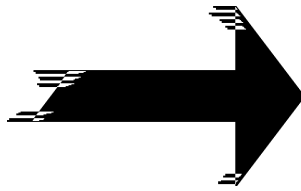
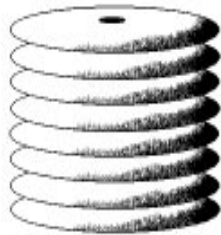


Xdr

User's Guide



Sequential Software, Inc.

Release 1.22

January, 1997

© Copyright Sequential Software, Inc. 1992-1997. All rights reserved.

10 Wilsey Square, Suite 9, Ridgewood, New Jersey 07450

Phone (201) 652-3300. Fax (201) 652-5596 Email: info@seqsoft.com www.seqsoft.com

Rev. 11/1997

Contents

Introduction	5
Performance	5
File Level Restore & Device Independence	5
Standard Label Tapes	5
Exploiting ECKD Storage Controls	5
Bypassing Input I/O Errors	6
Varying I/O Amount and Asynchronous Operations	6
Dumping & Restoring Using CMS Files Instead of Tape	6
Installation	7
Installing from tape (3480/3490 cartridge)	7
Installing from diskette	7
Overview of Main Operations	8
Dumping	8
Restoring	8
Copying	9
Disaster-Recovery	9
Command Line Syntax	10
I/O Definition Statements	12
Input	12
Output	16
SYsprint {cuu CONS}	20
NOPRINT	20
Operation Statements	21
DUMP	21
RESTORE	21
COPY	22
CMSREST	22
Stand-Alone Restores of Standard Labeled Tapes	24
Working with VM Tape Managers	26
XDRC	27
XDRINST	29
XDRTIME	30
XDRTP	31
XDRTP Messages	31
ZZAP	32
Examples	33
Messages	34
Summary of Unsupported DDR Functions	43
Index	44

Introduction

XDR is designed to speed up the most commonly used functions of the DDR utility supplied by IBM with VM. The basic functions are to COPY disks, DUMP data to tape, and RESTORE data back to disk. XDR is syntax compatible with the DDR utility; it supports the same basic control statements which are either entered interactively or provided through a CMS file. Tapes created by XDR can subsequently be used by DDR, and vice versa. The special header records found in a tape file are compatible between the two utilities. Tape volume handling (positioning, end-of-volume processing) is also compatible. Data written by XDR to disk will be the same as if DDR had operated.

XDR operates under all current releases of VM and supports most disk and tape devices. Older devices, such as 2311, 2314, and 3340's, are not supported. XDR will exploit the 31-bit architecture features of VM/ESA if used in such a CMS machine. Both its code and I/O buffers can reside above the magical 16M address line. XDR in general requires greater virtual storage than DDR since its I/O buffers are larger. The effects of greater storage use should be minimal since XDR will rarely be run simultaneously with other applications that are heavy users of virtual storage.

There are several new features available beyond what regular DDR supplies. Other new capabilities suggested by users can be added in the future.

Performance

XDR operates from 2 to 4 times faster than DDR. This is achieved by using sophisticated I/O techniques to read and write data. DDR essentially operates on one track's worth of data at-a-time, while XDR can operate on up to a cylinder's worth. Disk latency times are reduced. Improved synchronization between disk and tape devices also results from this. In addition, XDR will attempt to perform simultaneous reading and writing of data. When COMPACT is specified, XDR attempts concurrent reading, writing, and CPU processing.

File Level Restore & Device Independence

A handy feature is the ability to restore one or many CMS files from a backup tape file (created by either XDR or DDR). Backup/restore systems that are built around the DDR utility should benefit the most from this. To restore a single CMS file with DDR, the entire minidisk on which it resides must be RESTORED to a disk of equivalent size; the file can then be copied out to another disk. Additionally, the output device type must match what the original minidisk type was. These two items are no longer required when using the file level restore ability of XDR. In fact, XDR now gives a level of device independence for CMS disks backed up with DDR. A 3380 minidisk backed up to tape, for example, can be restored to a 3390 in the future. The only requirement is that the disk block size of the output disk must match the original block size. One or many files can be restored out of a backup tape. Several options are available to control items such as the timestamp of the file, whether previously existing files should be replaced, and whether to display a list of files being restored.

Standard Label Tapes

Standard IBM tape labels are supported through the SLOS option on an INPUT or OUTPUT statement. XDR can verify that the correct tape has been mounted for a particular DUMP or RESTORE operation. A tape containing an unexpected volume id will be unloaded, with a message sent to the user requesting that the proper tape be mounted. Label information is provided through the FILEDEF and LABELDEF commands. Multi-volume tape files are fully supported when the SLOS option is used.

Exploiting ECKD Storage Controls

XDR automatically detects the presence of modern (eg., 3990, 9340, RAMAC) storage controls and adapts itself

accordingly. DDR's usual method of reading disk tracks is with a CCW called "read-multiple count,key,data." A more efficient, and reliable CCW is available called "read-track", which XDR uses. Reductions in Vtime/Ttime will be realized when XDR can use "read-track", due to somewhat arcane reasons. Additionally, XDR's use of "read-track" requires a channel program to be built that should work with IBM's Escon setup, hopefully without any loss in performance when disk devices are located buildings away.

Bypassing Input I/O Errors

A permanent I/O error can optionally be bypassed if the EMSG or ESKIP options are specified on an INPUT statement. The former option results in error messages describing the problem, followed by a prompt asking the user whether or not to continue. The latter option simply issues the error messages, and continues the particular operation in progress. The troubled disk address is shown, alerting the user that data contained on that track is unavailable.

Varying I/O Amount and Asynchronous Operations

Command line options are available that increase or decrease the number of tracks processed per I/O request, and that force or disable the use of the SIO or SSCH instructions. XDR can process up to the equivalent of 1 cylinder's worth of data per I/O request. It may be useful in some circumstances to limit XDR to something less, although we have rarely seen any reason to with other program products. Today's expensive hardware is quite capable of handling large I/O requests. XDR will try to detect situations when asynchronous operations are possible, such as dumping from disk to tape where different physical paths to each device exist. There may be times when XDR cannot correctly determine the physical path, so options are available to force either synchronous or asynchronous I/O.

Dumping & Restoring Using CMS Files Instead of Tape

The DUMP and RESTORE functions can operate with CMS files in place of tape. This can facilitate the testing of some application since a physical tape drive need not be attached to the virtual machine.

Installation

Installing from tape (3480/3490 cartridge)

To install XDR issue **“TAPE LOAD * * A”** with the XDR installation tape mounted and ready on virtual device 181 in a CMS machine. A second TAPE LOAD brings down the optional help files.

Installing from the self-contained **XDRLD.EXC** file

The file called **“XDRLD.EXC”** containing only regular ASCII characters. It may be provided to you on diskette, as an email attachment, or on magnetic tape. Upload this file to CMS as **“XDRLD EXEC”** using a transfer method that converts from ASCII to EBCDIC, if necessary. Then simply run **“XDRLD”** which prompts for the disk mode letter to install the product on. A prompt is also issue if you wish to install the help files.

Either method loads several files to the target CMS disk. The most important file is

XDR MODULE

XDR MODULE is the essential routine for replacing the DUMP, RESTORE, and COPY functions of DDR. It is self-contained and immediately ready to execute.

There are two different ways to use XDR in place of DDR. The first involves substituting the command name **“XDR”** for **“DDR”** in any EXEC or program that calls DDR. Since DDR is typically used in only one or a few CMS machines, this substitution process is easy to accomplish.

The second method of using XDR instead of DDR uses the NUCXLOAD facility to load a small **“front-end”** routine which gets called whenever DDR is invoked. This routine is called **“XDRFE MODULE.”** Each user who wishes to replace DDR with XDR should enter the following command. It can be placed in the system profile exec. Note that certain DDR options are not available with XDR.

NUCXLOAD DDR XDRFE (SYSTEM

Several other files are loaded during the install procedure. More information about them is contained elsewhere in this Guide. They are,

XDRC - compares two disks byte for byte.

XDRINST - an exec to set default values such as IOLIM and TAPF.

XDRTIME - an exec to compare XDR with DDR.

XDRTP - view tape blocks, a debugging tool.

ZZAP - a maintenance tool.

Overview of Main Operations

Dumping

XDR can dump the physical contents of a disk device to a tape file. The number of cylinders processed is specified on the DUMP statement and subsequent extent statements if necessary. At the start and end of each tape file, including multi-volume files, are special volume header and trailer records. These record information on the physical characteristics of the disk device and a timestamp of when the dump was taken. Logically, each track is operated on as a single entity. A 32-byte track header record is prefixed to the track data, containing flag information, the original track address, tape recording density, and the number of bytes in the track. All DUMP operations create tape files in so-called "FTR" (full-track-read) format. The DUMP operation proceeds until each track in all extents have been written to tape.

An INPUT and OUTPUT statement defining a disk and tape, respectively, must be entered prior to the DUMP statement. Output tape positioning is performed with the SKIP and REWIND/LEAVE/UNLOAD options. Multi-volume tape files are created when there is not sufficient space on a tape to contain all the data that is being dumped. A slight reduction in VTIME is achieved when XDR can determine that the input disk is in CMS format.

Restoring

There are two levels of restores possible with XDR. The first is equivalent to DDR's restore function, namely rebuilding a disk from a physical image of it stored in a tape file. This is the inverse operation of DUMP; each disk track is restored from the tape. Sub-extents of the original disk can be selected for the restore operation, if desired. Sub-extent processing is useful when an entire disk pack has been dumped as a unit, while only a minidisk within it needs to be restored.

When restoring a physical image of a disk, the output device type should be the same as original device type dumped. It might be possible to restore to an unlike type, but strange errors are likely in whatever application will process the disk. XDR also attempts to check that the output disk is equal to or larger in size than the original disk; this takes place early in XDR's restore function.

The second restore possibility with XDR is called *file level restore*, and is only available with CMS minidisks. File level restore operates with exactly the same tape file as is produced by the dump operation. Tapes created by either XDR or DDR can be used. A control statement, CMSREST, unique to XDR specifies what file or files to restore from the tape. The entire original CMS minidisk must be dumped to tape to attempt a file level restore. An output disk must of course be available to restore to, and it must be ACCESSED by the CMS file system. The OUTPUT statement or the CMSREST statement defines the output disk. Previously existing files on the output disk can optionally be pre-erased to allow the file level restore to complete.

XDR will locate the file(s) to be restored through the directory file that is part of the dump data. The restored files should match byte for byte with their original incarnations. Multi-volume tape files may result in a tape being unloaded and then being reused; console messages will indicate what logical tape number to mount at a particular address. This non-sequential access is a result of having to make more than one pass over the tape data to first retrieve the directory, and then to restore the selected files.

Some tapes created by DDR may not be immediately usable with the file level restore option. If DDR encountered a recoverable data check while dumping a disk, the corresponding track image on tape will not be in so-called FTR format. It is unclear why DDR writes such data. In any event, such track images will trip up the file level restore function in XDR. To circumvent this, a tape to tape copy using XDR will insure that all track images

are in FTR format. It should again be noted that all tape files created by XDR can be used with the file level restore function.

Copying

XDR can copy, at high speed, data between two disks of the same device type. It also copies tape files created by XDR or DDR. Again, INPUT and OUTPUT statements must be defined before the COPY statement is entered. XDR will attempt the best possible method to reduce the elapsed time required to copy the data. If it appears that the two devices are physically different, then I/O overlap is employed. Minidisks appearing to be on the same real physical disk will be copied without trying for I/O overlap. Data is copied without regard to its underlying file system structure, unless the CMS option is specified on the INPUT statement for a disk, or when XDR automatically detects the disk to be in CMS format.

Multiple tracks will be processed for each read and write operation. The IOLIMIT option can control the number of tracks in each I/O. Generally, the default number should suffice. Either the read-multiple or read-track CCW's are used to sweep data into memory; output tracks are created using write-count-key-data CCWs. Any recoverable data checks on input are automatically corrected. XDR should copy disks roughly 4 times faster than DDR. Multiprogramming may affect this ratio, either increasing or decreasing it. However it has been our general experience that XDR's relative performance will improve even more in a multiprogramming environment.

Disaster-Recovery

As mentioned earlier, DDR can restore from tapes created by XDR and vice versa. DDR, when run stand-alone can restore a tape that XDR wrote, regardless of whether COMPACT or MODE XF were specified. Please note, however, that DDR cannot restore from a multi-volume tape file that was written using the standard label (SL) option. DDR will not know to skip over the label information at the start of secondary tape volumes. For single volume tape files, the SKIP option can be used to direct DDR over standard label information. However, with SLOS and FCYL, it is possible to build a procedure which dumps to standard labelled tapes using XDR, and restores, stand-alone, using DDR. Please see the chapter titled "Stand-Alone Restores of Standard Labeled Tapes" elsewhere in this manual.

Command Line Syntax

```
XDR [fn ft [fm]] [( IOLIMIT nn
                    OVERLAP | NOVERLAP
                    NORUN
                    TAPF
                    PROG
```

XDR's control statements are provided either interactively or through a CMS file as specified on the command line. If the fileid is specified, XDR will read and process the control statements in it; otherwise it will issue reads from the terminal which can be satisfied either from the program stack or directly from the terminal. The filemode defaults to "*" if not specified. Any options specified apply for all operations performed under the current invocation of the command.

IOLimit nn

where nn designates the number of tracks to process per I/O operation. nn must be an even divisor of number of tracks per cylinder of the disk unit involved, unless copying from tape to tape. If nn is not an even divisor, it is rounded down to the next such number and an informational message is produced. The current defaults for nn are

Iolim	Device Type
5	3390
5	3380
5	9345
6	3375
10	3350
19	3330
95	FB-512 (3370, 9335)

Sufficient virtual storage must be available in the CMS machine to handle the bufferspace implicitly defined by IOLIM. The amount of free contiguous storage must exceed

$ov * nn * \text{track-size}$

where $ov=2$ for asynchronous I/O operations (OVERLAP), and $ov=1$ for synchronous (NOVERLAP).

OVERlap

will force the use of the SIO or SSCH instructions which permit asynchronous I/O operations under VM. XDR normally detects whether I/O overlap can be achieved, but there may be instances where it must be explicitly instructed to try for it. Overlap implies that two separate I/O buffers be maintained, one for input, one for output. It therefore doubles the amount of virtual storage

necessary to operate in.

NOVErlap

prevents the use of asynchronous I/O operations. XDR will automatically select this mode when appropriate. This option is primarily available for debugging purposes.

NORUN

causes a tape to be rewound wherever it would normally be unloaded. A useful option when testing a new procedure.

TAPF

full tape block reads will be performed during a file level restore operation. XDR normally builds a complicated channel program to extract logical disk blocks from a dump tape. Some hardware configurations cannot handle these complexities, and result in a data overrun condition or a chaining check error. This option causes XDR to read in the entire tape block and then strip out the parts of it needed for the operation. If the tape is in COMPACT form, there is no need to specify this option since its use is implicit. TAPF can be made the default through the XDRINST exec.

PROG

causes an intentional operation exception if an internal abend is encountered. Message SEQXDR999E indicates that such an error has occurred. If the CP command TRACE PROG were entered before running XDR, then when the operation exception is forced, the general registers and memory locations can be inspected to help determine the nature of the error. This option is intended to be used when diagnosing a problem with Sequential.

I/O Definition Statements

Input

```
INPUT cuu type [alt-cuu | volser] [( [REWIND | UNLoad | LEAve]
                                     [SKip nn]
                                     [CMS] [CMST]
                                     [DISPL[#] [(] volser... )] ]
                                     [SL | SLOS]
                                     [EMSG | ESKIP]
                                     )] ]
```

The INPUT statement defines the input device and specifies any special processing needed for it. The device at CUU will immediately be checked for existence and XDR will flag an error if it is undefined or an unknown device type. A consistency check is made between 'type' and the actual device type, with the actual type overriding the specified type when they differ. Some options are applicable only to tape or disk devices, but are still allowed when one is incongruous.

cuu

is a 3 or 4 digit device address defining the input device.

type

specifies the expected device type of cuu. The actual device type will override the type parameter if they differ. Possible values are:

3380, 3390, 3330, 3350, 3375, 9345, FB-512, DASD, DISK
3420, 3430, 3480, 3490, 9347, TAPE

alt-cuu

is a 3 or 4 digit device address defining an alternate tape drive. This device will be used when a multi-volume tape file is encountered on input. It must be the same device type as the primary cuu. The primary cuu is rewound and unloaded before switching to the alt-cuu, except for file level restores. In that case, the tape on the primary cuu is not unloaded unless yet a third tape file is needed. The purpose is to minimize tape mounts for a file level restore operation.

volser

optionally specifies a volume label for a disk device. This label is compared with the real volume label found on the disk and if equal, the operation proceeds. If the volser differs, or if no volser was specified, then message SEQXDR028I is displayed, asking whether or not the operation should continue. No cross checking of the volume label is performed if SCRATCH is entered as the volser.

REWIND

indicates that the tape on cuu (or the alternate) is to be rewound after the operation completes. If a severe error occurs during the operation, such as an I/O error, HX, or otherabend, tape positioning does not take place. This option is accepted but has no effect for a disk device.

UNLoad

indicates that the tape on cuu (or the alternate) is to be rewound and unloaded after the operation completes. If a severe error occurs during the operation, such as an I/O error, HX, or otherabend, tape positioning does not take place. This option is accepted but has no effect for a disk device.

LEAve

indicates that the tape on cuu (or the alternate) will be positioned just after the file processed during the designated operation. If a severe error occurs during the operation, such as an I/O error, HX, or otherabend, tape positioning does not take place. This option is accepted but has no effect for a disk device.

SKip nn

nn tape files are forward spaced over before the operation commences. Use this option to skip over files that do not pertain to the current operation. Logical tape files are skipped when SL is also specified. If several operations are defined under this INPUT statement, then SKIP only has an effect on the first operation. For example,

```
INPUT 181 3420 ( SKIP 2 LEAVE
OUTPUT 191 3380 SCRATCH
RESTORE ALL
* Tape file 3 is used for the above RESTORE
OUTPUT 192 3380 SCRATCH
RESTORE ALL
* Tape file 4 is used; skip is effective only for the first operation.
```

CMS

indicates that the input disk should contain a valid CMS file system. This option reduces VTIME when processing disks using a non-3990 storage control. XDR can automatically detect a disk having a CMS format. With this option, XDR can pre-calculate precisely how much data will be read from each track. Otherwise, there is a tradeoff made between cpu time and elapsed time with the goal of reducing the latter. This option has no effect on input tape units, and is unique to XDR.

CMST

indicates that a CMS file will simulate a tape file. The fileid must be of the form:

XDR_{cuuFs} FILE A

where *cuu* matches the *cuu* on the INPUT statement, and *s* is the relative tape file number to process. In general, this file is created when the CMST option appears on the OUTPUT statement during a DUMP or tape-to-tape COPY. This option may prove useful in test situations where it is inconvenient to have a tape drive attached to the virtual machine. It is unique to XDR.

SL

standard IBM tape labels are present on the input tape. A CMS Labeldef command for "XDRIN" must be issue before running XDR. Particulars about the label information, such as volume id (*volser*) and file name, are communicated through the labeldef to XDR. The "CMS Command Reference" manual fully describes the Labeldef command. If the wrong input tape has been mounted, it will be unloaded with a message sent to the user requesting that the correct tape be mounted. Provide a list of volume ids through the Labeldef for multi-volume tape files. XDR provides enhanced support beyond the normal CMS facility. XDR will check each tape volume for the proper volume id in the order provided through the Labeldef. If SKIP is specified, XDR will position the tape to the logical file number, taking into account header and trailer information.

SLOS

very similar to the SL option, except that OS simulation routines (OPEN, CLOSE) are used to process OS standard labels. A CMS FILEDEF for "XDRIN" must be issued instead; a corresponding LABELDEF may also be active to provide additional information to the OS simulation routines. This option is useful for interfacing with the different VM tape management systems. The following command sequence could be used to read OS standard labels.

```
FILEDEF XDRIN TAPI SL I VOLID 123987
XDR
OUTPUT 190 DASD SCRATCH
INPUT 181 TAPE ( LEAVE SLOS
RESTORE ALL
```

DISPL[#] [(*volser...*)]

loads the 3480/3490 display with the tape *volser* to next mount. If the tape unit is not ready when XDR starts, then the display will alternately blink "MOUNT" and "*volser*" in 2-second intervals. If multi-volume tape processing is called for, then the display will be loaded with the next *volser* to mount. A list can be provided if it is enclosed in parentheses. If DISPL# is specified, then XDR will auto-increment the numeric portion of the last *volser* in the list as additional tapes are required. A few examples.

```
IN 181 TAPE ( LEAVE DISPL 925860
```

will only display 925860 if the tape is not-ready upon commencement.

```
IN 581 3480 ( REW DISPL# ( TUE001 TUE002 TUE003 )
```

will display TUE001, TUE002, TUE003, TUE004, TUE005, etc.

EMSG | ESKIP

allow XDR to continue after a permanent I/O error has occurred. The I/O error would occur on disk for a DUMP operation, and on tape during a RESTORE. In either case, message SEQXDR092W indicates which cylinder and head cannot be processed. The track will be empty after attempting to restore it; the underlying file system may be in error at that point. The file system may have to be repaired after trying to restore the track in error. In any case, either the input disk or input tape has encountered a severe error and attention should be given to it too.

EMSG will issue a prompt on whether or not the operation is to continue, SEQXDR049R, after displaying I/O error messages. ESKIP causes XDR to automatically continue after encountering an input I/O error. Messages are still produced detailing the nature of the error.

Output

```
OUTPUT cuu type [alt-cuu | volser] [( [REWIND | UNLoad | LEAve]
    [MOde {6250|1600|800|38K|XF|COMP|NOCOMP|3490B|3490C}]
    [COmpact] [CCompact] [FCYL]
    [SKip nn] [BLK64K] [SL | SLOS] [DISPL[#] [(| volser...|)] ]
    [CMST]
    |)] ]
```

The OUTPUT statement defines the output device and specifies any special processing needed for it. The device at CUU will immediately be checked for existence and XDR will flag an error if it is undefined or an unknown device type. A consistency check is made between 'type' and the actual device type, with the actual type overriding the specified type when they differ. Some options are applicable only to tape or disk devices, but are still allowed when one is incongruous.

cuu

is a 3 or 4 digit device address defining the output device.

type

specifies the expected device type of cuu. The actual device type will override the type parameter if they differ. Possible values are:

3380, 3390, 3330, 3350, 3375, 9345, FB-512, DISK, DASD
3420, 3430, 3480, 3490, 9347, TAPE

alt-cuu

is a 3 or 4 digit device address defining an alternate tape drive. This device will be used when a multi-volume tape file must be created on output. It must be the same device type as the primary cuu. Before switching to an alternate unit, the current tape is rewound and unloaded, regardless of any options to that effect. If the alternate tape also becomes full, then a flip flop of the primary and alternate units occurs again. This process continues until the operation completes.

volser

optionally specifies a volume label for a disk device. This label is compared with the real volume label found on the disk and if equal, the operation proceeds. If the volser differs, or if no volser was specified, then message SEQXDR028I is displayed, asking whether or not the operation should continue. No cross checking of the volume label is performed if SCRATCH is entered as the volser.

REWIND

indicates that the tape on cuu (or the alternate) is to be rewound after the operation completes. If a severe error occurs during the operation, such as

an I/O error, HX, or other abend, tape positioning does not take place. This option is accepted but has no effect for a disk device.

UNLoad

indicates that the tape on cuu (or the alternate) is to be rewound and unloaded after the operation completes. If a severe error occurs during the operation, such as an I/O error, HX, or other abend, tape positioning does not take place. This option is accepted but has no effect for a disk device.

LEAve

indicates that the tape on cuu (or the alternate) will be positioned just after the file processed during the designated operation. If a severe error occurs during the operation, such as an I/O error, HX, or other abend, tape positioning does not take place. This option is accepted but has no effect for a disk device.

SKip nn

nn tape files are forward spaced over before the operation commences. Use this option to skip over files that do not pertain to the current operation. Logical tape files are skipped over when SL is also specified. If several operations are defined under this OUTPUT statement, then SKIP only has an effect on the first operation. For example,

```
INPUT 191 3380 SCRATCH
OUTPUT 181 3420 (SKIP 2 LEAVE
DUMP ALL
* Tape file 3 is create by the above DUMP
INPUT 192 3380 SCRATCH
DUMP ALL
* Tape file 4 is created; skip is effective only for the first
operation.
```

MOde mmmm

specifies the recording density when writing an output tape file. This option only affects the tape drive if it is at load point. Different tape block sizes are used depending on the MODE value or if BLK64K is present.

Mode	Blksize
800	8192
1600	12288
6250	49152
38K, NOCOMP, 3490B	49152
XF, COMP, 3490C	49152

MODE XF, COMP, or 3490C indicate that data compression is to take place on the hardware level of a 3480/3490 equipped with the IDRC feature. If the feature is not present, the data blocks are written as is. If both MODE XF/COMP/3490C and COMPACT are specified for a 3480, then COMPACT is effectively ignored if the tape is capable of IDRC. CCOMPACT forces

software compression in this case. When the output device is a 3490E, COMPACT can be activated even if the hardware compression has been specified via the MODE option. We have found the hardware compression ratio to be better than that provided through the Compact option. Since Compact takes considerable amounts of CPU time, we would recommend use of hardware compression when available. The usual problem of data-interchange, however, must be considered since some sites may not have tape cartridge systems which support IDRC.

COcompact

activates both a Huffman compression and a run-length encoding algorithm when writing to tape. Each track image is analyzed separately and a particular encoding scheme is used for it. DUMP tapes created with this option are compatible with DDR's RESTORE, and vice versa. However, track images on tape will not match byte for byte. XDR attempts to make improved choices on how data is compressed, and usually dumps slightly different but compatible data. To achieve optimal performance prior to VM/XA and VM/ESA, it is recommended that channels be defined as BMX. This will allow the maximum amount of overlap between the CPU and I/O units. The same performance can also be realized if different virtual channels are used for the disk and tape devices. If both COcompact and MODE XF are specified for a 3480 device, then COcompact is ignored unless CCompact is present.

CCcompact

a unique XDR option that activates the COMPACT algorithm even when MODE XF has been requested. The advantage of this is to decrease the number of bytes sent through the channel to the tape device. Currently, we have no recommendation whether or not the additional CPU time consumed warrants the reduction in channel time.

CMST

indicates that a CMS file will simulate a tape file. The fileid created will be of the form:

```
XDRcuufs FILE A
```

where cuu matches the cuu on the OUTPUT statement, and is restricted to 181 or 182. s is the relative tape file number to process. This option may prove useful in test situations where it is inconvenient to have a tape drive attached to the virtual machine. It is unique to XDR.

SL

standard IBM tape labels are to be written on the output tape. A CMS Labeldef command for "XDROUT" must be issue before running XDR. Particulars about the label information, such as volume id (volser) and file name, are communicated through the labeldef to XDR. The "CMS Command Reference" manual fully describes the Labeldef command. If the wrong output tape has been mounted, a choice will be offered to initialize it. If not, it will be unloaded with a message sent to the user requesting that the correct tape be mounted. Provide a list of volume ids through the Labeldef for multi-volume tape files. XDR provides enhanced support beyond the normal CMS facility. XDR will check each tape volume for the proper volume id in the order provided

through the Labeldef. If SKIP is specified, XDR will position the tape to the logical file number, taking into account header and trailer information. Careful consideration must be given to any stand-alone requirements of tapes with standard labels. Stand-alone DDR can successfully handle a single tape with labels by SKIP-ing over them. Multi-volume tapes with labels will not be processed successfully by stand-alone DDR. Please see the chapter titled Stand-Alone Restores of Standard Labeled Tapes. Our recommendation is to have available a basic VM system upon which XDR can run in a CMS machine. If this basic system is to be built by stand-alone DDR, then the tapes it resides on must either be non-labelled or single tape files.

SLOS

very similar to the SL option, except that OS simulation routines (OPEN, CLOSE) are used to create OS standard labels. A CMS FILEDEF for "XDROUT" must be issued instead; a corresponding LABELDEF may also be active to provide additional information to the OS simulation routines. This option is useful for interfacing with the different VM tape management systems. Stand-alone considerations for SLOS are different than described above for the SL option. Although stand-alone DDR is still incapable of restoring a multi-volume tape file with standard labels, XDR writes only full cylinders (see FCYL) to each tape when SLOS is specified. This allows DDR to restore each tape as a separate job. SKIP I will be required on each INPUT statement to bypass the tape label. Please see the chapter titled Stand-Alone Restores of Standard Labeled Tapes. The following command sequence could be used to write OS standard labels.

```
FILEDEF XDROUT TAPI SL I VOLID 123987
XDR
INPUT 190 DASD SCRATCH
OUTPUT 181 TAPE ( LEAVE SLOS
DUMP ALL
```

DISPL[#] [(] volser... [)]

loads the 3480/3490 display with the tape volser to next mount. If the tape unit is not ready when XDR starts, then the display will alternately blink "MOUNT" and "volser" in 2-second intervals. If multi-volume tape processing is called for, then the display will be loaded with the next volser to mount. A list can be provided if it is enclosed in parentheses. If DISPL# is specified, then XDR will auto-increment the numeric portion of the last volser in the list as additional tapes are required. A few examples.

```
OUT 181 TAPE ( LEAVE DISPL 925860
```

will only display 925860 if the tape is not-ready upon commencement.

```
OUT 581 3480 ( REW DISPL# ( TUE001 TUE002 TUE003 )
```

will display TUE001, TUE002, TUE003, TUE004, TUE005, etc.

BLK64K

unique to XDR, this option forces a tape block size of 65,535, ie., 64K - 1. This option is necessary if a file level restore operation will eventually be performed using a COMPACTed DUMP tape of a 4K 3390. File level restore can only operate when it is known precisely how many tape blocks constitute a track

image (usually 1). A 3390 has a track capacity of approximately 57,000 bytes. The largest tape block size that DDR and XDR will normally choose is 49152. When COMPACT is used, a 3390 track image may be on 1 or 2 tape blocks. File level restore then cannot accurately predict where a particular CMS block is within the tape file. Using BLK64K solves this predicament. The option is also useful when dumping a 3390 without COMPACT. Fewer tape blocks are written, reducing the amount of wasted tape due to inter-record gaps. Please note that using this option precludes using DDR to restore the tape file.

SYsprint {cuu | CONS}

defines the location to direct the cylinder map report to. cuu is restricted to x'000E', ie., the virtual printer used by CMS. CONS designates that the cylinder report should be displayed at the virtual console.

NOPRINT

indicates not to create a cylinder map report for the succeeding operations. This option is unique to XDR.

FCYL

affects DUMP operations by ensuring that only full cylinders are written to each tape volume of a multi-volume set. The SLOS option defaults to this option.

Operation Statements

The general syntax for the DUMP, RESTORE, and COPY operation statements is identical. Here is their general layout, followed by a description of each operation.

```
{DUMP | RESTORE | COPY} { [FTR] cyl1 [TO] cyl2 [REORDER [TO] cyl3] |  
ALL | CPVOL }
```

DUMP

causes disk data to be placed on tape in a unique format. The format is compatible between XDR and DDR. It is essentially an image dump of the input disk, without any regard for the underlying file system on the disk. For example, if an INPUT statement defines a valid CMS minidisk, then all files and all freespace on the disk will be image dumped to tape. If the INPUT statement happens to define an OS or DOS disk, then the VTOC and all tracks on the disk will be dumped to tape. Care of course must be taken when defining the cylinder extents to include in the DUMP all the desired space

ALL indicates that the entire minidisk is to be dumped to tape. The disk size is pre-determined by XDR with help from either CP or the hardware. No I/O error messages are produced when the end of the disk is reached, as is the case with DDR. FTR is the default with XDR. Full tracks are read with a single CCW, and the output tape format is basically the track image laid down on tape. A 32 byte header record prefixes each track image, and then the count, key, and data fields of each record immediately follow.

CPVOL limits the cylinders involved to those marked as PERM or allocated DRCT (directory) as set by CPFORMAT in the allocation byte-map.

cyl1 specifies the beginning cylinder number, or FBA block, to begin dumping from. cyl2 specifies the ending cylinder number to include in the dump. REORDER can be used when a later restore operation will bring the data back to the cyl3 location on disk (REORDER could be specified at restore time with the same effect). Count fields indicating the cylinder number will not be updated to reflect any REORDERing of the data.

Multiple extents, if any, follow after the DUMP statement. When run interactively, a null line indicates that no more extent information is present. Any multiple extents must be entered in ascending order, without overlap.

RESTORE

causes data stored on tape to be written to a disk device. This is the inverse function of the dump operation. Tape files written by either XDR or DDR can be used. An image restore takes place, that is to say, the resulting output disk should be identical byte for byte with the original disk.

cyl1 is the first cylinder number to restore and cyl2 is the last cylinder. Use the REORDER parameter when you wish to restore the original cylinders to some other location on the output disk. Note that imbedded count fields will not be relocated when using REORDER. Subsets of the original disk can be restored by specifying a list of cylinder number pairs following the RESTORE statement. This may be convenient if say a full pack dump was taken, and then several individual minidisks are to be restored.

ALL simply indicates that all the original cylinders are to be restored to the output disk. FTR is meaningless for restore operations.

COPY

indicates that either a disk-to-disk or tape-to-tape copy operation is to be performed. Disks must be of the same device type. Tape copies are only valid when copying a file which was created with either XDR or DDR. At present, only ALL is a valid range for tape copies. FTR is the default and only possibility for copies.

Disk copies operate at high-speed, roughly 4 times faster with XDR than with DDR. The underlying file system is ignored, and the copy proceeds on a physical image basis. The copy can therefore operate with either CMS or non-CMS disks. A subset of the cylinders of the input disk can be selected using the cyl1 and cyl2 parameters. Those cylinders selected can be relocated on the output disk using the REORDER parameter. Note that imbedded count fields are not relocated when using REORDER. ALL indicates that the entire mindisk should be copied. No I/O error message is produced after the cylinder is copied with XDR, as is the case with DDR.

CMSREST

```
CMSREST fn ft [fm] [( BEXTent cyl
DORIG {3390 | 3380 | 3375 | 3350 | 3330 | 9345 | FB-512}
OLDDate
Type LISTOnly
PREErase
```

The CMSREST statement indicates that a CMS file level restore operation is to take place. The INPUT statement defines a tape file in which a previous XDR (or DDR) dump was written. This dump file must at least contain the entire CMS minidisk that the original file resided on. A file or set of files will be extracted from the tape file and written to an output disk. This output disk must be a valid CMS disk and already accessed. Files extracted from the tape will be merged with currently existing files on the output disk, unless an identical fileid is selected. In that case, it will either be skipped or erased depending on whether PREERASE is specified.

The output disk block size must match what the original disk's block size was. However, the device types may differ providing a level of device independence with CMS disks. The DORIG option then defines the original disk device type. Sufficient disk space must of course be available to allow the operation to completely succeed. If many files are being restored and a disk full condition is encountered, then some of those files will be available and others will not. If an alt-cuu is specified on the INPUT statement, then an attempt is made to minimize the number of tape mounts during the restore. A file level restore operation does not proceed straight through a backup file as does the usual RESTORE operation. Rather it performs "random" access within the tape file which sometimes necessitates re-using a tape volume in a multi-volume set.

A multi-volume tape file with standard labels and a VM tape manager is not supported with file level restore. XDR must make multiple passes over the tape file and we currently know of no mechanism to have a previous tape get re-mounted.

File level restore from COMPACTed DUMPs of 4K 3390's requires prior use of the BLK64K option. This sets a larger and incompatible tape block size to use. The discussion of BLK64K explains the reasons why this option may be needed.

fn ft [fm]

designates the file or files to restore. Either one file, many, or all files from the tape can be restored. To select just one file, substitute its filename and filetype. Enter '* *' to select all files from the tape. '*' can be specified for either the fn or ft individually. If fm is specified, then no OUTPUT statement need be entered; one will be defined internally to XDR with SCRATCH as the volser. No crossing checking occurs if fm is present and an OUTPUT statement exists. The fm specification takes precedence.

BEXTent cyl

defines the beginning cylinder number of a CMS minidisk within a tape containing a dump file. This is useful if say a full pack DUMP was taken containing many CMS minidisks. A file level restore can still operate if you know the relative cylinder number of the minidisk in which the desired file exists. XDR will then position the tape at the beginning of the portion where the minidisk was dumped.

DORIG {3390 | 3380 | 3375 | 3350 | 3330 | 9345 | FB-512}

specifies the original disk device type when the dump operation took place. Normally XDR assumes that the dumped disk was the same device type as the current output disk. If they differ, then the DORIG option must be specified. This option also allows a degree of device independence in that minidisks previously dumped with either XDR or DDR can later be restored to a disk with a different device type.

OLDDate

restores files with the same timestamp as when they were dumped. If this is not specified, then restored files are given the current date and time.

Type

causes message SEQXDR077I to be displayed for each file being restored. Note that the file level restore operation must complete before it can be assumed that the files have been restored; merely seeing such messages does not guarantee their restoration.

PREErase

will cause the erasure of an already existing file that matches a file to be restored. This option must be used with a bit of care. If the file level restore is interrupted for some reason after the existing file has been erased, but before the file restore has completed, then the output disk will not have the designated file.

LISTOnly

displays a list of the CMS files that are found on the tape and match the file selection criteria, without actually writing to the output disk.

Stand-Alone Restores of Standard Labeled Tapes

Here are some guidelines to help facilitate the use of standard labeled tapes in a stand-alone environment. An IPL-able version of DDR is used for stand-alone restores since XDR only runs under CMS. If standard labels are not present on the tapes to be restored, then normal DDR restore procedures are used without any changes. Procedures must be modified, however, with standard labeled tapes, in particular, for multi-volume tape files.

The SKIP option found on the OUTPUT statement is used to position a tape after the label information. If logical file "fseq" contains the data to restore, then the physical file number on tape is "(fseq * 3) - 2". For example, if logical file 3 contains the data to restore, use the following statement.

```
INPUT 181 TAPE ( SKIP 7
```

Multi-volume tape files must each be restored as a separate DDR function. This is due to DDR's end-of-volume transitioning logic. When a secondary tape is mounted, DDR assumes that the very first tape block contains its own data. It does not perform a SKIP on any tape but the first in a series. Therefore, DDR sees a "VOLI" record and rejects the tape.

We recommend using the cylinder map report created by XDR during the DUMP operation to drive the RESTORE. The report is sent to the SYSPRINT file, either a printer or console via the SYSPRINT control statement. Each line in the cylinder map indicates the high cylinder dumped to a particular tape volume. Alternatively, message SEQXDR042I can be used which also contains the high cylinder number found on the tape. When the SLOS option is used, XDR will only dump whole cylinders to each tape volume. This presumably makes life easier for DDR to restore the data.

An example is the best way to illustrate this process. Suppose the cylinder map report on the DUMP appeared as follows given these control statements.

```
LABELDEF XDROUT VOLID ?
111AAA
222BBB
333CCC
FILEDEF XDROUT TAP1 SL 1
XDR
SYSPRINT CONS
INPUT 150 DASD SCRATCH
OUTPUT 181 TAPE ( SLOS
DUMP ALL
```

SEQXDR051I	Input	Cylinder	Extents	Output	Cylinder	Extents
SEQXDR052I	Start	Stop		Start	Stop	
SEQXDR053I	0	400		0	400	
SEQXDR053I	401	795		401	795	
SEQXDR053I	796	884		796	884	

There would be three tape volumes used for this set, with volids of 111AAA, 222BBB, 333CCC. Create a tape with an IPL-able version of DDR and set it aside for contingency planning. When needed, IPL from this tape on the real hardware. From the master console, issue the following sequence of commands to restore the 150 disk.

```

SYSPRINT CONS
INPUT 181 TAPE ( SKIP 1 UNLOAD /* Use 111AAA */
OUTPUT 150 DASD SCRATCH
RESTORE 0 400
INPUT 181 TAPE ( SKIP 1 UNLOAD /* Use 222BBB */
RESTORE 401 795
INPUT 181 TAPE ( SKIP 1 UNLOAD /* Use 333CCC */
RESTORE 796 884

```

Alternatively, these control statements could be placed on tape after the IPL-able image of DDR.

If the cylinder boundaries are not recorded somewhere, then a slightly different sequence is needed to restore 150. DDR must be IPL'ed multiple times to effect the restore.

☞ IPL DDR from the "contingency tape." Mount 111AAA on 181.

```

SYSPRINT CONS
INPUT 181 TAPE ( SKIP 1
OUTPUT 150 DASD SCRATCH
RESTORE ALL /* Will restore cylinders 0-400 */

```

☞ DDR will eventually reach end-of-volume and call for the next tape mount. At this point, re-IPL DDR from the "contingency tape." Mount 222BBB on 181.

```

SYSPRINT CONS
INPUT 181 TAPE ( SKIP 1
OUTPUT 150 DASD SCRATCH
RESTORE ALL /* Restores cylinders 401-795 */

```

☞ DDR will again reach end-of-volume and call for the next tape mount. At this point, re-IPL DDR from the "contingency tape." Mount 333CCC on 181.

```

SYSPRINT CONS
INPUT 181 TAPE ( SKIP 1
OUTPUT 150 DASD SCRATCH
RESTORE ALL /* Restores cylinders 796-884 */

```

Of course, if at all possible, it is best to have a rudimentary VM system running in order to use XDR in a CMS machine. XDR's standard labeled tape support can be used to ensure a safe restore operation from the correct tapes. Assume the volid's used for the original dump were 111AAA, 222BBB, and 333CCC. Then the following CMS command sequence should safely restore 150.

```

LABELDEF XDRIN VOLID ?
111AAA
222BBB
333CCC
FILEDEF XDRIN TAP1 SL 1
XDR
SYSPRINT CONS
INPUT 181 TAPE ( SLOS
OUTPUT 150 DASD SCRATCH
RESTORE ALL

```

Working with VM Tape Managers

Using XDR's SLOS option, it is possible to work with the various VM tape management systems provided by other vendors. The examples below are not meant to be complete descriptions of the tape managers. These are meant to be guides in using XDR in conjunction with one of these products. The following trademark notices apply: CA-Dynam is a trademark of Computer Associates International, Inc.; EPIC is a trademark of Legent Corp; VMTAPE is a trademark of Sterling Software, Inc.

Dynam

Use the DYNOPEN command to request tape label processing from CA-Dynam. This command will build the necessary FILEDEF command that is needed by XDR's SLOS option. The volid's will be provided by the Dynam system software package during OPEN, EOVS, and CLOSE processing.

```
DYNOPEN XDROUT DSN my.filename ( OUTPUT WSL TAP1
XDR
SYSPRINT CONS
IN 191 DASD SCRATCH
OUTPUT 181 TAPE ( SLOS LEAVE
DUMP ALL
```

EPIC

Use EPIC's DD command to initiate label processing.

VMTAPE

Ensure that DMSTVI MODULE is on an accessed disk. Issue the following commands, perhaps embedded in an exec. The options specified on the first line may have to be adjusted for your environment.

```
push "QUEUE DEN EXF POOL SILO RETPD 2"
'filedef XDROUT tap1 SL 1 ( SYSPARM (?)'
'LABELDEF XDROUT FID my.filename VOLID SCRATCH'
XDR
SY CONS
IN 191 DASD SCRATCH
OUT 181 TAPE ( SLOS LEAVE
DUMP ALL
```

XDRC

Use the XDRC command to compare two disks.

```
XDRC cuu1 cuu2
```

where cuu1 and cuu2 define the two disks to be compared. They must both be the same device type and have the same size, for example two 15 cylinder 3390's. Message SEQXDC009I is produced if the two disks are identical, byte for byte. All count, key, and data fields are read and checked from each disk, excluding record 0 of each track. No attention is paid to the underlying file system, so all logical files on two disks may be identical, but unused space might show a difference using XDRC.

FB-512 and CKD disks are supported. CKD devices supporting the read-multiple-count-key-data and the read-device-characteristics CCWs are supported. XDRC will not work with storage controllers that do not implement these CCW's. In addition, recovery from correctable data checks using read-multiple is currently limited when using XDRC. Message SEQXDC006W will be issued when such data checks occur. Processing will continue with the next set of tracks to process.

A good estimate of the minimum amount of time necessary to compare two disks is:

$(.55 \text{ seconds}) * (\text{number of cylinders})$

This is fairly accurate for 3380's and 3390's, Multiprogramming/system load will in general lengthen the time necessary to compare the disks.

XDRC Messages

SEQXDC001E Command format is: XDRC cuu1 cuu2

Only two parameters are valid for XDRC, and must be valid hexadecimal disk addresses.

SEQXDC002E Device "cuu" in error, maybe unsupported.

The device is either not defined to the machine, or does not support the read-device characteristics CCW.

SEQXDC003E Disk device types differ.

SEQXDC004E Insufficient storage to run XDRC.

SEQXDC005E Disk sizes are different.

SEQXDC006W Correctable data chk reading dev "cuu", disk addr=CCHH.

XDRC does not currently support recovery from data checks occurring with the read-multiple CCW on a 3880 storage control. CCHH indicates the disk address encountering the error. XDRC will continue processing, but will not compare the track in error, nor the tracks immediately around it.

SEQXDC007E I/O error on "cuu".

A permanent I/O error has occurred reading from the disk.

SEQXDC009I Disks cuu1 and cuu2 compare equal.

This message appears when XDRC found no differences between the two disks. A return code of zero is passed back to the caller.

SEQXDC010E Difference detected at CCHH, offset dddd.

The track indicated by CCHH is different between the two disks. dddd is the offset within the track of the difference, including count and key areas. This message will appear up to 5 times before XDRC terminates. A non-zero return code is passed back to the caller.

SEQXDC011E Error threshold nn exceeded.

XDRC gives up after finding more than nn differences between the two disks.

XDRINST

Use the XDRINST EXEC to set default values in XDR. The module as it comes off the installation tape is in general ready to execute without change. There are a few areas where some customization might be needed and they are described below and elsewhere in this Guide. XDRINST runs interactively and will prompt for the changes to be made. Invoke XDRINST without any parameters.



XDRINST

You will then be given the chance to change the default for the following.

IOLIMIT - can change the maximum number of tracks that will be processed per start I/O operation. Increasing this value results in slightly better elapsed time values at the expense of greater virtual storage use and longer channel programs. Lowering this value might be necessary if, for example, you plan to run large backup operations during the middle of the day and there is only one channel available to the DASD. Further information about IOLIMIT can be found in the "Command Line Syntax" section. Each DASD device on the system can have a different IOLIMIT value associated with it.

TAPF - changes XDR's tape channel program structure during file-level-restore operations. Certain hardware configurations will encounter data-overflow I/O errors on XDR's complicated tape input channel programs. This should only occur on a file-level-restore from a non-COMPACTed tape. Setting this default should have minimal impact on elapsed time results.

XDRTIME

Use the XDRTIME EXEC to compare the performance of two different commands. It displays the vtime, ttime, elapsed time, and I/O count required by each command. The arguments entered for XDRTIME are passed on to the commands to be measured. In the form in which it is provided, the EXEC compares DDR and XDR. Therefore, in the following example,

```
XDRTIME DUMP CARDS A
```

the two commands are timed for the operation specified in the CMS file "DUMP CARDS A", presumably dumping data from disk to tape.

XDRTP

XDRTP is a utility program for reading and displaying tape blocks. Its primary purposes is for troubleshooting tape problems. Its format is as follows.

XDRTP cuu [nn]

where cuu is the tape device address to read from. If nn is not specified, then the next physical tape block is read and displayed. nn can be used to position the tape and another tape block, either in a forward or backward direction. A backward direction is indicated by a negative nn value. Only the first 64 bytes of a tape block is displayed.

This program runs in the user area of storage, only at location x'20000'. About 8K bytes beyond the module is used to read in the tape block; no DMSFREE or CMSSTOR is used to obtain this memory. Any information stored within this range will be lost.

XDRTP Messages

SEQRTP001E CUU "cuu" parm in error.

The cuu address shown is not valid or not defined to the virtual machine.

SEQRTP002E Error forward spacing blocks.

An I/O error probably occurred while trying to position the tape.

SEQRTP003E FSB value "nn" is invalid.

The nn parameter on the command line is wrong.

SEQRTP004E Error reading tape block.

An I/O error occurred reading the tape, or a tape mark or a reflector strip was encountered.

SEQRTP005I xxxx,xxxx,xxxx,xxxx iiii

This is the message used to display the first 64 bytes of a tape block.

SEQRTP006I Data blocksize is "nn".

The utility attempts to read in 64K-1 bytes from the tape. nn is 64K-1-"the residual count".

ZZAP

Use ZZAP to change the contents of a CMS module file. Statements indicating the changes to be made are supplied either through a CMS file or through the console. These statements are syntax compatible with the regular CMS ZAP program. ZZAP runs faster than ZAP as it only updates the module on disk when there is no more zapping to perform. If no filename is specified, ZZAP prompts at the terminal for zap statements. The filetype defaults to "ZAP" if not entered, and the filemode defaults to "A". The command format is as follows.

```
ZZAP [ filename ] [filetype [filemode]] [( options [ ) ]]
```

ZZAP can prevent most typographical errors if special hash-check codes are included following each VER and REP statement. The hash-check code will be indicated by a colon followed by two hexadecimal digits. An error message is produced and the zap terminates if the hash-check fails. A recommended way of applying zaps would be to run ZZAP with the VERO option to check the verify data and hash-check numbers, if any, before actually trying to update the module on disk.

```
ZZAP myfix (VERO
```

Options

HASH

will cause the zap file to be re-written with hash-check numbers on each VER and REP statement. This option is generally used by Sequential before distributing a fix.

MAINT

suppresses any verify reject messages at the terminal. It also displays a list of zaps successfully applied. This option is used by Sequential when distributing a set of zaps constituting some maintenance level.

VEROnly

processes just the verify statements. The module is not updated on disk. This is useful to check that all verify data is correct before applying maintenance. It can help prevent a multi-csect zap from being partially applied, when part of a zap verifies correctly and another part does not.

UNZap

inverts the usual meaning of VERify and REPlace statements. This option can back-off a previously applied zap. When used in conjunction with VEROnly, a module can be checked to insure that a particular zap is applied. Specifically, VEROnly prevents the module from being updated and UNZap causes the REP statements to be used to verify data.

Examples

Backup a Series of Minidisks to One Tape

```
SYSPRINT 00E
INPUT 200 3380 SCRATCH
OUTPUT 181 TAPE (MO 6250 LEAVE COMPACT
DUMP ALL
INPUT 201 3380 SCRATCH
DUMP ALL
INPUT 202 3380 SCRATCH
DUMP ALL
```

Restore a Disk From the Second Tape file

```
SYSPRINT CONS
INPUT 181 3480 (SKIP 1 UNLOAD
OUTPUT 201 3380 SCRATCH
RESTORE ALL
```

Restore a Single File from a Dump Tape

```
SYSPRINT CONS
INPUT 181 3480 (REWIND
CMSREST profile exec a (TYPE OLDDATE
```

Restore Many Files to a Disk of an Unlike Type

```
SYSPRINT CONS
INPUT 181 3480 (REWIND
CMSREST * * O (DORIG 3350 OLDDATE PREERASE
```

Restore a Single CMS File from a Dump of an Entire Disk Pack

```
SYSPRINT CONS
INPUT 181 3420 182 (LEAVE
CMSREST OUR MACLIB Q (BEXTENT 125 OLDDATE TYPE
```

Copy One Minidisk to Another

```
SYSPRINT CONS
INPUT 191 3380 CMS191
OUTPUT 2B1 3380 SCRATCH
COPY ALL
```

Messages

SEQXDR001E Diag 24 failed for device *cuu*, RC=*nn*.

*The device at address *cuu* could not be queried with respect to its device type via Diagnose 24. *nn* is the return code from the diagnose. It most likely is not defined to the virtual machine.*

SEQXDR002E Input & output device types differ.

A copy operation can only take place between two similar device types.

SEQXDR003E Error *nn* from SUBPOOL *ssss*.

*XDR attempted to define a SUBPOOL but encountered some unusual error. *nn* is the return code from the SUBPOOL create function.*

SEQXDR004E Insufficient virtual storage available to run XDR.

Increasing the virtual storage of the virtual machine should alleviate this condition. Releasing disks, dropping nucleus extensions, or performing other commands to free up storage may also help. The IOLIMIT and NOVERLAP options of XDR can be used to throttle the amount of storage it uses.

SEQXDR005E I/O error occurred on device *cuu*. CSW: *cccc,cccc* Sense: *ssss*.

This usually indicates a serious problem. Either a real I/O error is occurring on the device, or XDR has encountered an internal error. The CSW provides information relevant to the I/O operation at the time of the error. Under XA-mode, it is a simulated CSW. The first 4 bytes of sense information is shown to aid in problem solving. Please call Sequential Software if further assistance is required.

SEQXDR006E Iolimit value of *ii* is overridden by *oo*.

*The IOLIMIT value that was specified on the command line is not valid for the devices for the current operation. Either the value is too high or is not an even divisor of the number of tracks per cylinder. The operation continues using *oo*.*

SEQXDR007E Insufficient virtual storage.

Same as for message SEQXDR004E.

SEQXDR009E Device type of “*dddd*” is not supported.

SEQXDR010E Output disk is read/only.

A copy or restore function requires a write link to the output disk.

SEQXDR011E Disk *cuu* does not exist.

SEQXDR013E Error *nn* during *ssss* of input file.

An error occurred while processing the sysin input file. nn is the return code when using the ssss macro.

SEQXDR014E File “fn ft” is not found.

The file for providing XDR control statements cannot be found.

SEQXDR015E Lrecl too large in file “fn ft fm”.

The lrecl of the file for providing XDR control statements must not exceed 133.

SEQXDR016E Missing parameter(s) on “ssss” statement.

The statement designated by ssss is not complete and requires more information.

SEQXDR017E Invalid parameter “pppp” on the “ssss” statement.

Statement ssss contains an incorrect value.

SEQXDR018E Statement “ssss” is not valid.

Each input statement must start with a valid control name, such as INPUT, DUMP, etc. ssss is not one of these.

SEQXDR019E No extents specified for ssss operation.

The operation statement has no extent information defined for it. Either specify “ALL” or provide a list of cylinder/block extents to process.

SEQXDR020E Fileid for statements is not complete.

The fileid defining where the control statements come from must have at least a filename and a filetype. The filemode will default to the first occurrence of this file.

SEQXDR021I Calling IBM DDR routine required for “ssss”.

XDR cannot handle a particular feature supported by DDR. Please see a chapter in this User’s Guide which lists unsupported features. An attempt will be made to call the standard DDR utility. The control statements accumulated by XDR will be stacked for DDR to read in terminal mode. Several “ENTER:” prompts from DDR will appear which can be ignored.

SEQXDR022I Blank line entered.

SEQXDR023E Extent list area exhausted, please call Sequential Software.

This message indicates that the amount of virtual storage reserved for building extent list information has been exhausted. It practically should never appear.

SEQXDR024E Device type “ssss” is incorrect for “cuu”.

The device type from an INPUT or OUTPUT statement does not match the real device type for cuu.

SEQXDR025E Device “cuu” is not suitable for ssss operation.

The input or output device must be of particular types for a given operation. For example, a DUMP reads a disk and writes to tape, a RESTORE reads a tape and writes to disk.

SEQXDR026E Unable to process tape file.

Inconsistent information has been found in the header records written to tape. Either the wrong tape is mounted or some other unusual error has occurred.

SEQXDR027E Partial tape-to-tape copy is not supported.

Only ALL the data can be copied from on tape file to another, not a subset selection.

SEQXDR028I {Input | Output} cuu volser read is "volser" [not "volser2"] Continue? (Y/N)

A volser has been read from either the input or output disk. If this is the correct one, enter "Y" to continue or "N" to stop.

SEQXDR029I Failing CCW: cccc,cccc Arg: aaaa,aaaa,aaaa,aaaa

This message generally follows SEQXDR005E. It shows the CCW causing the I/O error.

SEQXDR030I Intervention required on device "cuu".

The tape at addresscuu requires manual intervention to resume operations. XDR will wait for the tape to become ready. An HX can terminate it if necessary.

SEQXDR031E FTR option not used for track image dumped.

Please call Sequential Software for assistance if this message appears. It may simply indicate that a tape contains a DUMP of an older device type that cannot be restored; however, no known situations of a supported XDR disk should encounter this.

SEQXDR032E Given extents cannot be processed.

For CKD devices, this message indicates that one of the following conditions is true: non-ascending extent limits have been entered; overlapping extent information entered; extents exceed size of the output disk. For FB-512 disks, the block-extents provided cannot be processed; please call Sequential Software for assistance.

SEQXDR033W Specified mode not supported on device "cuu".

The mode given on the OUTPUT statement is not capable on device cuu. Processing continues using the default density for the device.

SEQXDR034I No volume label found on {input | output} "cuu". Continue? (Y/N)

An input or output disk does not have a valid volume label on it. Processing continues if "Y" is entered as a response.

SEQXDR035I {DUMP | RESTORE | CMSREST | COPY} ING volser.

Indicates the operation in progress on the device with volser as a volume label. This label indicates the original input disk's during a restore-type operation.

SEQXDR036E Missing or invalid {INPUT | OUTPUT} statement.

Valid INPUT and OUTPUT statements must be entered before an operation can commence. An exception to this is a file level restore where an OUTPUT statement can be omitted if a filemode letter is provided.

SEQXDR037I Disk “cuu” must be accessed to use the CMS option.

SEQXDR038I Sysprint device address restricted to 00E.

The only valid device address for the printed cylinder map is X'00E'. Alternatively, CONS can be specified to direct output to the terminal, or NOPRINT can be used to suppress the map entirely.

SEQXDR039I

A blank line used as a filler.

SEQXDR040I Option “ssss” is invalid.

The command line option ssss is incorrect.

SEQXDR041E Alternate tape “cuu1” device type differs from primary device “cuu2”.

Any alternate tape units defined on the INPUT or OUTPUT statements must be of the same type as the primary unit.

SEQXDR042I End of volume, last CCHH/BLOCK “nnnn”. Mount next tape.

For a restore from a multi-volume tape file, nnnn is the last disk address processed from the current tape. It is in hexadecimal format.

SEQXDR043E Wrong input tape mounted. Please retry.

XDR checks the volume header for multi-volume tape files trying to insure that the correct tape has been mounted. While it cannot guarantee to prevent processing from the wrong tape, XDR does check that the timestamp recorded in the volume header is consistent from tape to tape. This message appears when a mis-match has occurred.

SEQXDR044W Tape “cuu” is write-protected. Please set the tape for writing.

XDR unloads tape unit cuu and waits for the drive to become ready again, presumably with writing enable for the tape.

SEQXDR045W XDR is not licensed for this CPU: cpuid.

A reminder message that XDR is not licensed for the processor displayed. Please contact Sequential Software if this message appears.

SEQXDR046W XDR expires in nn days!

nn days remain for the current module to continuing running. After that date, the following message will appear.

SEQXDR047E XDR has expired. Please call Sequential Software, Inc. at (201) 652-3300.

SEQXDR048W Original disk size, size1, is larger than the output disk size, size2. Continue? (Y/N)

A restore or copy function may encounter this problem for the indicated reason. Subsequent use of the disk may be unpredictable if you choose to continue the operation.

SEQXDR049R Continue? (Y/N)

Neither a "Y" or "N" was entered in response to a previous prompt, or the EMSG option was specified and a prompt to continue resulted. Please enter "Y" or "N".

SEQXDR050I {DUMP | RESTORE | CMSREST | COPY} Data mm/dd/yy at hh:mm:ss zone from volser

The specified operation is commencing. The timestamp is the current time for a DUMP or COPY. For RESTORE and CMSREST, it shows the time that the dump file was created. zone is the timezone. volser is the input disk's volume label for a DUMP or COPY, or is the original volume label for a restore type operation.

SEQXDR051I Input Cylinder Extents Output Cylinder Extents

SEQXDR052I	Start	Stop	Start	Stop
SEQXDR053I	nnn1	nnn2	nnn3	nnn3

These three messages constitute the cylinder map report. They are written to either the printer or console as directed by the SYSPRINT statement. The cylinder/block numbers summarize the extents that have been processed.

SEQXDR054I XDR 1.20A (C) Sequential Software 1992. For use at siteid

An informational message showing where this routine came from, and where it is installed.

SEQXDR055 - SEQXDR059

Reserved for future use.

SEQXDR060I Data block count too high in file "fileid". DBC=nn, RECS=rr.

A file has been encountered during a file level restored whose structure is a bit odd, but should otherwise be legitimate. DBC indicates the number of data blocks and RECS the number of records in the file.

SEQXDR061I File “fileid” is sharing block number nn.

Two files selected for a file level restore are sharing the same disk block. This condition existed on the original disk. The best remedy is to follow the current operation with a file level restore for just fileid. However, the data in the file is suspect.

SEQXDR062I File “fileid” has a bad file structure.

The indicated file has a corrupted index block. This condition existed on the original disk. XDR will not be able to restore this file. A complete RESTORE operation might be run to recover the disk in its original entirety.

SEQXDR063I File level restore cannot complete, code:nn

Some consistency error has occurred within XDR. Please call Sequential Software if this message appears.

SEQXDR064I Data block count discrepancy. IB=ii, IEX=jj, BM=kk.

A file level restore has uncovered some consistency error that may or may not be fatal. The operation proceeds, but might shortly fall on its face. The three values displayed show respectively: the number of total blocks to restore (from the FST's), the number of level 2 blocks, and the number of blocks to restore (from all file's index blocks).

SEQXDR065E Reserved minidisks cannot be resized.**SEQXDR066E File level restore cannot operate when track capacity>tape blksize.**

This message is produced if attempting a file level restore and the tape block size is less than the original disk track capacity, and the tape is in COMPACT form. This typically might result CMSRESTing a 4K 3390. Use the BLK64K option when dumping a 3390 if you later want to CMSREST it. Choose a different MODE for other devices causing this message.

SEQXDR067E Original disk was not in CMS format, file level restore cannot continue.

XDR checks the tape file containing the dump to see if the original volume label begins with “CMSI”. If it does not, then this message appears. Some positioning error of the tape might be the culprit. The DORIG option may be needed to remind XDR of the original device type. Insure the first tape of a multi-volume set is mounted.

SEQXDR068I End of job.

This message appears when XDR is returning to CMS.

SEQXDR069E Incorrect fileid on CMSREST statement.**SEQXDR070E File level restore option “ssss” is not valid.**

ssss is not a valid option on the CMSREST statement.

SEQXDR071E Invalid character in fileid “fn ft”.

Inspect fn ft for illegal characters. Some CMS systems allow unconventional characters in fileids. Please contact

Sequential Software if this is the problem.

SEQXDR072E Disk “cuu” must be accessed to perform file level restores.

When a filemode is not given on the CMSREST statement, XDR uses the cuu from the OUTPUT statement to locate the accessed disk to restore to. If cuu is not among the accessed disks, then this message appears and the operation terminates.

SEQXDR073I File “fileid” already exists. Use PREERASE to replace.

A file conflict has arisen during a file level restore. The indicated file is already present on the output disk and will not be altered. To replace the file, the PREERASE option must be specified on the CMSREST statement. If this is a multi-file restore, the operation continues with other files possibly being restored.

SEQXDR074E No files match restore specification.

XDR cannot find any files within the tape that match the restore criteria. This is usually a user error, unless some other problem is occurring.

SEQXDR075I nn files match file specification.

This indicates the number of files that will be restored, barring further problem.

SEQXDR076I The following files will be restored:

SEQXDR077I nn blocks from “fileidS”.

These messages appear when the TYPE option has been selected. A list of all the files to restore appears. Note that the appearance of these messages does not guarantee that the files have been restored. The operation must still continue without error.

SEQXDR078E Output disk blksize is not the same as original disk. File level restore halting.

XDR can only perform a file level restore to a disk matching the original disk block size.

SEQXDR079E Disk “mm” is full.

The indicated disk is full, however as many files as is possible that match the restore criteria will be written. This may be zero or more.

SEQXDR080W Abend/HX occurred at offset aaaaaa

This message appears during an abnormal termination of XDR. It may be useful to write down the address to aid in further problem determination, unless of course a simple HX is the cause. aaaaaa is an offset within XDR of the offending instruction.

SEQXDR081E SFS directories cannot be restored to.

Only CMS minidisks can participate in a file level restore operation. The Shared File System is ineligible.

SEQXDR082E DORIG of “ssss” is incorrect or not supported.

The DORIG option can tell XDR what the original disk device type was. This message appears when an invalid type is specified.

SEQXDR083W DORIG option possibly is incorrect.

This message may follow SEQXDR067E or SEQXDR078E. If the original disk device type does not match the current output type, then you must use the DORIG option to tell XDR of this.

SEQXDR084E Begin extent “nn” is not found.

The BEXTENT option specifies a cylinder/block that cannot be located in the current tape file. Tape positioning after this message is undefined.

SEQXDR085I Mount tape #nn at address cuu.

A multi-volume tape file is being used during a file level restore. nn indicates the relative tape number to mount at address cuu. XDR will attempt to retain tapes if an alternate device address was provided on the INPUT statement.

SEQXDR086E Disk “mode” is not accessed.

SEQXDR087E XDR is not loaded as a nucext, DDR cannot be called.

A function has been requested that only DDR can handle. DDR will automatically be called if XDR is running as a nucleus extension. This message indicates that XDR is not running as a nucleus extension and therefore cannot call DDR.

SEQXDR088W ERPA code “xx” encountered on tape cuu.

xx is an error recovery procedure action code set by a 3480 or 3490 tape unit. Some error codes are not serious and the operation will proceed normally. Other codes are serious and will be followed by an SEQXDR005E error message.

SEQXDR090I Bytes in: nn. Bytes out: mm.

On a DUMP, nn indicates the number of bytes fed into the compaction routine and mm is the number of bytes written to tape. This values include a 32-byte header record for each track image, though it itself is not compacted. On a RESTORE, nn is the number of bytes read from tape; mm is the number written to disk.

SEQXDR091I nn tracks were not compacted.

nn disk tracks could not be reduced in size through the COMPACT algorithm.

SEQXDR092W Cyl cccc head hh cannot be processed.

This message accompanies SEQXDR005E, indicating what disk track is affected by the I/O error. If either EMMSG or ESKIP were specified, this message would then indicate the troublesome track.

SEQXDR093E Error nn from TAPESL routine on ssssss.

An unexpected error has occurred processing standard labelled tapes. The nn code can be found in the CMS Macro Reference or more recently in CMS Application Development Reference for Assembler manual under TAPESL.

SEQXDR094E Output tape does not have a VOLI label.

The first (or only) output tape used during a DUMP must have a VOLI label if it is at load point.

**SEQXDR095E Wrong output tape mounted: vvvv1 mounted instead of vvvv2.
Retry.**

vvv1 was mounted instead of vvv2. The tape is unloaded and XDR waits until the drive becomes ready.

SEQXDR096E Missing LABELDEF for {Input | Output}.

Standard label tape processing requires a Labeldef of XDRIN for Input or XDROUT for output.

SEQXDR097E Insufficient VOLIDs provided for {Input | Output}.

Fewer volids were provided through the Labeldef than apparently are needed to process a multi-volume tape file.

SEQXDR098E Chaining check during FLR. Call us at 201-652-3300.

Certain hardware configurations cannot support the complicated tape channel programs that XDR builds to perform a file level restore operation. The TAPF command line option should solve this problem. Also, the XDRINST exec can make TAPF the default. This message should only appear when attempting a file level restore of a non-compacted tape. Please call Sequential Software for further assistance.

SEQXDR099I End of operation for volser.

SEQXDRI00R Enter:

SEQXDRI01R Enter next extent or null line:

SEQXDRI03E Disk cuu is not a CPVOL.

SEQXDR999E XDR internal abend ssss(nn).

A logic error has been encountered within XDR. ssss indicates the failing routine, with a particular error number, nn, within that routine. Please call Sequential Software for assistance. If possible, please preserve both the input and output devices in their current status to help in solving the problem.

Summary of Unsupported DDR Functions

These functions cause IBM DDR to be invoked from within XDR. Accumulated control statements are stacked, and then DDR is invoked in "terminal" mode. Several "Enter:" prompts may appear as DDR reads the control statements; they can safely be ignored. DDR messages will appear as the command proceeds. When the particular command is finished, control should return to XDR to proceed with any further commands.

NUCLEUS
TYPE
PRINT

In addition, stand-alone support is not available with XDR. However, tapes created by XDR are compatible with stand-alone DDR.

Index

A

ALL 21, 22, 33
Alternate cuu 12, 16, 22

B

BEXTENT 22, 23, 33
BLK64K 17, 19, 22, 39
Blksize 17, 19, 22, 39

C

CCOMPACT 17, 18
CCW 6, 9, 21
CMS 9, 12, 13
CMSREST 8, 22, 33
CMST 12, 13, 16, 18
COMPACT 5, 9, 11, 19, 22, 39, 43
compare 27
COPY 5, 9, 21, 22, 33
CPVOL 21, 42
cuu 12, 16
cylinder map report 24, 38

D

data check 6, 14, 27
Density 8. *See also* MODE
Device independence 22. *See also* File level restore
device type 12, 16, 23
disaster-recovery 9, 24
Disks 5, 12, 16, 22, 27, 29
 3390 20, 22, 39
 3990 5
 9340 5
DISPL 14, 19
DORIG 22, 23, 33
DUMP 5, 8, 21, 33

E

ECKD 5
EMSG 6, 14
ESKIP 6, 14
Extents 21

F

FCYL 16, 19, 20, 24
File level restore 5, 8, 19, 22, 33, 42. *See also*
 Device independence

FILEDEF 25, 26
Fileid 23
FTR 8, 21

H

HASH 32
Header records 5, 8, 21

I

I/O errors 6, 11, 15, 29, 42
INPUT 8, 9, 12, 33
Installation 7
Iolimit 6, 7, 9, 10, 29

L

Labeldef 5, 14, 18, 25
LEAVE 12, 13, 16, 17, 33
LISTONLY 22, 23

M

MAINT 32
maintenance 32
MODE 9, 16, 17, 18, 33
multi-volume 14, 18, 19, 24

N

NOPRINT 20
NORUN 11
Noverlap 10, 11
NUCLEUS 43

O

OLDDATE 22, 23, 33
OUTPUT 8, 9, 16, 23, 24, 33
Overlap 5, 6, 9, 10, 18

P

Performance 5, 9, 13, 18, 22, 29, 30
PREERASE 22, 23, 33
PRINT 43
PROG 11

R

read-multiple-ckd 27
REORDER 21, 22
RESTORE 5, 8–9, 21, 33
REWIND 8, 12, 13, 16, 33

S

SKIP 8, 9, 12, 13, 14, 16, 17, 19, 24, 33

SL 5, 9, 14, 18
SLOS 14, 19, 20, 26
stand-alone 9, 19, 24, 43
standard labels 5, 9, 14, 18, 19, 24, 26
SYSPRINT 20, 24

T

tape management systems 14, 19, 26
Tapes 12, 13, 14, 16, 19
TAPF 7, 11, 29, 42
TYPE 22, 23, 33, 43

U

UNLOAD 11, 12, 13, 16, 17, 33
UNZAP 32

V

VERONLY 32
Virtual storage 5, 10, 27
VM 5, 25, 26
Volser 12, 14, 16, 19, 26
Vtime/Ttime 6, 13, 18

X

XDRC 7, 27
XDRINST 7, 29
XDRTIME 7, 30

Z

zapping 7, 32
ZZAP 32