

CS **COLE SOFTWARE, LLC**
736 Fox Hollow Road • Afton, VA 22920
540-456-8210 • 540-456-6658: FAX
INTERNET: <http://www.colesoft.com>
E-MAIL: techsupt@colesoft.com

z/XDC[®]
RELEASE GUIDE

z/XDC[®] Release z1.3
for
z/OS and OS/390

David B. Cole

z/XDC[®] z1.3 RELEASE GUIDE

PREFACE

PROPRIETARY LEGEND

z/XDC[®] and its documentation (collectively, "Product"), including copies thereof, are the confidential and proprietary property of Cole Software, LLC ("Owner"). The Product may be used only by those organizations that are licensed by Owner for such use and only in the manner so licensed. The program and documentation may not be published, reproduced, distributed, or made available to third parties for any purpose without the expressed written permission of Owner; however, a reasonable number of copies may be made of the documentation (including the copyright notices and proprietary legends thereon) as is necessary for the legitimate use of the Product within a licensed organization.

Except as may be otherwise expressed in a signed agreement between Owner and Customer, Owner makes no representations or warranties, expressed or implied, including, but not limited to, the implied warranties of merchantability and fitness for a particular purpose, the warranty of freedom from rightful claims by way of infringement and the like, and any warranty as to accuracy.

WARNING! z/XDC[®] is a powerful tool for dynamically locating and correcting malfunctions in actively executing user programs and operating system programs and subroutines. Accordingly, it is inherent in its design, that unless the use of this Product is properly controlled, then under certain conditions a malicious or careless user can use the Product to alter, subvert, counterfeit, damage or otherwise disturb the normal execution of user programs or system routines including, under certain conditions, both its own and system security routines.

Therefore, even if advised of the possibility of loss or damages, under no circumstances shall Owner be liable for any loss or damage whatsoever (including death) arising from the Product, whether such loss or damage be direct, indirect, consequential, special or otherwise. Further, Owner shall not be obligated to indemnify any user of the Product in any manner for any loss which the user or anyone else may experience, of any kind or nature, arising out of the use or misuse of the Product.

CONTACTING COLE SOFTWARE

The XDC[®] family of products are marketed by **COLE SOFTWARE, LLC** with its principal office in Afton, Virginia. If you would like more information, please contact COLE SOFTWARE Marketing as follows:

Phone: **800-XDC-5150** or **928-771-2003**
FAX: **928-771-2005**
E-Mail: sales@colesoft.com
Home Page: <http://www.colesoft.com>

Our Technical Support contacts are:

Phone: **540-456-8210**
FAX: **540-456-6658**
E-Mail: techsupt@colesoft.com
Home Page: <http://www.colesoft.com>
FTP site: <ftp.colesoft.com>

Our Customer Services contacts are:

Phone: **540-456-8210**
FAX: **540-456-6658**
E-Mail: support@colesoft.com
Home Page: <http://www.colesoft.com>

z/XDC[®] z1.3 RELEASE GUIDE

(Preface)

Our snail mail address is:

Address: **Cole Software, LLC**
736 Fox Hollow Road
Afton, Virginia 22920
USA

Our home page provides the following services:

- General information about z/XDC.
- E-mail links to both Marketing, Technical Support, and Customer Services.
- FTP links for uploading diagnostic information and other files to Technical Support.
- FTP links for downloading current maintenance for z/XDC.
- Links permitting existing customers to download a full set of z/XDC's documentation.
- An order form for obtaining an upgrade of XDC to its current version (z/XDC) and release (z1.3).

TRADEMARKS

TFS[™], **XDC-TFS[™]**, **CDF[™]**, **XDC-CDF[™]**, and **FASM[™]** are trademarks of Cole Software, LLC. **Extended Debugging Controller[®]**, **XDC[®]**, **XDC-SE[®]**, and **z/XDC[®]** are registered trademarks of Cole Software, LLC. Other brand and product names referenced in this document are trademarks or registered trademarks of their various holders. Use of their names herein is for identification purposes only.

ADDITIONAL MANUALS

z/XDC customers may make as many copies of this manual as they feel is necessary for the legitimate use of z/XDC within their organization. Existing customers may download from our web site (www.colesoft.com) printable copies of all of z/XDC's manuals. Each manual is available in PDF format.

In addition, all manuals (except the Installation Guide) can be printed directly from within z/XDC itself. To print your own set of manuals, start an z/XDC debugging session (example: XDCCALL IEFBR14), then issue the following commands:

```
PRINT HELP USERGUIDE;SET PRINT CLOSE
PRINT HELP COMMANDS;SET PRINT CLOSE
PRINT HELP MESSAGES;SET PRINT CLOSE
PRINT HELP WHATSNEW Z13;SET PRINT CLOSE
```

Alternatively, you also can print these manuals by issuing z/XDC's **READ** command to run the MANUALS member of z/XDC's script library. Example: **READ DBCOLE¹.XDCZ13.XDCCMDS(MANUALS)**.

You also may print a **Quick Reference** for z/XDC by issuing z/XDC's **READ** command to run the QUICKREF member of z/XDC's script library. Example: **READ DBCOLE.XDCZ13.XDCCMDS(QUICKREF)**.

For more information about using the PRINT HELP and related commands, see **HELP HELP PRINTING**.

¹The library's high level qualifier may be different at your data center. Please ask your Systems Programmer.

z/XDC® z1.3 RELEASE GUIDE

CONTENTS

PREFACE	<u>ii</u>
PROPRIETARY LEGEND	<u>ii</u>
CONTACTING COLE SOFTWARE	<u>ii</u>
TRADEMARKS	<u>iii</u>
ADDITIONAL MANUALS	<u>iii</u>
CONTENTS	<u>iv</u>
INTRODUCTION	<u>1</u>
A Roadmap	<u>1</u>
Online Help Panels	<u>3</u>
Help Whatsnew	<u>3</u>
Help Whatsnew Z13	<u>3</u>
Help Whatsnew Z13 Autocloning	<u>4</u>
Help Whatsnew Z13 CAsensitivity	<u>4</u>
Help Whatsnew Z13 COmmands	<u>5</u>
Help Whatsnew Z13 COmmands DEletemaps	<u>6</u>
Help Whatsnew Z13 COmmands DMap	<u>6</u>
Help Whatsnew Z13 COmmands Equate	<u>7</u>
Help Whatsnew Z13 COmmands FInd	<u>8</u>
Help Whatsnew Z13 COmmands FOrmat	<u>8</u>
Help Whatsnew Z13 COmmands Indirect	<u>9</u>
Help Whatsnew Z13 COmmands LISTLked	<u>9</u>
Help Whatsnew Z13 COmmands LISTMap	<u>10</u>
Help Whatsnew Z13 COmmands Map	<u>10</u>
Help Whatsnew Z13 COmmands SEtformat	<u>11</u>
Help Whatsnew Z13 COmmands SHow	<u>12</u>
Help Whatsnew Z13 COmmands Using	<u>12</u>
Help Whatsnew Z13 COmmands Where	<u>13</u>
Help Whatsnew Z13 Equates	<u>14</u>
Help Whatsnew Z13 Help	<u>14</u>
Help Whatsnew Z13 MApdisplays	<u>15</u>
Help Whatsnew Z13 Programobjects	<u>16</u>
Help Whatsnew Z13 Scripts	<u>16</u>
Help Whatsnew Z13 MIsellaneous	<u>17</u>
INDEX	<u>18</u>

z/XDC[®] z1.3 RELEASE GUIDE

z/XDC[®] z1.3 RELEASE GUIDE

INTRODUCTION

Cole Software has pursued the goal of making z/XDC's online documentation as comprehensive as possible. Towards that end, we have devoted considerable effort to greatly expanding the amount of information online and to improving the accessibility of that information and the navigability of the Online Help database as a whole.

This manual is nothing more than a printout of a section of the Online Help database. It is provided for those people (like myself) who steadfastly prefer looking at paper instead of glass. (It's hard to write margin notes on glass.)

The information in the Online Help database has been segmented into five printed documents:

- **z/XDC[®] User Guide**
Contains comprehensive tutorials about the many features and capabilities of z/XDC.
- **z/XDC[®] Commands**
Contains the detailed syntax, usage descriptions, and examples of all of z/XDC's commands.
- **z/XDC[®] Messages**
Contains descriptions of all of the messages that can be issued by z/XDC and all of its various components.
- **z/XDC[®] z1.3 Release Guide**
Contains a history of all changes and upgrades made in the current release of z/XDC.
- **z/XDC[®] Quick Reference**
Contains brief lists of z/XDC commands, built-in equates, and other information.

There are a couple of important structural differences between the Online Help and these manuals:

- When the Help panels are displayed online, a large number of "hyperlinks" are available for easily pursuing subjects related to the current information. These hyperlinks do not exist in the printed manuals.
- The printed manuals contain comprehensive indexes to help you quickly find the specific information that you may be looking for. These indexes do not exist online.
- The PDF copies of the printed manuals can be searched using typical PC-style searching commands.
- "Release Guides" for older versions and releases of z/XDC are available online via the "HELP WHATSNEW" command.

A Roadmap

The structure of this manual follows the structure of the Online Help database. A consequence of this is that the sequence of information in this book, over all, is decidedly non-sequential. For those of you who prefer to read a manual from beginning to end, please accept my apologies. However, please let me make some suggestions.

If you are an experienced z/XDC user, then start with the **z/XDC[®] z1.3 Release Guide**. This will tell you what's new in this release of z/XDC. Online, the Release Guide can be reached by typing HELP WHATSNEW. You can then use hyperlinks to pursue the specific information that is of interest to you.

For new users, turn to the **z/XDC[®] User Guide**, and examine its Table of Contents carefully. You will see that there are about two dozen major topics arranged alphabetically: Addressing, Attentions, Breakpoints, ..., Virtmem, XDCCALL. Information within topics is presented more or less sequentially. The following **User Guide** topics are of particular interest:

- Perhaps the first topic that should be read is named "**DEBUGGING**". This and its subtopics give comprehensive information about whether and to what extent you may have to modify your program in order to use z/XDC.
- Another topic that should be read early on is named "**XDCCALL**". XDCCALL is a utility program that can be used to start a debugging session for your program.
- If you plan to debug programs that run as batch jobs or system tasks, then read the "**CDF**" topic. "Cross Domain Facility" is the component of z/XDC that permits user terminals to connect to debugging sessions for background jobs.

For z/XDC command information, turn to the **z/XDC[®] Commands** manual. Start with the basic commands. The DISPLAY,

z/XDC[®] z1.3 RELEASE GUIDE

(Introduction)

FORMAT, and LIST commands display storage and important program related structures. The AT and TRAP commands set breakpoints. You can use the TRACE command to step execution through your program slowly. The ZAP command allows you to change storage and registers.

If you wish to play with z/XDC's terminal and user interfaces, read the "**FULLSCREEN**" section of the **User Guide**. Also, try the PROFILE command for displaying and changing a very large number of session parameters.

Generally, the best approach is to plan your reading using the Table of Contents. And of course, if you can't find the information that you are looking for, call us. There's no charge, and we will be glad to help! Our number is 800-XDC-5150 (USA: 928-771-2003). If the information that you want is in the book, we will explain what you want to know and tell you where to find complete information. If it is not, then we will add it for our next release.

z/XDC[®] z1.3 RELEASE GUIDE

Online Help Panels

Help Whatsnew

XDC's Change History: For detailed information, type S at the left, then press ENTER. The information presented will be the most useful to experienced XDC users who want a concise summary of what has changed and a road map of where to look for more specific information.

```
z/XDC    z1.3 - (04/04) Full support for Loading module maps of Program Objects
           (PM2 through PM4).

z/XDC    z1.2 - (10/03) Z/Architecture support (64-bit addressing, etc.)
XDC/SE S2.0 - (12/00) Incremental changes implemented via maintenance.
XDC/SE S2.0 - (08/00) New release: Source Level Debugging Support!
XDC/SE S1.0 - (11/98) New version! PDS/E support! XMS Support! Etc.
XDC     X3.3 - (10/97) Incremental fixes and additions
XDC     X3.2 - (12/96) Incremental fixes and additions
XDC     X3.1 - (04/95) Beta-test release of X3.2
XDC     X3.0 - (06/94) MVS/ESA support
```

Help Whatsnew Z13

z/XDC z1.3 is an incremental advance with respect to the prior release (z1.2). It includes, of course, all fixes and other maintenance changes posted for the prior release. The main additional change is comprehensive support for reading and understanding module maps for all program objects through at least PM4 (and beyond?). For more detailed information, please select the following subtopics.

- **AUTOCLONING:** z/XDC now provides a method for automatically creating an entire series of equates or dsect maps to represent either a chain of control blocks or a set of entries in a table. Select this topic for more information.
- **CASESENSITIVITY:** Select this topic for a discussion of improvements in z/XDC's support of case sensitive section names and other labels within module maps.
- **COMMANDS:** There are no new or deleted commands in z1.3, but some commands and **built-in functions** have been changed.
- **EQUATES:** Several new built-in equates have been defined. Select this topic for more information.
- **HELP:** Several changes of varying significance have been made to the Online Help. Select this topic for more information.
- **MAPDISPLAYS:** Some improvements have been made in the displays of mapped areas of storage.
- **PROGRAMOBJECTS:** Select this topic for details about the improvements that have been made in z/XDC's support for program objects.
- **SCRIPTS:** Some of the command scripts distributed with z/XDC have been changed. Select this topic for more information.
- **MISCELLANEOUS:** This describes other minor changes have been made (miscellaneous one liners and such).

z/XDC[®] z1.3 RELEASE GUIDE

Help Whatsnew Z13 Autocloning

Autocloning is a new method by which you can automatically generate an entire series of equates or dsects maps to represent either each entry in a table or each control block on a chain. The equates or dsects that are created all have unique sequence numbers but share a common root name that you must provide.

Chains

The EQUATE and USING commands have new operands that let you define the following about chains:

- Where the chain field is.
- How wide the chain field is (3, 4, or 8 bytes).
- A mask for selecting a chain field's significant bits.
- The chain field value that signifies end-of-chain (zero, usually).
- The maximum number of chain entries to label or map.

Tables

z/XDC's autocloning supports tables of contiguous entries containing either fixed length or variable length entries. When variable, the table entries will (presumably) have a length field. This length field can be of any reasonable width, and it may define either the table entry's entire length or only the length of the entry's variable portion (in which case, all entries in the table are presumed to consist partly of common fields of a fixed size). In support of all this, the EQUATE and USING commands have new operands that let you define the following about tables:

- For fixed length table entries, or for variable table entries containing a fixed length root section, you can specify the length either of the entries or of the root section.
- For variable length table entries, you can specify where the length field is and how wide it is (1 through 8 bytes).
- You can specify either the length field value (if any) that identifies the end of the table or the address at which the table ends.
- Finally, you can specify the maximum number of table entries to label or map.

The equates or dsect maps that are generated through autocloning can all have the same attributes and characteristics that any other equate/dsect might have. In particular, they can be assigned either fixed bases or **floating bases**, thus the equates/dsects can be created to represent either particular instances or generic instances of table or chain entries.

Autocloning is implemented via new operands for the EQUATE and USING commands. For more information, see:

- HELP COMMANDS EQUATE
- HELP COMMANDS USING

I would like to thank Sonny Brucelas of Candle Corporation for being the most recent person to remind me to implement this facility.

Help Whatsnew Z13 CAsesensitivity

z/XDC now makes a distinction between module maps containing all uppercase labels and those containing mixed case labels. The former are considered to be case insensitive, while the latter are, of course, case sensitive.

Maps (both dsect and csect maps) constructed from Assembler produced ADATA or SYM data are always case insensitive. This is regardless of the fact that Assembler labels can have mixed case. This is because the Assembler (unlike the Binder) considers all labels that differ only in case to be the same label.

z/XDC[®] z1.3 RELEASE GUIDE

(Help Whatsnew Z13 CAsesensitivity)

When a map is case insensitive, commands can reference its labels regardless both of the case used in the command, and of the case used in the label and of z/XDC's UPCASE/ASIS setting.

On the other hand, when a map is case sensitive, then:

- When SET UPCASE is in effect, only uppercase labels can be referenced.
- When SET ASIS is in effect, all labels can be referenced, but the case given in the referencing command must exactly match the label's case.

z/XDC's current case setting can be displayed by the LIST UPCASE command.

The displays produced by both the MAP command and the LIST MAPS command have been improved to show the case sensitivity of the maps being loaded or displayed. For more information, see:

HELP COMMANDS MAP
HELP COMMANDS LIST MAPS

Help Whatsnew Z13 COmmands

The following commands and built-in functions have been changed for z/XDC z1.3:

- DELETE MAPS** - This command now has limited support for wildcarding to permit the deletion of one or more series of similarly named maps.
- DMAP** - This command has improved support for case sensitive maps, for program object maps (loaded as dsects), and for maps of deferred load classes.
- EQUATE** - This command now has several new operands in support of the **autocloning** of equates. Also, other operands have been added or changed in support of setting the equate's **formatting bias**.
- FIND** - The HEXDATA operand is being replaced by the **DATA** operand.
- FORMAT** - New operands have been added to allow you to control this command's initial formatting bias (instruction display vs. data display). Also, there is a display improvement.
- ~INDIRECT (mask)** - This built-in function now supports using a mask to define the indirect operation to be performed.
- LIST LKEDMAP** - New operands have been added to support display filtering and address formats.
- LIST MAPS** - The displays produced by this command now identify those maps that are case sensitive.
- MAP** - This command has improved support for case sensitive maps and for program object maps.
- SET FORMAT** - New operands have been added to allow you to control the default for the initial formatting bias (instruction display vs. data display) of the various storage display formatting commands (FORMAT, WHERE, and SHOW).
- SHOW** - New operands have been added to allow you to control this command's formatting bias (instruction display vs. data display).
- USING** - This command now has several new operands in support of the autocloning of dsect maps.
- WHERE** - New operands have been added to allow you to control this command's initial formatting bias (instruction display vs. data display). Also, there is a display improvement.

z/XDC[®] z1.3 RELEASE GUIDE

(Help Whatsnew Z13 COmmands)

Help Whatsnew Z13 COmmands DEletemaps

The DELETE MAPS command now accepts operands containing map names trailed by asterisks. They allow you to use a single operand to delete all maps whose names start with a common character string. Asterisks can be used with module map names, csect map names, dsect map names, and any legal combination thereof. Examples:

```
DELETE MAPS name*
DELETE MAPS mname.cname*
DELETE MAPS mname*.cname
DELETE MAPS mname*.cname*
```

For more information, see HELP COMMANDS DELETE MAPS.

Help Whatsnew Z13 COmmands DMap

Several improvements have been made to both the MAP and DMAP commands regarding case sensitivity, program objects, deferred-load classes, fragmented control sections, and reporting.

Reporting:

The MAP and DMAP commands have always provided information about the kind of maps they were loading, where they were loading them from, and the kind of data from which the maps were being built. However, this information was presented in a complex and confusing manner, and it relied upon the user's ability to make inferences in order to understand what had happened.

MAP and DMAP still produce those same messages, but now they also produce a final report that tables simple and direct information about what maps were built, what their names are, and what kinds of maps they are. Try it and see what you get. For more information about the MAP and DMAP commands, see:

```
HELP COMMANDS DMAP
HELP COMMANDS MAP
```

Case Sensitivity:

The MAP and DMAP commands now make a distinction between maps that need to respect the case sensitivity of their labels and those that don't. This distinction affects how a map's labels are referenced by other commands. For more information, see HELP ADDRESSING SYMBOLIC MIXEDCASE.

In addition, when the DMAP or MAP command loads maps that require case sensitive handling, the command's final report will show which do and which don't.

Program Objects:

Both the MAP and DMAP command now fully and correctly understand how program object classes are assigned to storage extents. Also, the commands now understand and properly identify "parts". Also, they now assign the correct numeric names to private labels, private sections, and such.

Deferred Load Classes:

The **MAP** command will map only those classes that reside in storage at the time that the MAP command is issued. Thus, if deferred load classes reside in storage at the time that the MAP command is issued, then the resulting map will include those classes; otherwise, it will not.

z/XDC[®] z1.3 RELEASE GUIDE

(Help Whatsnew Z13 COmmands DMap)

So, if you've issued a MAP command prior to the loading of a deferred load class, and then your program causes the deferred class to be loaded, the map will not automatically include information about the newly loaded class. In order to add that information to the map, you will have to delete the map and then reissue the MAP command to build a new one.

When the **DMAP** command is used to build a dsect map of the program object:

- It will build maps of the program object without regard to whether or not the object currently resides in storage.
- It will build maps of **all** loadable classes in the program object without regard to whether or not they are deferred classes.
- It will build a separate dsect map for each storage extent in which the program object would reside if it were in storage.

Normally, when the DMAP command builds a dsect of a load module or program object, the name of the map will match the name of the module or object. However, if it has to build more than one dsect, it has to give them each unique names. It does this by constructing a name that includes the name of the program object appended by the sequence number of the storage extent to which the map would apply. Example:

MYPGM_X#2

Fragmented Control Sections:

When a section (a control section, for example) contributes text to more than one loadable class, one result is that the section becomes fragmented within the program object; that is, the section appears in multiple pieces scattered throughout the object. So, in order to give each fragment a unique name, the MAP/DMAP command has to construct names that consist of the section's name appended by the name of the class within which the fragment resides. Example:

MYCSECT_B_TEXT24

Help Whatsnew Z13 COmmands Equate

The EQUATE command and the USING command have both been changed to support **autocloning**, the automatic generation of a series of equates or dsect maps to represent multiple entries in a table or on a chain. The EQUATE/USING commands now provide several new operands that can be used for defining the shape of the table/chain entries.

If the new autocloning related operands are not used, then the EQUATE/USING command's behavior remains unchanged from z/XDC's prior release.

Chains

The EQUATE and USING commands have new operands that let you define the following about chains:

- Where the chain field is.
- How wide the chain field is (3, 4, or 8 bytes).
- A mask for selecting a chain field's significant bits.
- The chain field value that signifies end-of-chain (zero, usually).
- The maximum number of chain entries to label or map.

Tables

z/XDC's autocloning supports tables of contiguous entries containing either fixed length or variable length entries. When variable, the table entries will (presumably) have a length field. This length field can be of any reasonable width, and it may define either the table entry's entire length or only the length of the entry's variable portion (in which case, all entries in the table are presumed to consist partly of common fields of a fixed size). In support of all this, the

z/XDC[®] z1.3 RELEASE GUIDE

(Help Whatsnew Z13 Commands Equate)

EQUATE and USING commands have new operands that let you define the following about tables:

- For fixed length table entries, or for variable table entries containing a fixed length root section, you can specify the length either of the entries or of the root section.
- For variable length table entries, you can specify where the length field is and how wide it is (1 through 8 bytes).
- You can specify either the length field value (if any) that identifies the end of the table or the address at which the table ends.
- Finally, you can specify the maximum number of table entries to label or map.

The equates or dsects that are generated through autocloning can all have the same attributes and characteristics that any other equate/dsect might have. In particular, they can be assigned either fixed bases or **floating bases**, thus the equates/dsects can be created to represent either particular instances or generic instances of table or chain entries.

For more information, see:

- HELP COMMANDS EQUATE
- HELP COMMANDS USING

The HEXDATA Operand

For the EQUATE command, the **HEXDATA** operand is being replaced by the **DATA** operand. HEXDATA will remain as an alias for DATA for the time being, but eventually it may be removed in a future release.

This is being done to maintain consistency with new operands on the **FORMAT**, **SET FORMAT**, **SHOW**, and **WHERE** commands, and to avoid conflicts with older operands accepted by those commands.

The NOBIAS Operand

INSTRUCTION, **DATA**, and now **NOBIAS** are a family of operands that related to whether or not an equate being created by the **EQUATE** command will affect the formatting bias of storage displays. See HELP COMMANDS EQUATE for more information.

Help Whatsnew Z13 Commands Find

The FIND command's **HEXDATA** operand is being replaced by the **DATA** operand. HEXDATA will remain as an alias for DATA for the time being, but eventually it may be removed in a future release.

This is being done to maintain consistency with new operands on the **FORMAT**, **SET FORMAT**, **SHOW**, and **WHERE** commands, and to avoid conflicts with older operands accepted by those commands.

Help Whatsnew Z13 Commands Format

The **FORMAT**, **WHERE**, and **SHOW** commands generally attempt to format storage contents as instructions, but this bias can be influenced by many factors, such as whether or not a PSW points to the storage being formatted, as well as the attributes of maps, fields, and equates that label the storage being displayed.

As a part of the management of this process, these commands maintain a concept called the **formatting bias** which, for a given display line, contributes information about how the preceding display line was formatted. In other words, if a display line was formatted as data, then the display formatter will be biased towards formatting the following display line also as data, unless new information (such as

z/XDC[®] z1.3 RELEASE GUIDE

(Help Whatsnew Z13 COmmands FOrmat)

label attributes) suggests otherwise.

New operands have been added to several commands to provide you with the ability to control the command's **initial** formatting bias. Those commands are:

SET FORMAT
FORMAT
SHOW
WHERE

The new operands are **INSTRUCTION**, **DATA**, and **NOBIAS**. For more information, see the Online Help topics for each of the affected commands.

A Display Improvement

The storage protect key will now be displayed for the storage being formatted by the **FORMAT** and **WHERE** commands, so it will no longer be necessary to use the **DISPLAY** command to obtain this information. For more information, see:

HELP COMMANDS FORMAT
HELP COMMANDS WHERE

Help Whatsnew Z13 COmmands Indirect

The **~INDIRECT(type)** built-in function can be used in place of an indirect operator (% ? !) to perform an indirect operation within an address expression ("load the address of a new location from the location computer so far").

Type can now be a mask that defines the indirect operation to be performed. The width of the mask defines the width of the pointer to be loaded from storage. The bits in the mask identify which bits in the pointer field constitute the pointer, and which bits are to be zeroed. For more detailed information, see **HELP FUNCTIONS INDIRECT**.

Help Whatsnew Z13 COmmands LISTLked

The **LIST LKEDMAP** command has been improved in several ways.

- Several operands have been added so that the resulting display can be filtered to show only a subset of a program object's or load module's map. Those operands are:
 - **EXTENTS**: Shows only information about the storage in which the program object (or load module) resides (pretty much the same information that is shown by default for unmapped modules/objects).
 - **CLASSES**: Shows the classes within a program object.
 - **CSECTS**: Shows the control sections within a program object or load module. (For program objects, classes also are shown.)
 - **FULL**: Shows everything there is to see (including all external symbols).
- The display produced by **LIST LKEDMAP** is now sensitive to the current **ADDRESS/OFFSETS** setting as established by the **SET FORMAT** command. This affects the way in which the addresses of class, section, and external symbols are displayed: either as virtual addresses or as extent/class-relative offsets. In addition, **LIST LKEDMAP** has two new operands (**ADDRESSES** and **OFFSETS**) that can be used for immediately overriding the **SET FORMAT** defaults, if desired.
- The display has been rearranged so that longer names can be displayed. Previously, the display of long names was truncated at 20 characters or so. Now, up to 63 characters can be displayed.

z/XDC[®] z1.3 RELEASE GUIDE

(Help Whatsnew Z13 COmmands LISTLked)

- When program object's classes are displayed, the class's bind flags (Vn_BNL_BIND_FLAGS) and load flags (Vn_BNL_LOAD_FLAGS) are displayed and interpreted.
- When program object's sections are displayed, the section's type is now shown.

For more information, see HELP COMMANDS LIST LKEDMAP.

Help Whatsnew Z13 COmmands LISTMap

The display generated by this command has been changed to annotate those maps that contain case sensitive labels. For more information, see HELP WHATSNEW Z13 CASESENSITIVITY.

Help Whatsnew Z13 COmmands Map

Several improvements have been made to both the MAP and DMAP commands regarding case sensitivity, program objects, deferred-load classes, fragmented control sections, and reporting.

Reporting:

The MAP and DMAP commands have always provided information about the kind of maps they were loading, where they were loading them from, and the kind of data from which the maps were being built. However, this information was presented in a complex and confusing manner, and it relied upon the user's ability to make inferences in order to understand what had happened.

MAP and DMAP still produce those same messages, but now they also produce a final report that tables simple and direct information about what maps were built, what their names are, and what kinds of maps they are. Try it and see what you get. For more information about the MAP and DMAP commands, see:

HELP COMMANDS DMAP
HELP COMMANDS MAP

Case Sensitivity:

The MAP and DMAP commands now make a distinction between maps that need to respect the case sensitivity of their labels and those that don't. This distinction affects how a map's labels are referenced by other commands. For more information, see HELP ADDRESSING SYMBOLIC MIXEDCASE.

In addition, when the DMAP or MAP command loads maps that require case sensitive handling, the command's final report will show which do and which don't.

Program Objects:

Both the MAP and DMAP command now fully and correctly understand how program object classes are assigned to storage extents. Also, the commands now understand and properly identify "parts". Also, they now assign the correct numeric names to private labels, private sections, and such.

Deferred Load Classes:

The **MAP** command will map only those classes that reside in storage at the time that the MAP command is issued. Thus, if deferred load classes reside in storage at the time that the MAP command is issued, then the resulting map will include those

z/XDC[®] z1.3 RELEASE GUIDE

(Help Whatsnew Z13 COmmands Map)

classes; otherwise, it will not.

So, if you've issued a MAP command prior to the loading of a deferred load class, and then your program causes the deferred class to be loaded, the map will not automatically include information about the newly loaded class. In order to add that information to the map, you will have to delete the map and then reissue the MAP command to build a new one.

When the **DMAP** command is used to build a dsect map of the program object:

- It will build maps of the program object without regard to whether or not the object currently resides in storage.
- It will build maps of **all** loadable classes in the program object without regard to whether or not they are deferred classes.
- It will build a separate dsect map for each storage extent in which the program object would reside if it were in storage.

Normally, when the DMAP command builds a dsect of a load module or program object, the name of the map will match the name of the module or object. However, if it has to build more than one dsect, it has to give them each unique names. It does this by constructing a name that includes the name of the program object appended by the sequence number of the storage extent to which the map would apply. Example:

MYPGM_X#2

Fragmented Control Sections:

When a section (a control section, for example) contributes text to more than one loadable class, one result is that the section becomes fragmented within the program object; that is, the section appears in multiple pieces scattered throughout the object. So, in order to give each fragment a unique name, the MAP/DMAP command has to construct names that consist of the section's name appended by the name of the class within which the fragment resides. Example:

MYCSECT_B_TEXT24

Help Whatsnew Z13 COmmands SEtformat

The **FORMAT**, **WHERE**, and **SHOW** commands generally attempt to format storage contents as instructions, but this bias can be influenced by many factors, such as whether or not a PSW points to the storage being formatted, as well as the attributes of maps, fields, and equates that label the storage being displayed.

As a part of the management of this process, these commands maintain a concept called the **formatting bias** which, for a given display line, contributes information about how the preceding display line was formatted. In other words, if a display line was formatted as data, then the display formatter will be biased towards formatting the following display line also as data, unless new information (such as label attributes) suggests otherwise.

New operands have been added to several commands to provide you with the ability to control the command's **initial** formatting bias. Those commands are:

SET FORMAT
FORMAT
SHOW
WHERE

The new operands are **INSTRUCTION**, **DATA**, and **NOBIAS**. For more information, see the Online Help topics for each of the affected commands.

A Display Improvement

The storage protect key will now be displayed for the storage being formatted by the **FORMAT** and **WHERE** commands, so it will no longer be necessary to use the DISPLAY command to obtain this information. For more information, see:

z/XDC[®] z1.3 RELEASE GUIDE

(Help Whatsnew Z13 COmmands SEtformat)

HELP COMMANDS FORMAT
HELP COMMANDS WHERE

Help Whatsnew Z13 COmmands SHow

The **FORMAT**, **WHERE**, and **SHOW** commands generally attempt to format storage contents as instructions, but this bias can be influenced by many factors, such as whether or not a PSW points to the storage being formatted, as well as the attributes of maps, fields, and equates that label the storage being displayed.

As a part of the management of this process, these commands maintain a concept called the **formatting bias** which, for a given display line, contributes information about how the preceding display line was formatted. In other words, if a display line was formatted as data, then the display formatter will be biased towards formatting the following display line also as data, unless new information (such as label attributes) suggests otherwise.

New operands have been added to several commands to provide you with the ability to control the command's **initial** formatting bias. Those commands are:

SET FORMAT
FORMAT
SHOW
WHERE

The new operands are **INSTRUCTION**, **DATA**, and **NOBIAS**. For more information, see the Online Help topics for each of the affected commands.

A Display Improvement

The storage protect key will now be displayed for the storage being formatted by the **FORMAT** and **WHERE** commands, so it will no longer be necessary to use the **DISPLAY** command to obtain this information. For more information, see:

HELP COMMANDS FORMAT
HELP COMMANDS WHERE

Help Whatsnew Z13 COmmands Using

The **EQUATE** command and the **USING** command have both been changed to support **autocloning**, the automatic generation of a series of equates or dsect maps to represent multiple entries in a table or on a chain. The **EQUATE/USING** commands now provide several new operands that can be used for defining the shape of the table/chain entries.

If the new autocloning related operands are not used, then the **EQUATE/USING** command's behavior remains unchanged from z/XDC's prior release.

Chains

The **EQUATE** and **USING** commands have new operands that let you define the following about chains:

- Where the chain field is.
- How wide the chain field is (3, 4, or 8 bytes).
- A mask for selecting a chain field's significant bits.
- The chain field value that signifies end-of-chain (zero, usually).
- The maximum number of chain entries to label or map.

Tables

z/XDC's autocloning supports tables of contiguous entries containing either fixed

z/XDC[®] z1.3 RELEASE GUIDE

(Help Whatsnew Z13 COmmands Using)

length or variable length entries. When variable, the table entries will (presumably) have a length field. This length field can be of any reasonable width, and it may define either the table entry's entire length or only the length of the entry's variable portion (in which case, all entries in the table are presumed to consist partly of common fields of a fixed size). In support of all this, the EQUATE and USING commands have new operands that let you define the following about tables:

- For fixed length table entries, or for variable table entries containing a fixed length root section, you can specify the length either of the entries or of the root section.
- For variable length table entries, you can specify where the length field is and how wide it is (1 through 8 bytes).
- You can specify either the length field value (if any) that identifies the end of the table or the address at which the table ends.
- Finally, you can specify the maximum number of table entries to label or map.

The equates or dsects that are generated through autocloning can all have the same attributes and characteristics that any other equate/dsect might have. In particular, they can be assigned either fixed bases or **floating bases**, thus the equates/dsects can be created to represent either particular instances or generic instances of table or chain entries.

For more information, see:

- HELP COMMANDS EQUATE
- HELP COMMANDS USING

The HEXDATA Operand

For the EQUATE command, the **HEXDATA** operand is being replaced by the **DATA** operand. HEXDATA will remain as an alias for DATA for the time being, but eventually it may be removed in a future release.

This is being done to maintain consistency with new operands on the **FORMAT**, **SET FORMAT**, **SHOW**, and **WHERE** commands, and to avoid conflicts with older operands accepted by those commands.

The NOBIAS Operand

INSTRUCTION, **DATA**, and now **NOBIAS** are a family of operands that related to whether or not an equate being created by the **EQUATE** command will affect the formatting bias of storage displays. See HELP COMMANDS EQUATE for more information.

Help Whatsnew Z13 COmmands Where

The **FORMAT**, **WHERE**, and **SHOW** commands generally attempt to format storage contents as instructions, but this bias can be influenced by many factors, such as whether or not a PSW points to the storage being formatted, as well as the attributes of maps, fields, and equates that label the storage being displayed.

As a part of the management of this process, these commands maintain a concept called the **formatting bias** which, for a given display line, contributes information about how the preceding display line was formatted. In other words, if a display line was formatted as data, then the display formatter will be biased towards formatting the following display line also as data, unless new information (such as label attributes) suggests otherwise.

New operands have been added to several commands to provide you with the ability to control the command's **initial** formatting bias. Those commands are:

SET FORMAT
FORMAT
SHOW

z/XDC[®] z1.3 RELEASE GUIDE

(Help Whatsnew Z13 Commands Where)

WHERE

The new operands are **INSTRUCTION**, **DATA**, and **NOBIAS**. For more information, see the Online Help topics for each of the affected commands.

A Display Improvement

The storage protect key will now be displayed for the storage being formatted by the **FORMAT** and **WHERE** commands, so it will no longer be necessary to use the **DISPLAY** command to obtain this information. For more information, see:

```
HELP COMMANDS FORMAT
HELP COMMANDS WHERE
```

Help Whatsnew Z13 Equates

Several new built-in equates have been defined, including a whole series for labeling the storage pointed to by registers. Briefly, the new equates are listed below. For complete information, see **HELP EQUATES BUILTIN**.

DEADZONE: This labels storage located above 2 gigabytes and below 4 gigabytes.

EFREGS: This labels z/XDC's internal save area for the user program's error level floating-point registers.

@Ern: This is a series of sixteen built-in equates that label the storage locations pointed to by the error level general registers. These equates do not appear unless the error level registers are different from the retry level registers.

GAGALAND: This labels **all** the storage located above 4 gigabytes.

@Rn: This is a series of sixteen built-in equates that label the storage locations pointed to by the retry level general registers.

The register target equates (@Rn and @Ern) should be particularly helpful. They will show up in storage displays that start within 4K of the locations pointed to by the registers. In other words, whenever storage is displayed that is addressable by a register, information about that register will be included in the display.

With the introduction of the register target equates, it will no longer be necessary to use the **REGPTRS** script to create these equates. **REGPTRS** is retained, however, because it is still useful for labeling the targets of register sets that may have been saved in request blocks, TCBs, or elsewhere.

Use of the new register target equates (instead of the **REGPTRS** script) will also improve z/XDC's performance since the resolution of built-in equates is vastly more efficient than floating equates.

Help Whatsnew Z13 Help

Of course changes have been made throughout the Online Help documenting the various command changes and additional capability being made available by this release; however, some changes have also been made to the Online Help specifically to improve its organization and to improve the information conveyed by various topics. These sorts of changes are described here.

SCRIPTS: All of the **HELP CMDLIBRARY ...** topics have been renamed to **HELP SCRIPTS ...**

EQUATES: The topic discussing built-in equates has been considerably enhanced and reorganized into a main topic (**HELP EQUATES BUILTIN**) and six subtopics.

z/XDC[®] z1.3 RELEASE GUIDE

(Help Whatsnew Z13 Help)

MAPS: The topics discussing maps (**HELP MAPS**) have been reorganized, enhanced, and made more comprehensive.

NOMENCLATURE CHANGE: Throughout the z/XDC product and its documentation, the terms "linkedit map" and "Binder map" have been replaced by "module map".

Help Whatsnew Z13 Mapdisplays

The displays of mapped storage have been improved in a couple of ways.

Assembly Offsets: When a control section has been assembled with a non-zero starting address, both csect maps and dsect maps that are built for that control section are now assigned a zero point that matches the assembly offset (instead of the start of the map). This change has the following consequences:

Csect Maps:

When such a csect has been established by the **SET QUALIFIER** command as being the default csect, then address expressions that start with **+** will now be computed relative to the control section's **assembly origin** not its actual origin.

Example: If a csect is located in storage at 0003D580, but at assembly time that csect's starting location was shown as being 00002080, then the command **FORMAT .+10** will display location 0003B510. That result is derived as follows:

```
- Csect's storage address:          0003D580
- Adjustment for csect's assembly time address:  -2080
- Adjustment from the .+10 address expression:    +10
                                         +-----
Result:          0003B510
```

Address expressions that reference such a csect's name explicitly will still resolve to the csect's actual origin, not it's assembly time origin.

When storage is displayed that is formatted under the control of such a map, the displayed location offsets will be relative to the map's assembly time origin, not it's actual starting point.

Dsect Maps:

When a control section having a non-zero assembler time origin is mapped via the **DMAP** command, that map will be assigned a zero point located before (not at) the start of the map. The distance of the zero point from the start will be equal to the value of the assembler time origin.

Address expressions that reference just the map's name (and not any symbol with the map) will resolve to the map's zero point, not its actual starting point.

To reference such a map's actual starting point, give the map's name **twice**.
Example: **FORMAT name.name**.

When storage is displayed that is formatted under the control of such a map, the displayed location offsets will be relative to the map's zero point, not it's actual starting point.

I would like to thank Don Muldoover of the Internal Revenue Service for suggestions leading to this change.

z/XDC[®] z1.3 RELEASE GUIDE

Help Whatsnew Z13 Programobjects

z/XDC's understanding of the internal structure of program objects has been considerably improved. Previously, there were circumstances under which z/XDC would not know or be able to guess how program object classes were arranged in storage, in which case z/XDC would issue message DBC508E (MAPPING ERROR ...) and then just give up. In recent years, program objects have gotten structurally more and more complex, especially those program objects containing C generated code. Consequently, z/XDC's shortcomings with its understanding of program objects were becoming more and more of a problem.

Now however, z/XDC's understanding of program objects has been improved such that the MAP command should no longer fail with the DBC508E message, just so long as the Operating System is OS/390 R2.10 or newer (i.e. any z/OS). (There is still a possibility of DBC508E being issued when the Operating System is OS/390 R2.9 or older.)

z/XDC's support for program object maps has limitations that are documented at HELP COMMANDS MAPS MODULEMAP.

In conjunction with the improved program object support, the LIST LKEDMAP command has been enhanced in several ways:

- New operands have been provided that allow the resulting display to be optionally filtered to produce only a subset of the display (just classes, for example).
- The addresses of objects within a program object can now be shown as offsets, if desired.
- The display has been rearranged so that long names can be displayed better. Previously, a max of only 20 characters were displayed. Now, up to 63 characters will be shown.
- For classes, bind flags (Vn_BNL_BIND_FLAGS) and load flags (Vn_BNL_LOAD_FLAGS) are displayed and interpreted.
- For sections, the section's type is now displayed.
- Support has been added for displaying "parts".

I would like to thank the following people for their interest in a solution to the program objects mapping problem:

Chris Bailey of Rocket Software
Chris Craddock of BMC Software
Rod Damron of Mobius Management Systems
Bob Green of BMC Software
Len Nerenburg of Mobius Management Systems
Andrew Rogers of Oracle
Bob Shannon of Rocket Software

Help Whatsnew Z13 Scripts

REGPTRS

The usefulness of the REGPTRS script has been largely replaced by the introduction of new built-in equates that label the storage targeted by the general registers. Nevertheless, this script has been retained because it is still useful for labeling the storage pointed to by registers that have been saved in request blocks, TCBs, and elsewhere.

In addition, the names of the equates created by the REGPTRS script have been changed from @Rn to RRn, and the RRn equates are now floated off an RREGS equate instead of @REGS. To cause the equates to label a different register set, just

z/XDC[®] z1.3 RELEASE GUIDE

(Help Whatsnew Z13 Scripts)

redefine the RREGS equate to label the save area for that set. For more information, see HELP SCRIPTS REGPTRS.

EREGPTRS

The usefulness of this script has been completely superseded by the new @ERn built-in equates. Consequently, this script has been removed from the z/XDC product.

Help Whatsnew Z13 Miscellaneous

The following miscellaneous changes have been made for release z1.3 of z/XDC:

RETRY AND ERROR LEVEL ENVIRONMENTS MISMATCH

When z/XDC receives control in a situation where the retry level and error level environments are **different**, a new, more emphatic, more comprehensive message is issued in an attempt to make the user aware that something unusual is happening and that he **must** stop what he is doing and take the time to figure out exactly what the situation is! This new message is DBC954W. See HELP MESSAGES DBC954 for more information.

MAPPING WHEN MULTIPLE COPIES OF A MODULE ARE PRESENT

Normally, when multiple instances of a load module are present in storage, a reference by name to that module will resolve to the first instance encountered in z/OS's standard search order. However, now when one of those instances has been mapped, by-name references to that module will resolve to the mapped instance, not necessarily the first instance. For more information, see HELP DEBUGGING TIPS DUPLICATES.

z/XDC[®] z1.3 RELEASE GUIDE

(Help Whatsnew Z13 Miscellaneous)

INDEX

Please note that this index is sorted according to the ASCII collating sequence, not EBCDIC. In particular, this means that digits sort in front of (not behind) alphabets, and that only some special characters sort in front of alphabets. Others sort behind alphabets.

The word processing program that is used here supports only two levels of index entries: main topics and sub-topics. When a sub-topic entry says "**see major topics**", this indicates that you should look for the same index entry among the main topics.

* (asterisk)	6
DELETE MAPS command\$	6
@ERn built-in equates	14
@Rn built-in equates	14
autocloning	4
chains	4, 7, 12
dsects	4
EQUATE command	7, 12
equates	4
floating dsects	4, 8, 13
floating equates	4, 8, 13
tables	4, 8, 13
USING command	7, 12
Binder map (see module map)	15
built-in equates	
@ERn	14
@Rn	14
DEADZONE	14
EFREGS	14
GAGALAND	14
built-in functions	
INDIRECT(...)\$	9
C support	
program objects	16
case sensitivity	
DMAP command!	6, 10
MAP command	6, 10
chains	
autocloning equates/dsects for)	4, 7, 12
change history	
z/XDC z1.3	3
changed commands	5
DELETE MAPS	6
DMAP	6, 10
EQUATE	7, 12
LIST LKEDMAP!	9
LIST MAP	10
MAP	6, 10
USING	7, 12
commands	
changed (see changed commands)+	5
DELETE	6
DELETE MAPS	6
DMAP	6, 10
EQUATE	7, 12
FORMAT	9, 11, 12, 14
LIST LKEDMAP	9
LIST MAP	10
MAP	6, 10
SET FORMAT	9, 11, 12, 14
SHOW	9, 11, 12, 14
USING	7, 12
WHERE	9, 11, 12, 14
DATA operand	
EQUATE command	8, 13

z/XDC[®] z1.3 RELEASE GUIDE

(Index)

FIND command	8
FORMAT command	<u>9</u> , <u>11</u> , <u>12</u> , <u>14</u>
SET FORMAT command#	<u>9</u> , <u>11</u> , <u>12</u> , <u>14</u>
SHOW command	<u>9</u> , <u>11</u> , <u>12</u> , <u>14</u>
WHERE command	<u>9</u> , <u>11</u> , <u>12</u> , <u>14</u>
DBC954W message	
error level environment+	17
retry level environment+	17
DEADZONE built-in equate	14
DELETE commands	
MAP operand	6
DELETE MAPS command	
wildcarding#	6
DMAP command	
case sensitivity!	6, 10
dsects	
autocloning	4
e-mail (see internet)	ii
EFREGS built-in equate	14
EQUATE command	
autocloning	7, 12
DATA operand	8, 13
formatting bias"	8, 13
HEXDATA operand"	8, 13
NOBIAS operand!	8, 13
equates	
autocloning	4
RREGS	17
RRn	17
EREGPTRS script (deleted)	17
error level environment	
DBC954W message+	17
FIND command	
DATA operand	8
HEXDATA operand	8
FORMAT command	
DATA operand	<u>9</u> , <u>11</u> , <u>12</u> , <u>14</u>
formatting bias"	<u>9</u> , <u>11</u> , <u>12</u> , <u>14</u>
INSTRUCTION operand&	<u>9</u> , <u>11</u> , <u>12</u> , <u>14</u>
NOBIAS operand!	<u>9</u> , <u>11</u> , <u>12</u> , <u>14</u>
formatting bias	9, 11-13
EQUATE command"	8, 13
FORMAT command"	<u>9</u> , <u>11</u> , <u>12</u> , <u>14</u>
SET FORMAT command&	<u>9</u> , <u>11</u> , <u>12</u> , <u>14</u>
SHOW command	<u>9</u> , <u>11</u> , <u>12</u> , <u>14</u>
WHERE command!	<u>9</u> , <u>11</u> , <u>12</u> , <u>14</u>
FTP address (see internet)	ii
functions (see built-in functions)	
'	9
GAGALAND built-in equate	14
HEXDATA operand	
EQUATE command"	8, 13
FIND command	8
history	
z/XDC z1.3	3
home page (see internet)	ii
INDIRECT(...) built-in function	
mask(9
INSTRUCTION operand	
FORMAT command&	<u>9</u> , <u>11</u> , <u>12</u> , <u>14</u>
SET FORMAT command*	<u>9</u> , <u>11</u> , <u>12</u> , <u>14</u>
SHOW command\$	<u>9</u> , <u>11</u> , <u>12</u> , <u>14</u>
WHERE command%	<u>9</u> , <u>11</u> , <u>12</u> , <u>14</u>
internet	
e-mail address	ii
FTP address	ii

z/XDC[®] z1.3 RELEASE GUIDE

(Index)

home page	ii, <u>iii</u>
web address	<u>ii</u>
legal statements	
trademark notice%	<u>iii</u>
usage warning"	<u>ii</u>
linkedit map (see module map)	
"	<u>15</u>
LIST command	
MAP operand	<u>10</u>
LIST LKEDMAP command	<u>9</u>
LIST MAP command	<u>10</u>
load modules	
multiple copies	<u>17</u>
MAP command	
case sensitivity	<u>6</u> , <u>10</u>
MAP operand	
DELETE command	<u>6</u>
LIST command	<u>10</u>
mapping	
load module search order\$	<u>17</u>
maps	
program objects	<u>16</u>
NOBIAS operand	
EQUATE command!	<u>8</u> , <u>13</u>
FORMAT command!	<u>9</u> , <u>11</u> , <u>12</u> , <u>14</u>
SET FORMAT command%	<u>9</u> , <u>11</u> , <u>12</u> , <u>14</u>
SHOW command	<u>9</u> , <u>11</u> , <u>12</u> , <u>14</u>
WHERE command	<u>9</u> , <u>11</u> , <u>12</u> , <u>14</u>
operands	
DATA	<u>8</u> , <u>9</u> , <u>11-14</u>
HEXDATA	<u>8</u> , <u>13</u>
INSTRUCTION	<u>9</u> , <u>11</u> , <u>12</u> , <u>14</u>
MAP	<u>6</u> , <u>10</u>
NOBIAS	<u>8</u> , <u>9</u> , <u>11-14</u>
program objects	
C generated code\$	<u>16</u>
mapping	<u>16</u>
message DBC508E#	<u>16</u>
OS/390 R2.10	<u>16</u>
OS/390 R2.9 and older)	<u>16</u>
z/OS	<u>16</u>
REGPTRS script	<u>16</u>
RREGS equates	<u>17</u>
RRn equates	<u>17</u>
retry level environment	
DBC954W message+	<u>17</u>
RREGS equates	
REGPTRS script	<u>17</u>
RRn equates	
REGPTRS script	<u>17</u>
scripts	
EREGPTRS (deleted)	<u>17</u>
REGPTRS	<u>16</u>
search order	
load modules, multiple copies.	<u>17</u>
SET FORMAT command	
DATA operand#	<u>9</u> , <u>11</u> , <u>12</u> , <u>14</u>
formatting bias&	<u>9</u> , <u>11</u> , <u>12</u> , <u>14</u>
INSTRUCTION operand*	<u>9</u> , <u>11</u> , <u>12</u> , <u>14</u>
NOBIAS operand%	<u>9</u> , <u>11</u> , <u>12</u> , <u>14</u>
SHOW command	
DATA operand	<u>9</u> , <u>11</u> , <u>12</u> , <u>14</u>
formatting bias	<u>9</u> , <u>11</u> , <u>12</u> , <u>14</u>
INSTRUCTION operand\$	<u>9</u> , <u>11</u> , <u>12</u> , <u>14</u>
NOBIAS operand	<u>9</u> , <u>11</u> , <u>12</u> , <u>14</u>
tables	

z/XDC[®] z1.3 RELEASE GUIDE

(Index)

autoclone equates/dsects for)	4 , 8 , 13
trademarks notice	iii
usage warning	ii
USING command	
autoclone	7 , 12
web address (see internet)	ii
WHERE command	
DATA operand	9 , 11 , 12 , 14
formatting bias!	9 , 11 , 12 , 14
INSTRUCTION operand%	9 , 11 , 12 , 14
NOBIAS operand	9 , 11 , 12 , 14
wildcarding	
DELETE MAPS command#	6

z/XDC[®] z1.3 RELEASE GUIDE

z/XDC[®] z1.3 RELEASE GUIDE

z/XDC[®] z1.3 RELEASE GUIDE