z/XDC®
RELEASE GUIDE

z/XDC® Release z2.2
for z/OS

David B. Cole

z/XDC® is a member of the XDC® (Extended Debugging Controller®) family of products

Generated 06/01/2019, 1:45am

PREFACE

USAGE WARNING AND LIABILITY DISCLAIMER

z/XDC® and its documentation (collectively, "Product"), including copies thereof, are the property of ColeSoft Partners, Inc. ("Owner"). Use of the product is licensed from ColeSoft Marketing, Inc. ("Licensor").

The Product may be used only by those organizations that are licensed by Licensor for such use and only in the manner so licensed. The Product may not be published, reproduced, distributed, or made available to third parties for any purpose without the express written permission of Owner or Licensor. However, a reasonable number of copies may be made of the documentation (including the copyright notices thereon) as is necessary for the legitimate use of the Product within a licensed organization ("Customer").

Except as may be otherwise expressed in a signed agreement between Licensor and Customer, Owner and Licensor make 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, or 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 routines. 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 or Licensor 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, neither Owner nor Licensor shall be obligated to indemnify in any manner against any person or organization for any loss of any kind or nature which the person or organization may experience, arising out of the use or misuse of the Product.

CONTACTING COLESOFT

The **XDC®** family of products is marketed by **ColeSoft Marketing, Inc**. with its principal office in Charlottesville, Virginia. If you would like more information, please contact ColeSoft Marketing as follows:

Phone:	800-XDC-5150
	928-771-2003
FAX:	928-771-2005
E-Mail:	sales@colesoft.com

Our Technical Support contacts are:

Phone:	540-456-8210
E-Mail:	techsupt@colesoft.com
FTP site:	ftp.colesoft.com

Our Customer Services contacts are:

Phone: 540-456-8210
E-Mail: support@colesoft.com

Our snail mail address is:

Address: **ColeSoft Marketing, Inc.**
414 3rd Street NE
Charlottesville, Virginia 22902
USA

ONLINE PRESENCE

ColeSoft Marketing maintains the following resources on the Internet:

[Home Page] ColeSoft's Home Page is www.colesoft.com. It 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
- A dialog for downloading current maintenance for z/XDC
- Links permitting existing customers to download a full set of z/XDC's documentation
- Online product delivery
- 24x7 self-service for temporary, short-term, license activation codes for use in D.R. tests and other emergencies

[Facebook] ColeSoft's Facebook presence is at facebook.com/colesoftware. This is where we will from time to time post information about ColeSoft people and activities.

[LinkedIn] ColeSoft has a users group named z/XDC Users Group. It can be reached by xdc.com. This is the "Go-To" place for all things z/XDC. So if you want to see what's coming up with z/XDC, then join this group. Things that we put here include:

- Notices about new releases and what they include
- Notices of new maintenance and what has been fixed, changed or added
- Notices of new training videos as we create them
- Creative ways to solve situations that our customers might encounter
- Short "how to" tips illustrating now to use z/XDC and what it can do

But we want this group to be a two-way street. We would love it if our customers would post to the group such things as:

- Questions about how to do something with z/XDC
- Suggestions about how to improve z/XDC
- Interesting experiences customers have had using z/XDC
- New ways to use z/XDC that make you smile
- Problems encountered with z/XDC that you would like help with
- Pretty much anything having to do with z/XDC

[YouTube] ColeSoft's YouTube page is at youtube.com/colesoftware. This is where you will find several "how to" videos describing various aspects of using ColeSoft products. This is a wonderful resource, particularly for new users.

TRADEMARKS

TFS™, **XDC-TFS**™, **CDF**™, **XDC-CDF**™, **FASM**™, **base/XDC**™ and **asm/XDC**™ are trademarks of ColeSoft Partners, Inc.

ColeSoft®, **Extended Debugging Controller**®, **XDC**®, **z/XDC**® and **c/XDC**™ are registered trademarks of ColeSoft Partners, Inc.

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 are necessary for the legitimate use of z/XDC within their organization. Existing customers may download from our web site (www.colesoft.com/product-support/zxdc-support/zxdc-documentation) printable copies of all of z/XDC's manuals. Each manual is available in PDF format.

Contents

INTRODUCTION

ColeSoft has pursued the goal of making z/XDC's internal documentation as comprehensive as possible. Towards that end, we have devoted considerable effort to greatly expanding the amount of information available within z/XDC and to improving the accessibility of that information and the navigability of the Help Database as a whole.

This manual is nothing more than a printout of a section of the 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 Help Database has been segmented into four 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® z2.2 Release Guide
  Contains a history of all changes and upgrades made in the current release of z/XDC.

There are a couple of important structural differences between z/XDC's internal Help and these manuals:

- 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 only via the "HELP WHATSNEW" command.

A Roadmap

The structure of this manual follows the structure of the 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® z2.2 Release Guide**. This will tell you what's new in this release of z/XDC. Within z/XDC, the Release Guide can be reached by typing HELP WHATSNEW. You can then use crosslinks 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 "**XDCSRVER**" topic and its subtopic, "**CDF**". "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, 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.

ONLINE PRESENCE

ColeSoft Marketing maintains the following resources on the Internet:

[Home Page] ColeSoft's Home Page is www.colesoft.com. It 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
- A dialog for downloading current maintenance for z/XDC
- Links permitting existing customers to download a full set of z/XDC's documentation
- Online product delivery
- 24x7 self-service for temporary, short-term, license activation codes for use in D.R. tests and other emergencies

[Facebook] ColeSoft's Facebook presence is at facebook.com/colesoftware. This is where we will from time to time post information about ColeSoft people and activities.

[LinkedIn] ColeSoft has a users group named z/XDC Users Group. It can be reached via xdc.com. This is the "Go-To" place for all things z/XDC. So if you want to see what's coming up with z/XDC, then join this group. Things that we put here include:

- Notices about new releases and what they include

- Notices of new maintenance and what has been fixed, changed or added
- Notices of new training videos as we create them
- Creative ways to solve situations that our customers might encounter
- Short "how to" tips illustrating now to use z/XDC and what it can do

But we want this group to be a two-way street. We would love it if our customers would post to the group such things as:

- Questions about how to do something with z/XDC
- Suggestions about how to improve z/XDC
- Interesting experiences customers have had using z/XDC
- New ways to use z/XDC that make you smile
- Problems encountered with z/XDC that you would like help with
- Pretty much anything having to do with z/XDC

Built-in Help Panels

Help Whatsnew Z22

z/XDC release **z2.2** includes all maintenance fixes to release **z2.1** and the following additional changes. For detailed information, type an **H** at the left to select directly, or use **HELP *NEXT** to proceed sequentially. Use HELP *FORWARD to skip.

↕	**AUTOSTEPPING**	- (For c/XDC) c/XDC can now automatically skip past sections of code that perform prologue and linkage convention functions within XL C/C++ and Metal C programs.
↕	**BUILTINHELP**	- Significant new and changed Built-in Help topics are summarized here.
↕	**CICS**	- Chipping around the edges... z/XDC can now find load modules within a CICS address space.
↕	**COMMANDS**	- Several commands have been added or changed. A few may have been removed.
↕	**CXDC**	- z/XDC z2.2 now includes initial support for debugging at the source code level programs written XL C/C++ and Metal C.
↕	**DDNAMES**	- Support for two new DDNAMEs has been added. They are **//xxxNOMAP** and **//xxxNSTEP.**
↕	**EQUATES**	- (For c/XDC) there are new EQUATEs defined.
↕	**FRR**	- Improvements to support for running z/XDC as an FRR.
↕	**MAPPINGTHEUNMAPPABLE**	- There now is a way to make z/XDC aware of the existence and locations of Privately Loaded load modules and program objects.
↕	**POINTANDSHOOT**	- (For c/XDC) There are new Point-and-Shoot commands.
↕	**PROFILEMENUINGSYSTEM**	- Some new settings pertaining to c/XDC have been added to Session Profiles.
↕	**PROFILERESETSANDDEFAULTS**	- New Factory Default profiles have been added in support of c/XDC.
↕	**REXX**	- A couple of changes have been made regarding rexx/XDC.
↕	**SHORTCUTCOMMANDS**	- z/XDC has introduced four new shortcut commands for c/XDC, and has added new functionality to the existing **L** shortcut command.

STARTUPPANEL - A new field for controlling **AUTOSTEP'ing** has been added to the z/XDC Startup Panel.

TRAP2 - z/XDC can now run as a **Trap Handler,** and so breakpoints and trace points can now (optionally) be created as **TRAP2** instructions.

THINGSFIXED - Things that have been fixed, both at product release time and subsequently via maintenance.

INCOMPATIBILITIES - Those changes that are **incompatible** with prior releases are described here.

Help Whatsnew Z22 Autostepping

Autostepping causes execution to automatically step past Language Environment prologues. This allows the user to avoid dealing with those assembly-language routines that prepare the environment for their XL C/C++ and Metal C program, and to deal instead with the business of debugging the user program.

- This service can be turned on or off, by the **SET STEP AUTOSTEP= command.**
- Its current setting is displayed by the **LIST STEP** command.
- This setting is saved in your Session Profile by the **PROFILE SAVE command.**
- It can also be displayed, set and saved by the **Profile Menuing System.**

A new field has been added to the **z/XDC Startup Panel** that let's you ignore the Session Profile's **AUTOSTEP** setting and forces it off during the Debugging Session startup process. (This is prior to your being able to enter your first command.)

For more information, see the Autostepping discussion in HELP DEBUGGING C DEBUGGING.

Help Whatsnew Z22 Builtinhelp

New and Significantly Changed Topics

As with any new release, the Built-in Help has been extensively updated to document the changes in this release. Also, numerous corrections and clarifications have been made throughout the HELP. So if there has ever been anything that you though was either unclear or missing, go back and look again. And if it's **still** unclear or missing, please let us know. Thanks.

In addition, the following topics have either been added or extensively revised, so particular mention is appropriate.

HELP ADDRESSING NUMERIC SCALINGCODES

Most operands that accept a hex numeric value also accept a decimal value when the value is trailed by the letter **N.** Recently, that feature has been extended to accept **scaling codes** that permit you to express decimal numbers scaled by various powers of **2:**

K means 2**10 (kibibytes)
M means 2**20 (mebibytes)
G means 2**30 (gibibytes)
T means 2**40 (tebibytes)
P means 2**50 (pebibytes)
X means 2**60 (exbibytes)

Examples:
- **4K** means **4,096** (not 4,000)
- **16M** means **16,777,216** (not 16,000,000)

HELP BREAKPOINTS

This has been substantially rewritten to incorporate new information and concepts arising from the new capability of using **TRAP2** instructions for breakpoints.

HELP BREAKPOINTS STEPPING

"Tracing" is an old term that we used to use to describe stepping execution through an Assembler program. But now that we have added c/XDC to the product, we have come to realize that **stepping** is a better term for this process.

Accordingly, HELP BREAKPOINTS STEPPING is a new topic that introduces a discussion of stepping as it applies to both Assembler and XL C/C++ and Metal C programs.

HELP BREAKPOINTS STEPPING C

This is a new topic that provides general information about stepping execution though XL C/C++ and Metal C programs.

HELP BREAKPOINTS STEPPING ASSEMBLER

This used to be named HELP BREAKPOINTS TRACING. It provides general information about stepping execution through Assembler programs and machine code in general.

HELP BREAKPOINTS TRACING

This topic has been renamed to HELP BREAKPOINTS STEPPING ASSEMBLER.

HELP BREAKPOINTS TYPES

Introduced by update PEM-1805B, z/XDC now supports using **TRAP2** instructions as breakpoints instead of illegal opcodes. This is optional. The choice is made with the **TRAP2|ZERO** operands of the **SET TRACE** commands.

The main advantage of using TRAP2 instructions is improved performance during **conditional** tracing and trapping.

For complete information, see HELP BREAKPOINTS TYPES.

HELP COMMANDS EWHERE

When breakpoints are built using **TRAP2** instructions (i.e. when **SET TRACE TRAP2** is in effect), the **Retry** level environment and the **Error** level environment become one and the same. This has consequences regarding the EWHERE command, so the HELP COMMANDS EWHERE topic has been revised accordingly.

HELP COMMANDS MAP

Maintenance update Z22-1702H Reorganized this topic, breaking it up into 9 subtopics. Also, the overall information is (hopefully) improved.

HELP COMMANDS MAP PRIVATELYLOADED

Maintenance update Z22-1702H added a substantial amount of new information pertaining...

- To a change in z/XDC’s use of information conveyed by the MAP command’s first operand when (and only when) a **START=** operand is also given.
- To a technique for mapping USS loaded load modules.

HELP COMMANDS SYNTAX CHARACTERSTRINGS WILDCARDS

This topic has been deleted in favor of a replacement: HELP COMMANDS SYNTAX MASKS

HELP COMMANDS SYNTAX MASKS

More and more z/XDC commands are being updated to accept wildcard masks as operands, so this topic has been written to be a central place where the syntax of such operands is described.

HELP DEBUGGING C

This heads a new set of topics that describes our **c/XDC** Feature. Subtopics include:

- **SETUP** - Preparing for an XL C/C++ and Metal C debugging session.
- **STARTING** - Starting a debugging session either in TSO or in the batch.
- **LIBRARYLISTS** - Helping the **MAP** command find **DWARF files** and **source files.**
- **DEBUGGING** - Conducting an XL C/C++ and Metal C debugging session.
- **VARIABLES** - Understanding and managing C variable displays.
- **CXDCHOOK** - Compiling a static hook into an XL C/C++ and Metal C program.
- **CXDCIS** - Determining what z/XDC **clone** to debug with.
- **EXTERNAL ISSUES** - Known issues within c/XDC that are beyond our control.

HELP EXITS

This branch of the Built-in Help has been revisited and significantly revised. Hopefully, the discussions have been improved.

HELP EXITS REGSTACK

This topic describes a new **Register Stack Exit** that will be helpful to Metal C programmers who have replaced their Metal C prologs with logic that uses register saveareas having **non-standard** formats and linkages.

HELP HOOKS DYNAMIC OTHERSPACES STARTINGNEWSESSIONS

This is a new topic that focuses on using cross-address space Dynamic Hooks to start new debugging sessions in jobs that have not been prepared to use z/XDC.

HELP MAPS PRIVATELYLOADED USSFILES

This is a new topic added by maintenance update Z22-1702H. It discusses issues regarding mapping load modules that were loaded into storage by Unix System Services (USS).

HELP MESSAGES DBC514

The DBC514I messages comprise the **System Interface Initialization Report. It contains messages describing every element of z/XDC's System Interface, and it is displayed to SYSLOG every time z/XDC decides it needs to repair or replace one or more elements of the System Interface.**

This topic has a large number of **subtopics** that describe in detail each element of the Interface.

This topic, along with all of its subtopics, have been significantly rewritten both to describe new System Interface elements that have been added in recent years, and to provided more detailed information about the Interface.

HELP SUPPORT FEATURESANDCAPS

Information has been added to this topic regarding:
- Concurrent Access Permits (CAPs),
- What they are,
- What triggers their use,
- When they are released,
- And how to release them early.

This topic replaces the old HELP SUPPORT FEATURES topic.

HELP XDCCALL DDNAMES

This topic had been missing information about //TASKLIB, //STEPLIB, //XDCPROF and //ISPPROF. That information has been added.

Help Whatsnew Z22 CIcs

z/XDC now knows how to find load modules within a CICS address space. This means:
- CICS load module names can be used in address expressions.
- CICS load modules can now be mapped.
- CICS load module names will now appear in the header lines of storage displays created by the FORMAT, DISPLAY, WHERE and EWHERE commands.
- The **LIST PGMS** command can now be used to display modules loaded by CICS.

This does **not** mean that z/XDC integrates well into CICS. For now, it remains the case that:
- When user program execution stops at (for example) a breakpoint, CICS will hang, while z/XDC is in control.
- z/XDC still cannot use the same 3270 user terminal that CICS uses.

For now, z/XDC supports these CICS releases: 4.2 through 5.5. Absent customer needs, we are not planning to roll our support back into older CICS releases. We are, of course, planning to support newer releases as they become available.

For more information, see HELP DEBUGGING CICS.

Help Whatsnew Z22 COmmands

The following commands are either new to z/XDC z2.2, changed in z/XDC z2.2 or deleted from z/XDC z2.2.

AT
ATX
TRAP
ADEFERRED
TDEFERRED
These commands have a couple of new operands:

- **SAVE=NO:** When an automatic commands string or a conditional expression is given, this operand prevents that string or expression from being saved for use as a default for subsequent breakpointing commands.

- **REMOVAL=DISABLE**
- **REMOVAL=PURGE**

When a transient breakpoint is automatically removed, this operand forces that breakpoint either to be purged or just to be disabled.

DELETE PROXYTASKS
Update **Z21-1609A** added a **DELETE PROXYTASKS** command that can be used to delete **Formal Proxy Tasks** from any accessible address space.

GO NOWHERE
NOWHERE is a new operand on the GO command that resumes user program execution in such a way that it immediately bounces back to z/XDC without doing anything at all.

The effect of this command is that it gives the System's Recovery/Termination Manager (RTM) the opportunity to reschedule z/XDC's execution. See HELP COMMANDS GO NOWHERE for all the ramifications of that.

The primary reason for writing this command is it can be used to complete the process by which an external debugging session has used a **SET AUTH command to change this session's authority level from non-authorized to authorized.**

GO RELEASECAPS
RELEASECAPS is a new operand on the GO command that can be used to end a debugging session without also ending the program being debugged.

LIBRARYLISTS (alias: **LL)**
(For c/XDC) Normally, when c/XDC needs to find a DWARF data file, it gets the file name from the C program's program object (where the hardcoded DWARF file name has been saved by the Binder). But when the DWARF dataset either has been renamed or is

otherwise unavailable, c/XDC has to find a correction for that name.

The **LIBRARYLISTS** commands are used to create and manage lists of corrected file names for both **DWARF data** files and **C source code** files.

LIST CXDC
This is a new command that reports on the current usability of the c/XDC Licensed feature. It reports whether or not the Feature currently is usable, and if not, then why not.

LIST MEMORYOBJECTS
This command now displays information about an address space's above-the-bar MEMLIMIT: how large it is, how much of it has been used, and how much remains to be used. (This is in addition to the memory object information that it was already displaying.)

LIST PGMS
Three changes have been made to this command:

- LIST PGMS **CICS:** When the target address space is a CICS region, the CICS operand directs the command to display only modules loaded by CICS.

  This operand can be used either:
  - When z/XDC is running within a CICS region,
  - Or when Foreign Address Space Mode is being used to display storage within a CICS region.

- LIST PGMS **namemask:** When the first operand contains wildcard characters (* and/or ?), the LIST PGMS command searches all load module queues, and displays all those whose names are matched by the mask.

- LIST PGMS **ALLTASKS:** The ALL operand has been changed to ALLTASKS because this name is more descriptive of what the operand actually does: It displays all modules that are associated with tasks.

  If you really do want to display all modules present on all queues, LIST PGMS * will do the trick.

LIST STEP
(For c/XDC) This command displays c/XDC's settings for the default meaning of the **STEP** command.

LIST TIOT
A few improvements have been to this command:

- By default the command now displays entries sorted in ddname order.
- It now accepts a SORT= operand for sorting displays in DDNAME, VOLSER, DSNAME/PATH or unsorted order.
- It now scans the XTIOT entries (instead of the TIOT).
- When an XTIOT entry has a corresponding TIOT entry, the report shows that entry's offset.

For more information, see HELP COMMANDS LIST TIOT.

LIST VARIABLES
(For c/XDC) This command displays one or more variables arrays, structures, unions, etc. that are defined in a High Level Language program.

LIST VSETTINGS
(For c/XDC) This command displays settings and limitations pertaining to HLL variables.

LIST VSTACK
This command displays the stack of variable pools (also known as Language Environment Stack Frames) listed from the oldest to the newest.

MAP
(For c/XDC) This command now accepts new operands **(DWARFDATA** and DWARFDATA=) that allow you to request that the **MAP** command load maps built from DWARF data.

Maintenance Update Z22-1702H has changed the way that the MAP command's first operand is use by z/XDC when (and only when) a **START=** operand is also given.

OFF
This command has a new operand: **NOPURGE.** It causes the command to remove breakpoints by only disabling them (not purging them).

In other words, the NOPURGE operand causes the OFF command to behave like the the **SET BREAKPOINTS DISABLE** command.

There is also, of course, a PURGE operand that causes the OFF command to purge the breakpoints (just like it always has).

SET AUTH
This is a new command that can be used from an authorized debugging session to start the process of making another, non-authorized session authorized. This is done without making the program being debugged also authorized.

SET CXDC
This is a new command that either enables or disables the c/XDC Licensed feature. It doesn't license it, but if it is licensed, then this turns it on or off.

SET PROXYTASKS
Update **Z21-1609A** added a **SET PROXYTASKS** command that can be used to create Formal Proxy Tasks ATTACH'd to any suitable task running in any accessible address space.

SET STEP
(For c/XDC) This command allows you to change c/XDC's settings for the default action taken by the **STEP** command. This setting can be saved in your session profile.

SET TRACE
This command as new operands that allow you to control whether breakpoints are implemented via **TRAP2** instructions or via **X'00'** illegal opcodes.

- SET TRACE **TRAP2** causes z/XDC to use TRAP2 machine instructions when setting breakpoints.
- SET TRACE **ZERO** causes z/XDC to use X'00' illegal opcodes when setting breakpoints.

For complete details, see HELP BREAKPOINTS TYPES.

(Note, technically, these aren't actually new operands, but now they work!)

SET VDISPLAY
(For c/XDC) This command allows you to change c/XDC's settings for adjusting the appearance of the output created by the **LIST VARIABLES command. These settings can be saved in your Session Profile.**

SET VSETTINGS
(For c/XDC) This command allows you to change certain HLL variable controls and limitations. These settings can be saved in your session profile.

STEP
(For c/XDC) This command allows the current source program statement to be executed, and it stops execution prior to executing the next. Operands are provided that allow you step into, out of or over subroutines.

TRAP
This command has a couple of new operands:

- **SAVE=NO:** When an automatic commands string or a conditional expression is given, this operand prevents that string or expression from being saved for use as a default for subsequent breakpointing commands.

- **REMOVAL=DISABLE**
- **REMOVAL=PURGE**
  When a transient breakpoint is automatically removed, this operand forces that breakpoint either to be purged or just to be disabled.

Help Whatsnew Z22 CXdc

In this release of z/XDC, we are introducing **c/XDC:** Supporting Source Level Debugging for programs written in XL C/C++ and Metal C. c/XDC provides commands, displays and methodologies oriented towards the C programmer, yet it retains all of the Assembler oriented and z/OS oriented capabilities that have been built up over the decades. So the C programmer will see only displays that make sense to him, yet the multilingual C and Assembler programmer will be able to access the best of both worlds, source displays, object displays, system structures, etc.

A c/XDC User's Guide (separate from the Built-in Help) can be downloaded from the ColeSoft website.

For more information, see HELP DEBUGGING C, but in a nutshell, **c/XDC provides the following:**

- **Source Statement Displays** (of course): When **DWARF** data libraries and source program libraries are available, c/XDC will automatically detect when a program is written in XL C/C++ and Metal C and it will then automatically build a source image map of that program **(AUTOMAPPING), and it will automatically advance program execution to the start of the user code (AUTOSTEPPING).**

  So when you start debugging a C program, the first display you will see will be your program with execution already advanced to its first C statement.

- **Variables, Arrays, Structures, Unions, whatever:** Using information available from DWARF data, c/XDC has full knowledge of all of the C variable types, syntax and formats:
  - When displaying C variable data, their values are shown according to their types and formats.
  - When zapping C data, you can do so using either C formatted data or z/XDC's classic hex and string data formats.
  - c/XDC properly handles the displays of arrays, structures, arrays of structures, structures of arrays... whatever. c/XDC just does it!

  - c/XDC has full knowledge of both C variable pools and LE stacks (when present):
    - Global constants, subroutine variables, and local variables can always be displayed (and zapped).
    - Similarly, variables in pools located in any older Language Environment Stack Frame also can be displayed/zapped.

- **Slices:** This support allows you to tell the **LIST VARIABLES** command to display only portions of an array instead the whole thing. You can display:
  - Selections of consecutive elements for one dimensional arrays,
  - Selected rectangles for two dimensional arrays,
  - Selected cubes for three dimensional arrays (etc.),
  - And discontiguous collections of any of the above.

  To display a **collection** of elements, separate the indices by commas. Example: **LIST VAR intArray[1,5,8]** displays array elements #1, 5 and 8.

  To display a **range** of elements, use a colon to define the range. Example: **LIST VAR intArray[2:4]** displays the third, fourth and fifth elements of the array. (In C, array indices are 0-origined.)

  The two syntaxes can be combined. Example: **LIST VAR intArray[1,2:4,5,8]**

  To display a rectangular slice of a two dimensional array: **LIST VAR intArray[1:4][5:7]**

- **Parsers:** The syntax rules for variables used in C differ and conflict with the rules for Assembler. Accordingly, z/XDC now supports multiple Language Parsers, one for each supported programming language.

  z/XDC also now has **Parser Management.** This allows you, the user, to control:
  - Which Language Parsers (ASM or CEE [or future]) are called for parsing a variable name,
  - The order in which the Language Parsers are called (ASM first or CEE first, etc.),
  - And which Language Parser generates the error message should they all fail.

  The Parsing Order can be displayed, set globally and saved in your session profile. It can also be overridden for individual commands.

- **New automatic commands syntax is now REQUIRED:** This is applicable to existing z/XDC customers. The old colon delimited syntax (for appending automatic command strings to AT and TRAP commands) is incompatible with use of c/XDC. According, if you wish to use c/XDC, then you **must** insure that **SET TRACE QUOTEONLY** is in effect for all your saved profiles. See HELP WHATSNEW Z22 INCOMPATIBILITIES for more information.

- **Where Are You?** Like always, the **WHERE** command can be used to display your program's current execution location. The resulting display will be formatted according to whatever maps have been loaded. If the map is a C map, then you will see C source statements. Otherwise, you will see Assembler or machine code.

- **Breakpoints:** For setting breakpoints, just use the **TRAP** and **AT** commands the same as you would for any other program.

- **Stepping Through Your Code:** There is a new command for stepping through your code at the language statement level. It's called **STEP,** and it can be used:
  - To step from one statement to the next,
  - To step into a subroutine,
  - To step out of a subroutine.
  - To step over a subroutine (i.e. let the subroutine run and recapture execution upon return).

  For more information, see HELP COMMANDS STEP.

- **Inserting Hooks:** If you wish to insert a dynamic hook into your C program, just use the **HOOK** command just as you would for an Assembler program.

  If you wish to compile a **static** hook into your C program (like what the #XDCHOOK macro does for assembler programs), your code needs to call a subroutine that we provide named **CXDCHOOK.** See HELP DEBUGGING C CXDCHOOK for more information.

- **The Underlying Machine Code:** Of course, even though c/XDC allows you to display code as if it were High Level Language statements, what's really in storage are machine instructions. So if you are also an Assembler programmer, you can see the underlying instructions simply by using the **BOTH** operand on the **FORMAT** command, or by using the **SET FORMAT** command to set **BOTH** globally.

- **The TRACE command:** You can still use the **TRACE** command to step through execution at the machine instruction level. In fact, **TRACE** commands and STEP commands can be freely intermixed.

- **Library Lists:** c/XDC provides a new **LIBRARYLISTS** command that enables you to control where c/XDC looks for DWARF data and C source code. It can be used either when DWARF/SOURCE data location information is incorrect in the program object (perhaps because you've moved it) or when the location information is simply unavailable (as in the METAL C case).

- **New Factory Default Profiles:** z/XDC now provides multiple Factory Default profiles, some suitable for assembler debugging and some suitable for **C** debugging. You use the **PROFILE RESET name** command to select which profile you want to load. For more information, see HELP COMMANDS PROFILE RESET.

- **Known bugs:** There are several known issues within c/XDC that may affect its operation. See HELP DEBUGGING C EXTERNALISSUES for more information.

Once Licensed, the c/XDC Licensed Feature can be turned on and off by the SET CXDC command.

The current state of the c/XDC Feature (licensing, availability and usability) can be displayed by the **LIST CXDC** command.

c/XDC has been developed by **Michael Lewis.** He is a top level developer with decades of experience. He is, in fact, one of the original developers of z/XDC itself. His prior work includes:
- Fullscreen Terminal Support
- Cross Domain Facility
- Session Profile Support
- Session Logging

Here is a partial list of c/XDC topics that are very useful to C programmers:

HELP COMMANDS LIST VARIABLES
HELP COMMANDS LIST VSTACK
HELP COMMANDS STEP
HELP DEBUGGING C

Here is a partial list of base z/XDC topics that also are very useful to C programmers:

HELP COMMANDS AT
HELP COMMANDS FORMAT
HELP COMMANDS KEYS
HELP COMMANDS PROFILE RESET
HELP COMMANDS TRAP
HELP COMMANDS WHERE

Help Whatsnew Z22 Ddnames

A couple of new ddnames are now supported:

- **//xxxNOMAP DD DUMMY:** At Debugging Session start time, the presence of this DD

card causes AUTOMAP'ing to be suppressed even when AUTOMAP=YES is set in the Session Profile being loaded.

- **//xxxNSTEP DD DUMMY:** At Debugging Session start time, the presence of this DD card causes AUTOSTEP'ing to be suppressed even when AUTOSTEP=YES is set in the Session Profile being loaded.

The purpose of these DD cards is to make it possible for you to affect these settings at the very start of your debugging session, prior to your being allowed to enter your first command.

Support for the **//xxxNSTEP DD** allocation has been added to the **z/XDC Startup Panel** (in case you really do want to trace through initial prolog code after all).

Help Whatsnew Z22 Equates

Some new built-in and automatic equates have been implemented:

VS#n_GC.varname [Global Constants Pool]
VS#n_SV.varname [Subroutine Variables Pool]
VS#n_LV.varname [Local Variables Pool]
VS#n_LV.area#.varname [Block Variables Pool]

These equates are applicable to debugging LE compliant programs. They are (re)created by the **LIST VSTACK** command. For each Stack Frame level in which variable pools exist, these equates label the starts and lengths of those pools.

For more information, see **HELP EQUATES BUILTIN.**

Help Whatsnew Z22 Frr

Starting with update **Z21-1509H** to the prior release (z2.1), it no longer is necessary to pre-create Proxy Tasks when using z/XDC as an FRR. They now will be created dynamically when needed and reused when they preexist. See HELP DEBUGGING FRR for more information.

This makes it easier to debug programs that have not been started under the control of xxxCALLA.

It also removes an impediment to starting debugging sessions dynamically via cross address space **HOOK** commands.

Update **Z21-1609A** also added improvements to FRR-mode debugging support. It implemented two new commands that can be used to create and delete Formal Proxy

Tasks either in the current debugging session or in any accessible foreign address space. The commands are:

- **SET PROXYTASKS**
- **DELETE PROXYTASKS**

The update also included a rewrite of the **HELP DEBUGGING FRR** topic.

Help Whatsnew Z22 Mappingtheunmappable

When a load module or program object is **Privately Loaded** into storage, its existence is not recorded into system control blocks. This makes the presence of the module invisible to the system and to z/XDC. This used to mean that the module could not, for example...

- Show up in **LIST PGMS** displays,
- Show up in the header lines of **FORMAT** and **WHERE** commands,
- Be displayed by the **LIST LKEDMAP** command,

Now there is a way by which you, the user, can tell z/XDC where such modules reside. This then allows you to reference the module as fully and freely as you can any other module in the system.

In simple terms, when you tell z/XDC where a Privately Loaded module is, it creates a private LPDE to describe the module, and its CDE/LPDE search routines have been updated to find it, thereby making the module known to the rest of z/XDC.

There are two ways to tell z/XDC about a Privately Loaded module:

- One is through changes to the existing **DMAP** and **USING** commands.
- The other is through a new **START=address** operand to the **MAP** command.

The **LIST PGMS** command has a new **PRIVATE** operand that displays a report showing all known Privately Loaded load modules collectively.

You can find detailed information starting at HELP MAPS PRIVATELYLOADED.

Mapping USS Loaded modules

Maintenance update Z22-1702H added to z/XDC’s Privately Loaded module mapping support the capability of assigning module maps to the storage occupied by load modules loaded by USS. For details start with HELP MAPS PRIVATELYLOADED USSFILES.

Help Whatsnew Z22 POintandshoot

When a window is displaying registers or storage, the display frequently includes groups of 1-, 2-, 3-, and (most often) 4-byte long hexadecimal values. If such a group contains an "ad-con" (i.e. an address of a storage location), then that is considered to be a **pointer field.** If the location being pointed to is of further

interest to you, then you can display that location by tabbing the cursor over to the pointer field and then overtyping the first one or two digits with a "point-and-shoot command".

c/XDC introduces the following new point-and-shoot command in support of High Level Language programs written in XL C/C++ and Metal C:

V - LIST VARIABLES

The location being pointed to is examined to determine if it contains a High Level Language variable. If it does, the contents of that variable are displayed via z/XDC's **LIST VARIABLES** command.

The resolved pointer is considered to be either 24, 31, or 64 bits wide according either to the user program's current addressing mode or the AMODE override character (% ? or !) that you may type along with the **V command.**

The **V** pointandshoot command is, of course, in addition to numerous older similar commands. See HELP FULLSCREEN POINTANDSHOOT for more information.

Help Whatsnew Z22 PROFILEMenuingsystem

Several new c/XDC related settings have been added to the Profile Menuing System necessitating the creation of an entirely new panel and the addition of new fields to a few other panels. For a description of the new panel, see **HELP PROFILES MENU HLLOPTIONS.**

Help Whatsnew Z22 PROFILEResetsanddefaults

Two new Factory Default profiles have been created in support of c/XDC.

With the advent of **c/XDC,** it became clear that the default profile used for Assembler debugging simply was not appropriate for C debugging. So we've added two new default profiles, one for wide terminals and one for the classic 80-column terminals. For more information, see HELP PROFILES DEFAULTPROFILE.

Help Whatsnew Z22 Rexx

The following minor changes have been made regarding rexx/XDC's REXX support:

- The **RXTSTENV** sample exec has been changed to illustrate, not just the wrong way to issue z/XDC commands (via the XDC Environment), but also the right way to do so (via the XDCCMD() built-in function).

The command that **RXTSTENV** now issues is **LIST XDC** which reports, among other things, the name of the z/XDC clone within which the exec is running.

- Tweaks have been made throughout the Built-in Help regarding rexx/XDC to clarify the distinction between the **XDC Environment** (in the REXX sense), and the **rexx/XDC Interface.**
  - See **HELP REXX** where the term **rexx/XDC Interface** is defined.
  - See **HELP REXX ENVIRONMENT** where the term **XDC Environment** is defined.

Help Whatsnew Z22 SHortcutcommands

Most lines displayed at the terminal have underscores (_) or periods (.) showing at the left side of the screen. These are "shortcut input fields", and any of several 1-character shortcut commands can be entered there. Different display lines accept different sets of shortcut commands.

c/XDC introduces the following new shortcut commands in support of High Level Language programs written in XL C/C++ and Metal C:

`*` - ("EBCDIC") High Level Lauguage programs have the ability to specify the default codeset used to define character constants, as well as to override that default for an individual variable. This shortcut causes c/XDC to display the affected variable using the EBCDIC codeset.

`|` - ("ASCII") This shortcut causes c/XDC to display the affected variable using the ASCII codeset.

`\` - ("string") C character strings can be viewed by one of two paradigms:
  - Either as a fixed length array of individual characters and other byte values,
  - Or as a variable length string of characters terminated by an end-of-string null character **(X'00').**

This \ shortcut command instructs c/XDC to display the affected variable as a variable-length, null-terminated, string.

`/` - ("array") This shortcut command directs c/XDC to display the affected variable as a fixed length display of characters and other byte values.

L - ("list") c/XDC also introduces new functionality to the

existing **L** shortcut command. It can be used within the displays produced by the **LIST VSTACK** and **LIST VARIABLES** command, but its behavior varies a bit depending whether it is used within the Working Window or a Watch Window:

- For the Working Window, an **L** just causes the display to be replaced with a display of the array's or structure's elements.

- For Watch Windows, an **L** is a toggle causing the existing display to expand or collapse to reveal or hide the array's or structure's elements.

Help Whatsnew Z22 STartuppanel

A new field for controlling **AUTOSTEP'ing** has been added to the **z/XDC Startup Panel.** See HELP WHATSNEW Z22 AUTOSTEPPING for more information.

Help Whatsnew Z22 TRap2

z/XDC Can now run as a **Trap Handler!** This means that breakpoints and trace points can now use **TRAP2** (X'01FF') instructions (rather then s0C1 abends) to cause the interrupts necessary to pass control to z/XDC. This has a couple of consequences:

- For one thing, the System overhead for processing abends is avoided. There is no program check. The System's Recovery/Termination Manager (RTM) does not get involved. No search occurs for the right Recovery Routine.

  Instead, with TRAP2 instructions, the hardware simply passes control directly to the Trap Handler routine (z/XDC) as a part of the instruction's normal processing. Done.

- For another, the distinction between error level and retry level simply goes away.

For complete details, see **HELP BREAKPOINTS TYPES.**

The use of TRAP2 instructions for breakpoints and tracing is controlled by the **SET TRACE** command. See HELP COMMANDS SET TRACE for details.

The installation of Trap Handler routines is performed **automatically** by z/XDC as the need arises. It occurs when z/XDC detects an **s0D3** abend caused by the absence of a Trap Handler when needed.

While as wonderful as TRAP2 instructions are, they **do not eliminate** the need for

z/XDC also to run as an Abend Recovery Routine pretty much like it always has. This is needed for two reasons:

- When an **actual abend occurs,** z/XDC needs to be there to capture it.
- When a TRAP2 instruction is encountered before a Trap Handler has yet been installed, z/XDC needs to be there to capture the resulting s0D3 abend so that it can **dynamically install** the missing Trap Handler.

Help Whatsnew Z22 THingsfixed

All maintenance from prior releases has, of course, been promoted into this release. In addition, the following issues from the prior release also have been corrected at product release time. (For things fixed by maintenance after product release, see HELP * MAINTENANCE.)

Issue with the NXINST() Built-in Function

This addressing function is supposed to resolve to the machine instruction that is to be executed next after the retry level environment's current instruction. Instead, it was resolving to the self same current instruction.

Post-Release Maintenance

To see what's been fixed since product release, see HELP * MAINTENANCE.

Help Whatsnew Z22 THingsfixed Maintenance

The following maintenance updates have been APPLY'd to z/XDC z2.2 since it was released. They are listed below, newest first. For more detailed information, some of the following topics are selectable.

DBC-1905H - Changes have been made to the Built-in Help generator to improve its management of hiliting controls. This has allowed us to correct several hiliting errors within the panels.

I have also taken the opportunity to change the formatting of all the HELP MESSAGES DBCnnn panels so as to display the doc'd messages in an improve and standardized manner. (Select at the left for an example.)

PEM-1905G - This update changes the way that z/XDC handles corruption of the TRAP save area when using TRAP2 style breakpoints.

Prior to this fix, z/XDC was very conservative when deciding that a corruption may have occurred. It assumed there may have been a corruption when many times there hadn't. Also, when a possible corruption was detected, z/XDCs TRAP handler would ABEND. This made it impossible to continue the debugging session.

With this fix applied, z/XDC only reports a corruption if one actually has occurred. The user is warned about the corruption, and can take steps to repair the corrupted items (the PSW, general registers, and AR15). Following the repair it is possible to resume the program being debugged.

CBC-1905F - This update corrects a parsing error in CBC-1905E. The **CMDS= and COMMANDS=** operands were not being recognized. (Fortunately, CMD= was being accepted, so all was not lost.) For more information, see HELP COMMANDS SET WINDOW CREATE.

CBC-1905E - This update adds a **CMDS=** operand to the **SET WINDOW CREATE command. When this command is used within a script, this new operand makes it possible for the script to define the content of the windows that it creates. For more information, see HELP COMMANDS SET WINDOW CREATE.**

We would like to thank Bob Berry of 21st Century Software for suggesting this enhancement.

DBC-1905D - This update makes a handful of changes:

- Mainly, this update implements a new built-in function named **PCLOCATE(pcnumber)** where the **pcnumber** is given as an address expression whose **resolved value** (after suitable trimming) is used as the PC number to be resolved. The result of the function is the location (address and aspace) of the PC routine that's connected to the given PC number. See HELP FUNCTIONS PCLOCATE for details.

- When z/XDC's disassembler displays a PC instruction, that display line will now accept a point-and-shoot command to display the PC routine that the instruction calls... **BUT note the warnings** in HELP FULLSCREEN POINTANDSHOOT MACHINEINSTRUCTIONS.

- The **HELP FULLSCREEN POINTANDSHOOT** topic has been revised, enlarged and restructured into multiple panels.

- A bug has been fixed in the **SYSINFO** script: The LIST XYZZY LMHT SORT=NAME command has a syntax error.

FHC-1905C - This update fixes an S0D7-24 abend that may pop in z/XDC when debugging space switching PC routines. The S0D7 arises due to AX=0 or a value that prevents the PT[I] instruction to be issued.

This fix will set AX=1 underneath the covers before issuing the PT/PTI instruction. Because setting the AX requires supervisor state, this fix will only attempt to set the AX when z/XDC is running in supervisor state.

We would like to thank Tom Marchant of Compuware for bringing this to our attention.

FHC-1905B - This update fixes a bug introduced by PEM-1812B that corrupted the SDWA if z/XDC is called conditionally. If z/XDC is called again non-conditionally, it can behave unexpectedly such as no longer recognizing breakpoints or getting stuck in real addressing mode.

We would like to thank Mitchel Dooley, Dick Nunke and Jay Cicardo of BMC for bringing this to our attention.

DBC-1905A - This update fixes an 0C4 failure that could happen in XDC31.DBCSUBS3. It would happen intermittently when displaying storage or issuing other module related commands while running in Foreign Address Space Mode.

DBC-1904D - This update makes a minor tweak (for clarity) to the **LIST LSTACK** command's report.

MDL-1904C - This update fixes support for the display of c++ reference variables within c/XDC

We would like to thank Adrian Smart of VISA for bringing this to our attention.

FHC-1904B - Another tweak to the ISPF checks introduced in DBC-1903C.

We would like to thank Lou D'Agnolo of Innovation Data Processing for bringing this to our attention.

MDL-1904A - This update fixes the addresses assigned to C language functions when c/XDC is mapping CSECTs compiled under IBM's System Programming C

Facility.

We would like to thank Uwe Kerstan of Beta Systems for bringing this to our attention.

FHC-1903E - A minor update to DBC-1903C to tolerate CA's PDSMAN product in ISPF's task structure.

We would like to thank Mikael Nystrom of SEB for bringing this to our attention.

MDL-1903D - This update fixes CSECT mapping support when c/XDC is running on z/OS 2.4.

We would like to thank Lee Paik of Compuware for bringing this to our attention.

DBC-1903C - z/XDC refused to use ISPF Display Services even though ISPF clearly was usable. Instead, z/XDC's Fullscreen TPUT Interface would be used, and z/XDC's SPLIT, SWAP and =jump commands would be unavailable.

z/XDC's logic for determining the availability of ISPF's Display Services was flawed.

This issue has recently impacted several customers. We would like to apologize for the problems they endured.

DBC-1903B - This update fixes a linkage stack corruption that occurred when an anticipated abend occurred within the process that generates and displays messages. The anticipated abend is recovered successfully, but a side effect of z/OS's abend recovery logic is that linkage stack entries can be lost, and z/XDC's logic did not anticipate that.

This led to another anticipated s0C1 being percolated instead of handled.

We would like to thank Bujji Reddy Regalla of BMC-India for bringing this to our attention.

MDL-1903A - This update adds support to c/XDC for the debugging of CSECTs created using IBM's System Programming c Facility (SPC).

We would like to thank Uwe Kerstan of Beta Systems for bringing this to our attention.

FHC-1902C - We saw some evidence of a possible storage overlay bug in HOOK processing and added a bit of logic in the DBC-1902B fix to generate diagnostics information. However, the newly added logic had a bug that caused HOOK processing to fail. This update fixes that bug.

DBC-1902B - This update continues the profiles support rework started in DBC-1902A, and it fixes several related bugs as well:

- One customer allocates //ISPPROF to **spool**! This caused several problems:
  - z/XDC fell into a loop generating **thousands** of **IEC130I XXXPROF DD STATEMENT MISSING** messages.
  - During z/XDC's attempt to open //ISPPROF, OPEN abended with an **IEC141I 013-A4** message. z/XDC had abend protection on, so OPEN terminated, but z/XDC did not. However, z/XDC mishandled the recovery badly!
  - OPEN's **SYSZTIOT** enq was retained even after OPEN terminated. This blocked all subsequent attempts to open or close anything by anyone in the same aspace.
- This update also cleans up several edge cases with the new LIBRARY= support that were not being properly handled by the DBC-1902A update.
- The **LIST PROFILES** report has been tweaked:
  - It now more clearly shows how profile library member names are constructed from profile names and clone names.
  - It now shows which profile names the **XDC** alias name potentially references.
- The Built-in Help for the PROFILE READ command has been significantly reworked. (Similar reworks for the PROFILE SAVE and LIST PROFILES commands will occur in a future update.)
- For more information, see:
  HELP COMMANDS PROFILE READ
  HELP COMMANDS PROFILE SAVE
  HELP COMMANDS LIST PROFILES

DBC-1902A - The major change implemented by this update is the addition of a **LIBRARY=** operand to the **PROFILE READ** and **SAVE** commands. This will be particularly helpful to those who want to use Session Profiles while debugging batch jobs.

This update also includes several lesser changes you may also want to know about. For detailed information, see HELP * DBC-1902A.

I would like to thank Dave Stedman of BankNet for discussions leading to this update.

DBC-1901G - Clarifications have been added to:

- HELP DDNAMES REFR8
- HELP DDNAMES RENT8
- HELP DEBUGGING REENTRANT
- HELP DEBUGGING REFRESHABLE

To make it clear that these ddnames, if used, should be used only for debugging non-authorized programs (via the xxxCALL utility), **not** authorized programs being debugged via the xxxCALLA utility.

I would like to thank Stefan Lehnert of Beta Systems for bringing this issue to our attention.

CBC-1901F - When z/XDC is invoked in **exotic** execution environments, numerous checks are made to see whether or not it can, in fact, run at all. If any of these checks fail, then z/XDC will abort. But before aborting, it will attempt to issue message **DBC902T to explain what the problem is.**

Normally, z/XDC will send this message to SYSLOG via **WTO; however, if it finds itself to be running within TSO, it will use TPUT** instead. Unfortunately, we had a bug that would cause the TPUT to fail with an **s15D** abend. This update fixes that bug.

Notes:

- **Exotic** execution environments include SRB routines as well as FRR protected task mode code.
- TPUT may (or may not) cause the message to be displayed at your terminal. If not, then you won't see it. But...
- TPUT may (or may not) also cause the message to be echoed to SYSLOG. If so, you may or may not see it depending upon whether or not you think to look for it there.

CBC-1901E - z/XDC's Built-in Help panels are contained in a load module named **xxxHELPM.** When a **HELP** command is issued and xxxHELPM has not yet been LOAD'd into storage, z/XDC will LOAD it.

Occasionally, situations may arise where xxxHELPM will be removed from storage outside of z/XDC's control. Normally, this does not present a problem because when z/XDC detects that this has happened, it just LOADs it back in again, and life goes on.

However, if the xxxHELPM-missing condition is detected on a HELP command

whose first operand is a **relative** panel reference, then an s0C4 failure would occur. This update fixes that.

FHC-1901D - This fix corrects a logic bug introduced by the PEM-1812D fix that prevented c/XDC from drilling down into a C/C++ variable. This bug also prevented the string shortcut commands from working in the LIST VARIABLE display.

CBC-1901C - This update extends z/XDC's CICS support to include **CICS R5.5. See HELP DEBUGGING CICS for a full statement of support.**

FHC-1901B - This update fixes a minor logic error in handling point-and-shoot for the BASR and BALR instructions.

FHC-1901A - This update improves z/XDC's performance when displaying mapped storage as object code. This update also changes what is displayed by z/XDC in this case.

One cool feature that z/XDC provides is the automatic resolution and display of an address pointer field to its target object.

Unfortunately, we've discovered that when FORMAT'ing SYMDATA mapped storage or ADATA mapped storage as OBJECT, there was a huge performance hit arising from all that address resolution processing.

When the **FORMAT** command (and the **WHERE** command too) is used to produce disassembled displays, z/XDC considers each 3-byte and 4-byte field as a potential address pointer, and it will attempt to determine if the field points to some object such as a load module, a CSECT or DSECT map, a z/XDC equate, etc. Since the address pointer can point to anywhere in storage, we have to rescan everything for each pointer. The processing time can take a few seconds to minutes depending how much mapped storage is being FORMAT'd.

While this performance hit is dependent on the size of storage being displaying, it is such a large performance hit that we felt it is necessary to prevent the address resolution from occurring in this specific case.

With this update applied, z/XDC will display a decimal interpretation instead of trying to resolve a 3 or 4 byte value into an address and its related object.

This performance did not occur when displaying an ADATA mapped control block (or data area) as source code images. But it did occur:

- When displaying an ADATA mapped control block (or data area) as object code (example: **FORMAT address OBJECT),**

- When displaying an SYM data mapped control block (or data area),

- In other words, When displaying mapped storage via z/XDC's internal disassembler.

MDL-1812D - This update adds HELP text for message DBC320W. see **HELP MESSAGE DBC320.**

FHC-1812C - This update fixes a bug in the disassembly of the **LOCHI, LOCGHI and LOCHHI** assembler instructions.

We would like to thank Robert Skorpil of Broadcom for bringing this to our attention.

PEM-1812B - This update is a internal changes update. However #DBCPARM required updates to reflect those changes. Otherwise, there are no visible changes to the user.

MDL-1812A - This update introduces the **SET EXITS** command, to assist in the utilization of the new c/XDC Miscellaneous Exits Interface routine introduced by **Z22-1803A.** For detailed information, see see **HELP COMMANDS SET EXITS.**

We would like to thank Ron Colmone of CA Technologies Mainframe for bringing this to our attention.

MDL-1811E - This update fixes a minor bug within c-language mapping for the c/XDC product. The bug caused the FORMAT command to drop occasional display lines when the source was obtained from a RECFM=VB dataset.

We would like to thank Ron Colmone of CA Technologies Mainframe for bringing this to our attention.

FHC-1811D - This update fixes a minor bug in DBC-1811C that will cause CDF sessions to fail with a S0C4 or with garbled SYSLOG messages.

DBC-1811C - This update adds a significant new facility (Latent Commands) to z/XDC, and it also tweaks numerous other commands. For detailed information,

see HELP * DBC-1811C.

CBC-1811B - Changes the #XDCHOOK macro to use instructions at the Principles-00 level. PEM-1806E introduced 2 instructions at the Principles-06 level.

We would like to thank Gregory Mercer of Rocket Software for bringing this to our attention.

CBC-1811A - When in line mode, the output of L REGS, L RWREGS, L FREGS, and other commands was truncated. This fix resolves that problem.

We would like to thank Ray Mullins of Trident Services for bringing this to our attention.

FHC-1810H - DBC-1809H fixed a bug in ADATA processing and bit-alignment handling in ADATA maps. That fix lead to the discovery of an issue in the ADATA generated by the Dignus System Assembler tool.

This fix tweaks DBC-1809H to tolerate the ADATA source image record from System Assembler.

We would like to thank Kerry Tenberge and Nitzan Mordhai of BMC for bringing this to our attention.

FHC-1810G - Fixes a bug similar to the one fixed in FHC-1810D. Instead of license processing being affected, this time it’s HELP.

We would like to thank Dave Stedman of Banknet for bringing this to our attention.

FHC-1810F - Fixes a bug introduced by PEM-1805B that may lead to a S0C4-10 when debugging in a cross memory environment (HASN<>PASN)

We would like to thank James Magill of BMC for bringing this to our attention.

FHC-1810E - This is a small tweak to the FHC-1810D fix.

We would like to thank Dave Stedman of Banknet for bringing this to our attention.

FHC-1810D - This fixes a problem with license processing in constrained environments, such as starting a z/XDC debugging session in the initiator task.

We would like to thank Dave Stedman of Banknet for bringing this to our attention.

DBC-1810C - This update has internal changes only. There are no external changes that customers will see.

FHC-1810B - This update fixes 2 bugs introduced by PEM-1805B that may cause S0C4 and S0C1 abends and will cause z/XDC to become "stuck" tracing the same instruction over and over when z/XDC is used in any sort FRR protected, cross memory environment.

We would like to thank Bob Price of BMC for bringing this to our attention.

DBC-1810A - This update adds a new shortcut command. **LO** can be used on machine instructions to display all storage locations that the instruction references. It issues a **LIST OPERANDS** command against that instruction. For more information, see HELP SHORTCUTCOMMANDS LO.

I've also added cross-links to the various HELP SHORTCUTCOMMANDS topics from the Syntax discussions for those commands that have shortcuts.

Shortcut command input fields are two characters wide. With the advent of 2-character shortcuts, I have create a requirement that all shortcut commands start in the leftmost of the two columns. z/XDC will no longer accept shortcut commands that start in the 2nd column.

DBC-1809I - This update includes a variety of small fixes and changes:

- Added a warning message (DBC496W) when the server job (xxxSRVER) is not up. This message is displayed:
  - At the start of the debugging session,
  - As a left-side annotation on z/XDC's Title Line: (DBC496W!) appears,
  - Among the messages produced by the **LIST MSGS** command.
- Added **(q)** to the **LIST MAPS** report to show which map (if any) has been set as the default map by the **SET QUALIFIER command.**
- Added maintenance update level information to the **DBC830I message in z/XDC's Opening Salvo.**

- Added support for an **H** shortcut command as an alias for **S in Built-in Help. This improves consistency between navigating Built-in Help displays and navigating a LIST HELP report. H** can now be used in both places for selecting topics for display.

DBC-1809H - This update fixes a bug in ADATA maps. When a single DC or DS instruction defines multiple bit-align fields, the Assembler mashes them altogether into a single field. The ADATA mapping, however, was treating each bit-aligned subfield as starting on a byte boundary. Worse, this was throwing off the locations in the map of all following fields! This has been fixed.

I've also taken the opportunity of making some tweaks to the SYSINFO script.

DBC-1809G - Spurious DBC500T message issued followed by product termination. (This is a bug introduced by DBC-1809C. Fortunately, we caught it before any customers saw it).

DBC-1809F - This is a stopgap update to detect and fix a rare corruption in the DCVTHSRB field before it can cause problems.

The presence of the corruption manifests itself long long after its cause. A SA-type SLIP trap is needed, but we have had one set for many days now, but the corruption has not reoccurred.

Unfortunately, this stopgap can only detect the corruption after it has occurred, so a dump at that point in time is pointless. So all this update does is reports the event and then corrects the damage.

CBC-1809E - **XDCREMOV** is a new utility that has been added to z/XDC's distribution libraries. Its purpose is to disable (and, therefore, logically remove) z/XDC's System Interface from your running z/OS.

This is occasionally necessary when you find that you have to back off from your current maintenance level to an older one. Normally, z/XDC does not allow that to happen.

But if you are really really sure that you really really want to do that, then you can run this **XDCREMOV** utility to do that. Then run **XDCCALLA IEFBR14** (for example) to install a complete replacement of the System Interface at whatever maintenance level you like.

XDCREMOV is documented in **HELP SUPPORT MAINTENANCE REMOVAL.**

DBC-1809D - This update adds the word **(AUTH)** to the Title Line of most z/XDC displays. (This removes the need to issue **LIST XDC** or LIST MSGS commands just to learn z/XDC's current authorization level.)

DBC-1809C - This update contains a grab bag of externally minor changes. (The internal work, on the other hand, was substantial!)

- Update DBC-1804E created an exposure for sB38-08 abends. That has been fixed.
- I made a tweak to z/XDC's disassembler (the FORMAT command) such that conditional jump/branch instructions following branch/jump-and-save type instructions will now show their arithmetic names (JNZ for example) instead of their logical names (JNE).
- The doc for the System Initialization Report **(DBC514I)** has been substantially rewritten to give more complete information for when things go wrong during the installation of z/XDC's System Interface Elements.
- I significantly revised z/XDC's Summary Dump messages (DBC914T) to provide better information for troubleshooting product abends without having to resort to dumps.
- I corrected misinformation in our formal **Statement of Support.**
- I significantly revised the way that z/XDC carries maintenance level information internally. This was the most complex of the changes in this update, yet there is no external changes created by this effort.

DBC-1809B - This update fixes a couple of sB78-8 abends that occurred under circumstances that are highly unlikely to occur at customer sites.

DBC-1809A - This fixes a pervasive bug (introduced by FHC-1808F) that would cause a variety of execution failures in z/XDC. One manifestation was s0C1 failures when issuing the LIST ASID command while z/XDC was running in problem state.

Fortunately, this bug was discovered in house prior to any customers actually installing the broken maintenance.

MDL-1808G - This fix adds support to c/XDC for the display of inherited variables

within a c++ class, and for variables defined within a c++ namespace.

We would like to thank Adeline Ho of Visa for bringing this to our attention.

FHC-1808F - This update corrects a severe performance problem that occurred when z/XDC was running **non-authorized.** (This did not occur for programs running authorized.)

For non-authorized debugging sessions, z/XDC needed to call our Service SVC to perform various storage protection tests that can only be performed while authorized. These Service SVC calls added up to the point that they caused a very noticeable increase in z/XDC’s command execution times.

Authorized debugging sessions are able to perform these storage protection tests inline, without resorting to an SVC service. That is why authorized sessions were not affected by this performance issue.

With this fix, when z/XDC is running non-authorized, and if z/XDC’s Server Space is available, then instead of calling the Service SVC, a PC routine will be invoked to perform the same protection tests. (PC routines have vastly less System overhead than do SVC routines.)

For non-authorized debugging sessions, you may expect to see many commands run in about a third the time (or better) than they took previously. In fact, they should run nearly as quickly as they do in authorized debugging sessions. (Authorized debugging sessions will not see any changed in performance.)

As pleased as we are with the success of this effort, we still feel that z/XDC runs slower than we’d like. So we are continuing our efforts to find additional improvements to make.

We would like to thank Adeline Ho of Visa, and David Stedman and Ralph Spadafora of Banknet for bringing this problem to our attention.

FHC-1808E - This fix makes several minor tweaks to z/XDC’s internal trace (ITRACE). There are no user visible changes.

MDL-1808D - This fix repairs an unending loop in the c/XDC client that can occur if the c/XDC server encounters an error at a critical point in it’s processing.

We would like to thank John Moore of ASG Technologies for bringing this to our attention.

FHC-1808C - This update fix 3 bugs introduced by the PEM-1806C fix.

- When setting a deferred breakpoint/HOOK using a pure program name or a program name plus an offset, instead of setting the breakpoint/HOOK relative to the entry point of the program, the the breakpoint/HOOK will be set relative to the physical start of the program.

- When setting a deferred breakpoint/HOOK into key 0, store protected storage, the breakpoint/HOOK will not actually be set even though z/XDC will report one was set.

- When setting a deferred breakpoint/HOOK in a RMODE64 program, processing may in a couple of different ways.
  - A S0C4 abend.
  - A pair of DBC912E and DBC909E error messages.
  - With no errors but the HOOK may not be correctly set.

  This particular bug depends on the shape of storage and may also present itself in ways not listed.

I would like to thank Rod Damron of ASG for bringing these issues to our attention.

FHC-1808B - This update fixes a bug with the LIST TASK command that displayed ***UNKNOWN*** or a random string as the name of a task. This mainly occurred when the user defined a TCBFSA that resides in 31-bit storage.

We would like to thank Dick Nunke of BMC for bringing this problem to our attention.

CBC-1808A - This update fixes DEAD traps #8188 and #8189 that could occur when trying to format storage that is located above the bar and is mapped by an ADATA DSECT map.

We would like to thank Eddie Ruth of Vanguard Integrity Professionals for bringing this to our attention.

DBC-1807F - This update fixes a couple of bugs in the **#XDCHOOK** macro:

- The user program's **SYSSTATE** settings were being corrupted. For example, the **&SYSALVL** value was always being changed to 2 (ARCHLVL=2).
- Under certain circumstances, the TITLE, PUSH, POP and DROP Assembler instructions were being replaced by macros, but customers probably did not notice this because there was little function change, and the only visual changes were:

- The loss of comments from the PUSH, POP and DROP instructions,
- The echoing of the TITLE instruction on the page just prior to where it took effect.

This update changes the **#XDCHOOK** macro so that it now preserves and restores the Assembly State data that it might otherwise alter.

I would like to thank Dave Kunkle of Informatica for bringing the SYSSTATE issue to our attention.

PEM-1807E - This update fixes a bug in PEM-1806C that may cause S0C6 abends when a hook is set in page protected storage.

FHC-1807D - This update fixes a bug in the DBC072E message displayed when z/XDC is unable to acquire storage.

I would like to thank Ray Mullins of Trident Services for bringing this to our attention.

CBC-1807C - This update changes the order of the display of the values of the operands shown by the **LIST OPERANDS** command. For example, when the instruction is an MVC, CLC, or XC, the first operand storage value is now displayed first.

FHC-1807B - This update fixes a bug in PEM-1806C that may cause S0C4-10 abends when starting z/XDC debugging sessions on zOS 2.2 or older systems.

I would like to thank Howie Nayberg of Nastel Technologies for bringing this to our attention.

DBC-1807A - This update adds a new operand to the UP and DOWN scrolling commands:

- **UP TRACE** moves the Scroll Area upwards to position it to the display (usually a WHERE display) produced first after the next prior time z/XDC received control from the program being debugged.

- Likewise, **DOWN TRACE** moves the Scroll Area downwards to position it to the display produced first after the next following time z/XDC received control from the program being debugged.

After you have stepped through your code for a bit, you can use UP T;RETRIEVE to replay backwards your program’s execution through its code.

You can run the "movie" forwards and backwards by intermixing DOWN T with UP T.

I got this idea from one of our customers who would prefer to remain anonymous.

PEM-1806E - This update fixes a bug introduced by PEM-1806C in the #XDCHOOK macro.

The #XDCHOOK macro was generating the wrong values for some reason codes under some circumstances.

Please note that as a consequence of this change you should reassemble any source that has the #XDCHOOK macro in it, and ensure that you use the equates for the reason code values rather than the absolute numeric values.

FHC-1806D - This update fixes a bug in HOOK processing when all the following conditions are met

- AR14 contains some value.
- A HOOK has been used to initiate the debugging session.
- Another HOOK has been set in and hit in the debugging session.
- The 2nd HOOK set does not specify the ASNAME= operand.
- The target code is running in supervisor state.

Depending on the value in AR14, this may cause a s0C4 or a s0E0 abend.

We would like to thank Ken Scott of Trident Services for bringing this to our attention.

PEM-1806C - This update alters HOOK and deferred breakpoint support so that they now support setting hooks and deferred breakpoints above the bar.

In addition, dynamic hooks can optionally support code running in SECONDARY or HOME ASC-MODE.

Also, the #XDCHOOK macro has been changed so that it no longer has any literals and all reason codes have been made unique. Please note that as a consequence of this change you should reassemble any source that has the #XDCHOOK macro in it, and ensure that you use the equates for the reason code values rather than the absolute numeric values.

The HOOK and HDEFERRED commands have 2 new operands that allow you to specify a work register and whether or not the hook needs to support SECONDARY or HOME ASC-MODE.

When setting a hook in above-the-bar code you must nominate a work register.

When setting a hook that needs to support SECONDARY or HOME ASC-MODE you must nominate a work register.

Deferred breakpoints (and hooks) are now supported for load modules that reside above the bar (That is, the binder had RMODE(64) specified and the operating system is z/OS 2.3 or later).

If a deferred hook does not have a work register specified on the definition then an error message is produced if the load module resides above the bar (RMODE64). (Note that this error situation can only be detected when the module is loaded).

The LIST BREAKPOINTS command (when listing deferred hooks) and LIST HOOKS commands have had their output messages changed to show the work register number and whether the hook supports odd ASC modes.

MDL-1806B - This fix corrects a problem in c/XDC **STEP OUT** in non-XPLINK execution environments.

MDL-1806A - This fix adds support to c/XDC that enables the **STEP IN** command to operate correctly for DLL Function calls in 64-bit XPLINK and 31-bit non-XPLINK execution environments.

We would like to thank John Moore of ASG Technologies for bringing this to our attention.

PEM-1805B - This update introduces to z/XDC a **new kind** of breakpoint! Users can now request that z/XDC construct breakpoints from **TRAP2 instructions instead of from invalid opcodes. For details, see HELP BREAKPOINTS TYPES.**

TRAP2-type breakpoints are of only limited usefulness. They are not intended to replace the old X'00' opcodes, but they do supplement them.

In particular, the use of TRAP2-type breakpoints does not remove the need to always establish z/XDC as your program's newest recovery routine. The reasons why are discussed in the "Why z/XDC Still Must be an Abend Recovery Routine" section of the HELP BREAKPOINTS TYPES topic.

The choice of which to use (TRAP2 or X'00') can be made:

- When you issue a **SET TRACE** command,
- And when you issue individual **TRAP, TRACE, AT,** etc. commands.

The default setting can be saved in your session profile.

The **main advantage** of TRAP2-type breakpoints is improved performance during **conditional tracing and trapping.** We still have a ways to go with this issue, but while the performance improvement of conditional tracing won't be dramatic, it will be noticeable.

There are several consequences of using TRAP2-type breakpoints to keep in mind:

- RTM processing does not get involved, so RBs (Request Blocks) will not longer be created by z/OS for running either RTM itself or z/XDC as a recovery routine (i.e. as an ESTAE).

- The whole error level vs. retry level distinction thing simply goes away. The error and retry levels become one and the same.

One somewhat dramatic consequence of all this is, when you issue a **LIST RBS** command, what you will see will not be what you are used to seeing.

MDL-1805A - This update fixes a 12K memory leak in 24-bit storage at end-of-session time. For some customers, this was causing s878 abends in the xxxSRVER job.

DBC-1804F - Two new items have been added to the **LIST FEATURES** report:

ECTG - Shows whether or not the current machine supports the ECTG machine instruction.

LPDEL - Shows the size of Link Pack Directory Entries in the current z/OS.

Also, the **HELP MESSAGES DBC670** topic has been completely rewritten to provide real world solutions to try when the DBC670Q WTOR appears.

Finally, This update adds a CPU time used field to z/XDC’s Internal Trace records generated by the SET XYZZY ITRACE command (used by ColeSoft Support when certain diagnostics are needed).

DBC-1804E - This update fixes a 16K memory leak in 31-bit storage.

MDL-1804D - This fix alters cs-cdf/XDC message DBC670 to enable system operators to control the action taken by z/XDC when a required VTAM APPLID is not available during startup.

I would like to thank Adeline Ho of Visa for bringing this to our attention.

DBC-1804C - This is a housekeeping update. It makes no external changes to z/XDC.

DBC-1804B - This is a housekeeping update. It makes no external changes to z/XDC.

PEM-1804A - This fix changes XDCCALL (and XDCCALLA, XDCCMD and XDCCMDA) so that they now support programs loaded above the bar (that is, RMODE64).

FHC-1803H - This fix re-applies fix Z22-1706A during maintenance. Due to a quirk in our maintenance process, Z22-1706A was not correctly applied previously.

I would like to thank Kevin Lynch of Winsopia for bringing this to our attention.

MDL-1803G - This fix allows c/XDC to tolerate the presence of DOS-style filenames embedded within program objects and DWARF data as generated by the DIGNUS Systems/C compiler.

I would like to thank Peter Haines of GT Software for bringing this to our attention.

DBC-1803F - This update adds a secure mechanism by which an authorized debugging session can confer authority upon a non-authorized session without changing the authority level of the program being debugged. This update includes the following changes:

- A **SET AUTH** command is added by which an authorized debugging session can designate which non-authorized session is to be made authorized.

- A **GO NOWHERE** command is added by which the non-authorized debugging session targeted by the SET AUTH command can complete the authorization process.

These two commands replace the **MAKEAUTH** scripts that previously had to be used to accomplish the same purpose. The scripts have not been removed, but commentary has been added to suggest that these new commands be used instead.

The need for this simplified process arises most commonly when a non-authorized program needs to step execution through reentrant load modules. Such modules usually are loaded into key 0 storage, which is problematic. While z/XDC has mechanisms for causing such modules to be loaded into key 8 storage (see HELP DEBUGGING REENTRANT), such schemes could not always be used. This update provides an alternative.

Z22-1803E - Fixes a bug in z/XDC that displays a blank screen on the initial connection to cs-CDF.

Z22-1803D - Fixes a bug in c/XDC that prevents the display of certain valid variables and structures.

Z22-1803C - Fixes a bug in c/XDC and server/XDC that causes integrity problems within the client/server work element queues.

I would like to thank Adeline Ho of Visa and Eric Johnson of CA Technologies for bringing this to our attention.

Z22-1803B - Fixes a bug in c/XDC that prevents the mapping of CSECTs that contain static functions only.

I would like to thank Nitzan Shahar and Mike Shorkend of Attunity for bringing this to our attention.

Z22-1803A - Introduces support for a new z/XDC user exit. The exit is used by c/XDC during variable discovery when a program uses non-standard Register Savearea formats and/or linkages. In such cases, a user-written exit routine is the only way to communicate to c/XDC who the subroutine’s caller is and what its registers are.

This kind of problem can arise in Metal C programs where the customer has replaced standard prolog logic with custom prologs.

I would like to thank Fred Bohle of Rocket Software for requesting this feature.

Z22-1802H - Fixes a bug in Z22-1802D that causes S0C4 abends during task termination in the z/XDC Server Space.

I would like to thank Slavomir Kucera of Computer Associates for bringing this to our attention.

Z22-1802G - Fixed a bug introduced in Z22-1709A where an incorrect field in the CDE was used on zOS 2.2 and older systems.

I would like to thank Bill Allen of Information Builders for bringing this to our attention.

Z22-1802F - Fixed a bug introduced in Z22-1707B where the storage location of any extent in a program object beyond the first is resolved incorrectly.

I would like to thank Lev Perelmuter of Information Builders for bringing this to our attention.

Z22-1802E - To assist performance, z/XDC maintains an internal Load Module History Table (LMHT) that is used to shorten the time taken by the many load module lookups that z/XDC performs. This update adds an internal command (LIST XYZZY LMHT) that vchecks the table and reports information about it. (The command was added because of suspected problems with the table.) This command will not be further documented within the product.

Z22-1802D - This update fixes a memory leak with cs-cdf/XDC in the 24-bit private region.

I would like to thank Tony Curry and Bill Mileski of BMC for bringing this to my attention.

Z22-1802C - This fix updates z/XDC’s ability to scan for modules in CICS 5.4.

I would like to thank Ian Sheehy of CNESST for bringing this to my attention.

Z22-1802B - This update fixes a bug with determining a clone’s name.

Z22-1802A - This update redesigns z/XDC’s internal interface for its internal Abend Protection Management Routines. The intention is to improve the efficiency of that interface resulting (hopefully) in a noticeable improvement in performance.

We also (finally) write Built-in Help doc that was missing for messages DBC164 thru DBC167.

Z22-1801E - This update enables fix Z22-1801B, which was initially published disabled.

Z22-1801D - This update fixes a S0D7-25 abend during cross memory debugging.

Z22-1801C - This update corrects a subtle problem with deferred variable discovery of C language structures introduced by fix Z22-1801B.

Z22-1801B - This update corrects an ABEND0C4 that can occur during c/XDC variable discovery.

I would like to thank Slavomir Kucera of CA Technologies for bringing this to our attention.

Z22-1801A - This update makes several changes to z/XDC. They include the following:

- The **LIST MOBJECTS** command has been updated to display the above-the-bar **MEMLIMIT** size and usage information. (This is addition to the memory object descriptions it was already displaying.)

- Both the Built-in Help and all messages have been updated to discontinue using the terms kilobytes, megabytes, gigabytes etc. Instead, the more precise terms, kibibytes, mebibytes, gibibytes, will be used.

  This is because of the ambiguous meanings of the older terms: For Example, does kilobyte mean 1,000 bytes? Or 1,024 bytes? Kibibytes, on the other hand, unambiguously means 1,024, and mebibytes means 1,024**2, etc.

  For more information, checkout en.Wikipedia.org/wiki/Kibibyte.

- Messages that may include very large decimal numbers may now be displayed as scaled values. They may be displayed, not as byte counts, but as counts of kibibytes, mebibytes, gibibytes, etc.

  This scale will be indicated by a suffix letter: k, m, g, t, p and x. You can see examples of this in the displays produced by the LIST MOBJECTS command.

- Commands that accept raw hex values as inputs have always also accepted decimal numbers when that number was trailed by the letter **n.** For example, **FORMAT 10** and **FORMAT 16N** both display the location of the System’s CVT anchor field.

  This support has now been extended to all scaling factors: k m g t p and **x.** So for example, **FORMAT 16M** will show the location of **the line,** while **FORMAT 2G** will show the location of **the bar** (not much there, I’m afraid). For more information, see HELP COMMANDS SYNTAX NUMERICDATA.

- A new, far more efficient and far more flexible Internal Trace

(ITRACE) has been implemented to assist with debugging issues that may arise from time to time in the product. Efficiency improvements include:

- Using a very large above-the-bar memory object for buffering ITRACE records.
- Using individual, private storage ITRACE buffers instead of sharing one in common storage.
- Not writing ITRACE records to DASD until an export command is explicitly issued to do so.
- Supporting filtering to suppress by type unneeded ITRACE records.

The ITRACE is managed via SET XYZZY ITRACE commands. Its state can be displayed via LIST XYZZY ITRACE commands. These commands are not further documented in the Built-in Help.

Z22-1712A - This update corrects an ABEND0C4 that can occur during c/XDC variable discovery.

I would like to thank Slavomir Kucera of CA Technologies for bringing this to our attention.

Z22-1711C - This update adds support for two mechanisms for displaying the length of the various C language variables, arrays, and structures identified by c/XDC.

- First, this update adds a new command - **LIST SIZEOF** - to c/XDC specifically designed to select and display the length of the various C language variables, arrays and structures.

- Second, this update adds support for a fourth column within the c/XDC **LIST VARIABLES display which contains the storage length of the various C language variables, arrays and structures.**

By default, the new display column will not be immediately visible. To make the lengths appear, the user needs to make use of one of two commands:

- The **RIGHT command will shift the fourth column into view**

- The **SET VDISPLAY command can make adjustments to the width of the other three columns in the display making room for the length column.**

Please note that because the size of a structure is not recorded in DWARF data, c/XDC has to manually calculate the size of the structure. The size by c/XDC may differ by a few bytes from the value reported by the SIZEOF() function.

I would like to thank Adeline Ho of VISA for requesting this feature.

Z22-1711B - This update adds support for the rest of the Z14's new machine instructions and their several extended mnemonics. Since many of those new instructions are vector instructions, I took the opportunity to review and correct support for many of the older instructions as well.

One of the things I discovered along the way was that support was broken for zapping those extended mnemonics that are defined as if they had 4-byte opcodes. (VFTCISB is one example. There are literally hundreds more!)

I also added support for extended mnemonics that are defined as if they have 5-byte opcodes. This allowed me to pick up support for a few mnemonics that I was unable to support from prior releases. Examples: LLHFR NHHR OHHR XHHR LHHR etc. This support is also required for many new extended mnemonics as well.

And finally, I added support for the **VNOT** extended mnemonic. The weird thing about VNOT is that, while all other extended mnemonics coerce specific values in various operand nibbles, the VNOT instead is defined by the **relationship** between two of its operands: It is any VNO instruction where the V2 and V3 registers are the same register! What this means is that there are 32 variations of the VNOT "opcode".

Z22-1711A - c/XDC contains a workaround fix to allow the mapping of program objects that do not contain a reference to class C_@@PPA2 within CSECT IEWBLIT, which are not otherwise supported by IBM's Common Debug Architecture. To fully support c/XDC variable discovery, the workaround needed to be updated to contain an entry for class C_WSA as well.

I would like to thank Stephen Wilson of IBM Sterling for bringing this to our attention.

Z22-1710D - This update adds an internal debugging feature used mainly to assist in the debugging of one release of z/XDC with another.

Z22-1710C - This update allows c/XDC to tolerate and understand CSECTs written in Language Environment compliant Assembler within a C/C++ program object.

I would like to thank Stephen Wilson of IBM Sterling for bringing this to our attention.

Z22-1710B - This update fixes 2 minor bugs with the LIBRARYLIST command.

- LIBRARYLIST CSOURCE REDIRECT may silently fail.

- When deleting individual entries in a candidate list, the last entry in the list will cause z/XDC to abend with a S0C4.

Z22-1710A - This update fixes a problem with the **@CDP** and **@BEA** built-in equates. z/XDC was always assigning them to home space locations even when the execution address space was not the home space. This update causes them now always to be assigned to the **primary** address space.

For the **@CDP** equate, this update completely fixes the problem, but for the **@BEA** equate, it's a different story... The problem is, the address space within which the flow of execution was most recently changed is not always knowable. So unconditionally assigning the BEA into the primary address space, while usually is correct, is not always correct. Built-in Help has been updated to explain this. For more information, see HELP COMMANDS LIST BEA.

I would like to thank Bob Price of BMC Software for bringing this problem to our attention.

Z22-1709E - This update adds support for the Z14's new **BIC** instruction and its several pseudo-mnemonics (BIO BIE BIL BI etc.). z/XDC now understands:

- How to format BIC instructions (using its pseudo-mnemonics when appropriate),
- How to find the 8-byte pointer field (even when it's in a different address space or data space),
- How to interpret Point-and-Shoot commands issued against BIC instructions (They are resolved to the actual branch target, not to the pointer field.),
- How to TRACE through BIC instructions.

Z22-1709D - Who knew! Apparently, when z/OS's **SETPROG LPA,DELETE** command deletes a module from the DLPA, it:

- Nulls out the CDNAME field,
- Sets the CDENTPT field to X'7FFFFBAD',
- FREEMAINs the storage that had been occupied by the deleted module.

But:

- It does **not** clear or otherwise nullify the module's Extent List.
- And is does not clear the CDXLE flag.

This little detail was causing z/XDC to misidentify storage that had been occupied by a deleted module. This gets to be a real pain in the butt when said storage is reused for a **SETPROG LPA ADD** added module and then you want to try to map that module.

I would like to thank Dave Cole of ColeSoft for bringing this to my attention.

Z22-1709C - This update brings the **LIST CRWn** up to the **Z14** level of zSystems Hardware. The displays of almost all individual control registers have been at least tweaked by this update, but in specific regard to the **Z14:**

- **LIST CRW0** now reports:
  - The Clock Comparator Sign Control
  - The Instruction Execution Protection Enablement Control
- **LIST CRW2** now reports:
  - The Guarded Storage Facility Enablement Control
- **LIST CRW9** has been brought up to date.

Z22-1709B - This fix detects and repairs a VTAM environmental problem that led to connection errors and message DBC653E in cs-cdf/XDC. Message DBC690E replaces DBC653E to indicate that the connection request should be reattempted.

- New Message DBC690E

I would like to thank Deborah Greer of ASG Technologies, Bill Mileski of BMC, Narender Eshwar Sajnani of CA Technologies, and Martin Wittow of Rocket Software for bringing this to our attention.

Z22-1709A - This update is a further fix to a performance issue involving possibly thousands of silently suppressed s0C4 abends that occurred whenever z/XDC needed to scan modules located in the LPQ, the DLPA and the PLPA (which is pretty much all the time). This issue manifested itself only on z/OS R2.3 Systems. It resulted from a misdesign of the previously published RMODE64 support (Z22-1707B).

We’d like to thank Bill Mileski of BMC Software for bringing this problem to our attention.

Z22-1708B - This update fixes a performance issue that caused dozens (or possibly hundreds) of silently suppressed s0C4 abends upon every press of the ENTER key. This arose from a misdesign of the logic that determined the length attribute of the **@PRIOR built-in equate.**

Z22-1708A - According to the doc, when storage with the range of an Area Equate is displayed, and SET FORMAT OFFSETS is in effect, the offsets displayed would always be computed relative to the start of the Area Equate. Well... not always.

Previously, this would not happen for Area Equates created inside mapped csects and dsects. This update removes that exception. Now, it doesn't matter where the Area Equate is placed, it will always override z/XDC's choice for an offsets base.

We would like to thank Dante Ferman of Innovation Data Processing for bringing this issue to our attention.

Z22-1707E - This update improves the performance of the c/XDC STEP command for some C language statements that contain branch-and-count type machine instructions.

I would like to thank Fred Bohle of Rocket Software for bringing this to our attention.

Z22-1707D - This update implements a new **CFRIENDLY** operand on the **LIST LKEDMAP** command. When present, the command's report is pared down to exclude all LE C boilerplate. All that remains are those external symbols (ESDs) that were created directly from customer written C code, C functions and Assembler CSECTs.

This operand is intended to make it easier for customers to see those code sections that contain user written C language statements (and Assembler code), and therefore, are sections that the customer might want to map.

The, following Built-in Help topics have significant changes:

- HELP COMMANDS LIST LKEDMAP
- HELP DEBUGGING C
- HELP DEBUGGING C MAPPING [new]

So new PDFs of the Built-in Help have been published at www.colesoft.com.

This update also fixes a problem introduced by Z22-1707B. That update broke the ability to set deferred breakpoints based on csect information from a Binder map.

We'd like to thank Bill Mileski of BMC Software for bringing this problem to our attention.

Z22-1707C - This update corrects an ABEND0C4 that can occur during c/XDC variable discovery for AMODE(64) CSECTs.

I would like to thank Peter Goldberg and Lev Perelmuter of IBI for bringing this to our attention.

Z22-1707B - This update implements **RMODE64** support in z/XDC for Load modules located **above-the-Bar** They are now understood and properly handled. For more information, see HELP DEBUGGING RMODE64.

As part of this support, following reports and displays have been revised (and hopefully) improved:

- **LIST FEATURES:** A line item has been added to this report indicating whether or not RMODE64 support is present in z/OS.

- **LIST MAPS:** The following changes have been made:
  - The display now has a less sloppy appearance.
  - A map's data type (ESD, SYM, ADATA or DWARF) is now shown.
  - Maps representing Privately Loaded modules are now identified (p).

- **LIST PGMS:** The following changes have been made:
  - The report columns have been rearranged into a more sensible order.
  - The ENTRY ADDRESS field has, of course, been widened to accommodate 8-byte addresses.
  - When a module's storage is inaccessible to z/XDC, its address is now reported, and s0C4 is now indicated in the SUBPOOL column. (Previously, (INVALD) was shown in place of the module's address.)
  - Automatic equates are now generated to represent the locations of the CDEs, LPDEs and CICS APEs from which the report is built.
  - A SEQ# column has been added to correlate with the automatic equates.

In addition, the **LIST PGMS** command has been revised to generate automatic equates to represent those CDEs, LPDES and CICS APEs from which its report is built.

The **LIST EQUATES** command has been revised to display in binary sequence number order those equates whose names end with sequence numbers. (Previously, such equates were being displayed strictly in text value order. This would cause TCB#17, for example, to display ahead of TCB#2. This is particularly undesirable when large numbers of equates are involved. [Think LIST PGMS PLPA.])

The following Built-in Help topics have been revised:

```
HELP COMMANDS LIST EQUATES
HELP COMMANDS LIST FEATURES
HELP COMMANDS LIST MAPS
HELP COMMANDS LIST PGMS
HELP COMMANDS LIST PGMS REPORT
HELP EQUATES BUILTIN AUTOMATIC
HELP EQUATES BUILTIN INTERNAL
HELP DEBUGGING RMODE64 (new)
HELP MAPS PRIVATELYLOADED
HELP MAPS PRIVATELYLOADED DMAPANDUSING
HELP MAPS PRIVATELYLOADED MAP
HELP MESSAGES DBC508
```

HELP WHATSNEW Z22 POINTANDSHOOT

These changes are sufficiently extensive that we have updated the PDF versions of the manuals at our website: colesoft.com/zxdc-release-z2-2

Z22-1707A - This update splits cs-cdf/XDC message DBC653E into three distinct messages numbers. The split should shed some light on the root causes that can lead to cs-cdf/XDC connection problems at various customer sites.

- Revised Message DBC653E
- New Message DBC669E
- New Message DBC678E

I would like to thank Deborah Greer of ASG Technologies, Bill Mileski of BMC, Narender Eshwar Sajnani of CA Technologies, and Martin Wittow of Rocket Software for bringing this to our attention.

Z22-1706E - This update corrects slightly overzealous server cleanup during the termination of address spaces that have been debugged using c/XDC. The error caused some terminating c/XDC tasks to suffer an ABEND0C4 in their end-of-task exits.

I would like to thank Slavomir Kucera of CA Technologies for bringing this to our attention.

Z22-1706D - This update adds three features designed to give c/XDC testers more granular and easier mechanisms to disable C language support.

- User settings for the **SET CXDC ON | OFF** command are now preserved within the z/XDC profile.

- Support has been added for DDname **xxxCOFF** which will override a profiled **SET CXDC ON** command.

- A new command has been created **SET VSETTINGS DISCOVERY=ON | OFF which controls automatic variable discovery. This feature allows debuggers of large C-language application to use C-source mapping without incurring the CPU overhead of variable identification.**

Z22-1706C - This update contains 3 minor updates:

- Removed outdated information in HELP SHORTCUTCOMMANDS.
- Fixed a bug in LIBRARYLIST where extraneous information was being displayed.
- Updated the sample C and Metal C program, CSAMPLE/MSAMPLE, with more variable types and updated the JCL to conform to zOS 2.2 compiler

options.

Z22-1706B - This update allows c/XDC to overcome inconsistencies in the alignment and content of Metal C control block structures as constructed by the compiler. These issues prevented c/XDC from properly identifying certain CSECTs as containing mappable Metal C source code.

I would like to thank Fred Bohle of Rocket Software for bringing this to our attention.

Z22-1706A - This update fixes several problems with the **LIST TIOT** command that included DEAD traps, failures when used against Foreign Address spaces, and simply random incorrect information in the command's display.

Unfortunately, the command's logic was so badly broken that the changes for release z2.2 had to be abandoned. So this update reverts the command to its z2.1 level. Mainly, this means that support for the **SORT=** operand has had to be temporarily dropped.

I would like to thank Ian Sheehy of CNESST for bring this problem to our attention.

Z22-1705B - This is a serviceability update for our client/server products - server/XDC, c/XDC, and cs-cdf/XDC.

Z22-1705A - This is a toleration update for using z/XDC on z/OS R2.3 systems. z/XDC was failing to find load modules located in the PLPA.

I would like to thank Tony Curry of BMC for bringing this to our attention.

Z22-1704I - This update changes z/XDC's source code to use z/OS R2.3 macro libraries. This is an internal only change. There is no external effect of this update.

Z22-1704H - Update Z22-1704B made radical changes to the multi-tasking timings of our client/server products - server/XDC, c/XDC, and cs-cdf/XDC. In the specific case of cs-cdf/XDC, the server allowed the product to initialize too fast - cs-cdf/XDC tried to open it's ACBs before APPLID-defining SYSIN parameters were available to be read. In it's haste, cs-cdf/XDC attempted to use the wrong APPLIDs.

I would like to thank Bill Mileski of BMC for bringing this to our attention.

Z22-1704G - This update adds the ability to deal with square brackets for codepage 037 when used in c/XDC and z/XDC commands. This update only adds support for parsing commands, it does not correct the square brackets in XDC displays.

I would like to thank Adeline Ho of Visa for bringing this to our attention.

Z22-1704F - This update corrects a problem that may arise when c/XDC displays source-file sequence numbers for variable length record files.

I would like to thank Adeline Ho and Steve Albert of VISA for bringing this to our attention.

Z22-1704E - This update improves handling of ADATA maps for External DSECTS.

I would like to thank Bill Meany of GE Digital for bringing this to our attention.

Z22-1704D - This update fixes a bug in Z22-1702H that caused DSECT maps to be created with an invalid length.

Z22-1704C - This update adds c/XDC support for mapping CSECTs that have been lengthened by external programs after compilation.

I would like to thank Narender Eshwar Sajnani of CA Technologies for bringing this to our attention.

Z22-1704B - This update makes significant changes to the internal operation of server/XDC resulting in quicker initialization and cleaner termination of itself and the products it supports - such as c/XDC In addition, the update improves the user’s c/XDC experience on loaded systems by decreasing the occurrence of client/server timeouts during the execution of c/XDC commands. Lastly, the update corrects ABENDs that occurred within the server as a result of timeouts.

I would like to thank Alan Playford of Winsopia and Lev Perelmuter of IBI for bringing this to our attention.

Z22-1704A - This update fixes an abend s0C6 c/XDC encounters when attempting to recover from other failures in AMODE(64) environments.

I would like to thank Mike Hans of Visa, and Deborah Greer of ASG Technologies for bringing this to our attention.

Z22-1703H - This update fixes a bug introduced in the Z22-1702H maintenance.

I would like to thank Greg Grounds of Imperva and Mike Behne of BMC for bringing this to our attention.

Z22-1703G - This update changes the #DBCVRSN macro so that it can be assembled with z/OS 1.13, 1.12 and 1.11 libraries. This update does not allow z/XDC to run on any systems older than z/OS 1.13.

We would like to thank Michael Padreny of Oracle for bringing this to our attention.

Z22-1703F - This update fixes a bug in the #XDCHOOK macro when the HOOKIFACE= parameter is used.

We would like to thank Bob Edmund of BMC for bringing this to our attention.

Z22-1703E - Update Z22-1702E introduced code that tested the availability of the OMVS security database segment. OMVS authority is required for proper operation of c/XDC. This update corrects a problem where the new test interferes with the user’s execution environment.

We would like to thank John Moore of ASG Technologies for bringing this to our attention.

Z22-1703D - This update adds support for the 8-character TSO userid support announced by IBM for the upcoming z/OS R2.3.

We would like to thank Ed Jaffe of Phoenix Software for bringing this issue to our attention.

Z22-1703C - This updates adds a minor feature to the mapping logic of c/XDC for XL C/C++ programs. If the user explicitly requests that c/XDC map the first byte of Entry CSECT CEESTART (or CELQSTRT for AMODE64) c/XDC will map the CSECT that contains the user’s ’main’ function instead.

Z22-1703B - This updates corrects a looping problem within server/XDC after it recovers from an ABENDing subserver or processing task for c/XDC or cs-cdf/XDC

We would like to thank Lev Perelmuter of Information Builders for bringing this to our attention.

Z22-1703A - This update corrects a minor problem in c/XDC pertaining to the ZAP Point and Shoot command under the ISPF terminal interface.

Z22-1702H - One emerging z/XDC limitation is its inability to load map data (other than DWARF data) for load modules loaded into storage via Unix System Services (USS) and from files located in a USS file System.

Now while there remains much to be done before this all will work properly, this update does create a workaround to make it possible to load both module maps and csect maps for USS loaded load modules. It's a rather ugly workaround, but it's a workaround nonetheless.

Briefly, if you have copies of your load modules located in classic PDS[E] libraries (that's the ugly part), then you can use z/XDC's mapping support for **Privately Loaded** load modules to build and place module maps on top of USS loaded modules.

One part of this update changes the way in which z/XDC uses the MAP command's first operand when (and only when) the **START= operand is used.**

For details of the USS mapping process and an example, see HELP COMMANDS MAP PRIVATELYLOADED.

This update also includes the following significant changes to Built-in Help:

- The HELP COMMANDS MAP topic has been reorganized and broken up into 9 subtopics.
- The HELP COMMANDS MAP PRIVATELYLOADED topic contains a substantial amount of new information pertaining both to Privately Loaded modules and USS loaded modules.
- The HELP MAPS PRIVATELYLOADED USSFILES topic is a new topic containing a brief discussion of dealing with USS loaded load modules.

Look for more improvements coming in future updates.

Z22-1702G - Update Z22-1701D introduced a dependency on c/XDC having an understanding of the format of Language Environment's PPA2 timestamp data area. Z22-1702G corrects an error in locating the SOS options area when the timestamp area also contains a Service level string.

We would like to thank John Moore of ASG Technologies for bringing this to our attention.

Z22-1702F - This updates corrects a secondary problem where c/XDC attempts to continue processing a command after a processing task terminates.

We would like to thank Bill Allen of Information Builders for bringing this to our attention.

Z22-1702E - This updates allows c/XDC to detect and issue messages pertaining to the absence of an OMVS segment in RACF security profiles. c/XDC may require OMVS services to function properly.

We would like to thank Peter Morrison of Rocket Software for bringing this to our attention.

Z22-1702D - This updates fixes a problem in map management that causes z/XDC to prematurely discard ESD data, SYM data, or ADATA for certain aliased modules.

Z22-1702C - This updates fixes a problem where c/XDC interferes with assembly language ADATA or SYM data mapping in z/XDC

We would like to thank James Boysen of BMC Software for bringing this to our attention.

Z22-1702B - This updates the sample JCL in the z/XDC package to compile, link and run the CSAMPLE program. JCL also have been added to compile, link and run the MSAMPLE program, which is a Metal C version of the distributed LE C sample program.

Z22-1702A - This update fixes a problem where mapping C programs can result in XDC abending at dead trap #2194.

We would like to thank Lev Perelmuter of IBI for bringing this to our attention.

Z22-1701K - This updates fixes a wildcard resolution bug in SET READ DSN= and the READ commands.

We would like to thank Ray Mullins of Phoenix Software for bringing this to our attention.

Z22-1701J - This update, along with Z22-1701I, fixes an error within c/XDC AUTOSTEPing where certain code paths cause z/XDC to remove itself from the debugging session while active breakpoints remain in the user's CSECT. ABEND0C1 occurs when execution encounters the orphaned breakpoints.

We would like to thank Bob Fowler of Rocket Software for bringing this issue to our attention.

Z22-1701H - This update fixes an error with CAP management in the LIBRARYLIST command when Z22-1701E is installed.

Z22-1701G - This update adds a workaround for a problem IBM's Common Debug Architecture encounters while attempting to locate the Program Prologue Area 2 in certain XL/C Elements.

Z22-1701F - This update changes help text for the STEP command.

Z22-1701E - This update fixes an error within c/XDC that caused it to prematurely consume c/XDC licensing CAPs.

Z22-1701D - This update fixes an error within c/XDC mapping when identifying a CSECT's source language within a multi-language program.

We would like to thank Ron Colmone of CA Technologies and Deborah Greer of ASG Technologies for bringing this issue to our attention.

Z22-1701C - This update fixes a problem with use of the **#XDCHOOK** macro. An inner

macro (#DBCVRSN) calls z/OS's SYSSTATE macro with an ARCHLVL= operand. In this z2.2 release, the macro was changed to use ARCHLVL=OSREL. That works fine in z/OS R2.1 and newer systems, but in R1.13 and older systems, it blows up the assembly. That's because support for ARCHLVL=OSREL was new in z/OS R2.1.

#DBCVRSN has been changed to be sensitive to SYS1.MACLIB's release level.

We would like to thank Neil Grobler of ASG Technologies and Dave Warner of Rocket Software for bringing this problem to our attention.

Z22-1701B - One customer reported a problem with the **#XDCHOOK** macro. They couldn't use it because, under the covers, it would generate a QMARK equate. That would lead to assembly errors because their own shop had a "common definitions" package that also generated a QMARK equate.

So I've updated the **#DBCVRSN** macro (the inner macro that creates QMARK) to change the names of three equates as follows:

- BANG is now INDIR64.
- QMARK is now INDIR31.
- PCENT is now INDIR24.

We would like to thank Brian Vohs of Tone Software for bringing this problem to our attention.

Z22-1701A - This update does the following:

- It adds a **SET CXDC ON|OFF** command.
- It revises the **LIST CXDC** command to provide more information about the licensing, availability and usability of the **c/XDC** Licensed Feature.
- It adds an **enable/disable c/XDC** line item to the **HLL Settings** panel of the profile menuing System.
- It significantly improves the content of and doc for the DBC852 message.

Z22-1612G - This update fixes a deadlock that sometimes occurs when debugging an SRB when no Formal Proxy Tasks are available. This situation requires z/XDC to ATTACH a new Proxy Task; however, the ATTACH'ing process stalls.

We would like to thank Jeremy Schwartz of Imperva for bringing this family of issues to our attention.

Z22-1612F - This update adds **ISGYQUAARQ** (and friends) to the set of IBM Control Blocks whose dsect maps can be loaded by a **DMAP xxxMAPS.dsectname** command.

We would like to thank Ray Mullins of Phoenix Software International for bringing this issue to our attention.

Z22-1612E - This update fixes a problem that arises within ColeSoft's internal development process. Yeah, we found a way to make every attempt to use any clone of any release of z/XDC fail. Just gotta do something stupid when using one clone to debug another clone's Service SVC. (Tee hee hee)

This update makes that problem considerably less likely to occur.

This problem would never occur at a customer's site.

Z22-1612D - Implemented the **RELEASECAPS** operand for the **GO** command. This allows a user to end a debugging session without also ending the program being debugged.

I would like to thank Alla Bord of MVS Solutions for making the suggestion about doing this.

Z22-1612C - This update fixes two minor issues with shortcut commands available within a LIST VARIABLES display. First, **D** (display) is now a valid shortcut. Second, the shortcut generated for **F (format) now includes the DATA** formatting option.

Z22-1612B - This update fixes a bug in error messaging when the LIBRARYLISTS command has not been used to define a location for C source, and the program was compiled with the source supplied in the JCL as an inline SYSIN dataset.

Z22-1612A - This is an update to z/XDC's Opcodes Tables. Mainly, it adds a large list of Extended Mnemonics that I had overlooked for the various Compare and Branch/Trap instructions as well as the several Load on Condition instructions.

This update also fixes a problem with the ...NE, ...NL and ...NH Extended Mnemonics for the Compare and Branch/Trap instructions. Those instructions do not permit masking for the 03 condition code, so in those cases, the masks need to be B'0110', B'1010' and B'1100', respectively. (This is different from all other ...NE, ...NL and ...NH Extended Mnemonics, all of which expect the B'xxx1' bit to be on in the mask.)

The main results of this update are:

- The **FORMAT** command now produces more accurate disassemblies,

- The **Z** shortcut command's resulting zap is now correct for the affected machine instructions.

I would like to thank Charlie Pitts of Software AG for bringing this second problem to our attention.

Z22-1611A - This update fixes a bug in the LIBRARYLISTS command where CSOURCE redirects are ignored.

Z22-1610A - This update unlocks z/XDC to permit it to run.

Help Whatsnew Z22 THingsfixed Maintenance DBC-1902a

The major change implemented by this update is the addition of a **LIBRARY= operand to the PROFILE READ** and **SAVE** commands. This operand will allow you to load and save Session Profiles from/to arbitrary libraries, not just those allocated to xxxPROF, ISPPROF and ISPTLIB ddnames. For more information, see:

- HELP COMMANDS PROFILE READ
- HELP COMMANDS PROFILE SAVE

This change will be particularly helpful to those of you who debug programs running in the batch, where the xxxPROF, ISPPROF and ISPTLIB ddnames usually are not available.

It also will make it easier to create System wide profiles located in ISPF Table Libraries. There are some examples of this in HELP COMMANDS PROFILE SAVE.

This update also includes the following lesser changes:

- The **PROFILE SAVE** command now stores ISPF statistics into the directory entries that it creates or updates.
- The **Profile Menuing System's** root panel has been redesigned to (hopefully) make it more easily understood.
- The **LIST PROFILES** report has been redesigned to provide more and (hopefully) clearer information.
- The **IOB** map has been added to the **DMAP** command's **Nicknames Table.** This is a table of dsects for which **DMAP** will (among other things) automatically adjust the zero point to point just past the prefix fields (thus causing the prefix fields themselves to have negative offsets in the map). For more information, see HELP MAPS XDCMAPS.

Help Whatsnew Z22 THingsfixed Maintenance DBC-1811c

The DBC-1811C update adds a significant new facility (Latent Commands) to z/XDC, and it also tweaks numerous other commands.

- **Latent Commands**

  This is a string of commands that is automatically issued under the covers whenever z/XDC receives control from the user program, for example whenever a hook or trap is reached or whenever an abend occurs.

- The default commands string for Assembler programmers is L PSWE;L BEA;L RWREGS.
- For C programmers, the default is a **L VSTACK** command.
- The displays produced will not appear in the Scroll Area, but they will appear in the **Session Log!** This addresses a long requested need to record register and PSW information in the session log. But it does so without unnecessarily cluttering up the Scroll Area.
- The default command can be changed, and it can be saved in your session profile.
- Also whether or not the command reports will appear in the Scroll Area can be turned on or off.
- The report produced by the **LIST LOG** command has been modified to include information about the Latent Commands String.
- Support for a **LATENT=** operand has been added to the **SET LOG** command. This operand can be used both to define a custom Latent Commands String and to turn on or off the display of Latent Commands in the Scroll Area.
- For more information, see HELP FULLSCREEN LOGGING LATENTCOMMANDS.

  Several people have requested this over the years. Perhaps the most persistently has been David Kreiss of BMC. (Sorry it’s taken so long, Dave.)

- **LIST LOG Command**
- The **LIST LOG** report has been revised to make it more readable (hopefully). It also includes information about the current Latent Commands String settings.

- **SET LOG Command**

  The operands syntax for the **SET LOG** command has been revised to convert all operands

to a **KEYWORD=value** format. All relevant doc has been revised accordingly. (But don't worry. Support for the old syntax has been retained. It just won't be documented going forward.)

SCANLOG Command

Quoted string support for the **SCANLOG** command has been fixed. It will now always perform case sensitive or insensitive searches based solely upon the syntax used for providing the search string. The current **SET UPCASE|ASIS setting is now completely ignored by the this command.**

For **case sensitive** searches, support has been added to SCANLOG for a c'string' syntax.

SET PROFILE Command

The SET PROFILE DESCRIPTION='text' command has been fixed so that it now can be used to create a **mixed-case** description for the current profile. (This description shows up in a **LIST PROFILE** report.)

Built-in Help

Significant changes have been made to the following Built-in Help topics:

- HELP COMMANDS LIST LOG
- HELP COMMANDS SCANLOG
- HELP COMMANDS SET LOG
- HELP COMMANDS SET LOG DATASET
- HELP COMMANDS SET LOG SYSOUT
- HELP COMMANDS SET LOG SCROLLAREA [new topic]
- HELP COMMANDS SET LOG MANAGEMENT [was mgmt]
- HELP COMMANDS SYNTAX CHARACTERSTRINGS
- HELP FULLSCREEN LOGGING
- HELP FULLSCREEN LOGGING LATENTCOMMANDS [new topic]
- HELP FULLSCREEN SCROLLING
- HELP PROFILES FACTORYDEFAULTS AWIDE
- HELP PROFILES FACTORYDEFAULTS CWIDE

Because there are so many changes, I have taken the opportunity to rebuild the **PDF** versions of the Help that are posted at our website: colesoft.com/zxdc-release-z2-2

Spelling Corrections

More recently, Ray Mullins of BMC pointed out that I was misspelling the name of a stem variable in my REXX doc. (TEAFLGS should have been TEAFLAGS.) This seemed important, so I fixed it, and I thank him for letting me know.

Our process for rebuilding the PDFs includes a Spell Check, so as always, I took the opportunity to fix all the spelling errors that it reported.

Help Whatsnew Z22 Incompatibilities

Some changes have been made that are incompatible with prior releases of z/XDC. For details, type an **H** at the left to select directly, or use **HELP *NEXT** to proceed sequentially. Use **HELP *FORWARD** to skip.

EXRL - z/XDC has begun to make use of the EXRL instruction within its code. This limits the use of z/XDC to **z10** and newer hardware.

AUTOCMDSTRINGS - The Factory Default is changing pertaining to how you may provide Automatic Command Strings on breakpointing commands (AT TRACE TRAP and friends).

PANELVER - Changes have been made to the **z/XDC Startup Panel.** It is now at the **v6** version level (as shown in the panel's upper right corner).

Older versions of the panel will not be accepted in this release.

But older releases of z/XDC will accept this newer panel without problems.

Help Whatsnew Z22 Incompatibilities Exrl

z/XDC Has Begun to Use EXRL Internally

We are more and more using Relative Immediate Addressing in our new code, and now we have begin to use the **EXRL** instruction as well.

Unfortunately, EXRL was a latecomer to the Relative Immediate party. IBM did not implement it until 2008 when they introduced the **z10 EC** processor.

For a list of IBM processors and their introduction dates, search for **wiki List of IBM products.**

If you are running on a **z9** or older machine, then **you will not be able to upgrade to z/XDC z2.2!**

We will continue to maintain our older **z2.1** release until further notice.

Help Whatsnew Z22 Incompatibilities Autocmdstrings

Factory Default is Changing Pertaining to Automatic Command Strings

In the prior release of z/XDC, a new syntax was introduced for appending automatic command strings to breakpointing commands. Briefly, the new syntax allowed automatic command strings to be appended to **AT, TRAP** and other breakpointing commands via quoted strings (rather than via leading colons). See the **New Automatic Commands Syntax** section within the HELP WHATSNEW Z21 SYNTAXCHANGES topic for details.

However:
- The new syntax (quoted command strings) remained optional,
- And the factory default remained as colon delimited command strings.

Well, now the factory default is changing. Going forward, quoted command strings will be the default that will be set by the PROFILE RESET command.

The older colon delimited syntax will continue to be supported. It just no longer will be the factory default.

Why We're Doing This

This change was needed in order to support multiple address expression parsers and the ability to **tag** individual address expressions with an override for which parser to use. For more information, see HELP ADDRESSING PARSERS.

Tagged address expressions are used internally by our new c/XDC support for coercing proper parsing even when global settings favor assembler syntax parsing over C syntax.

Unfortunately, the older colon delimited automatic commands syntax is incompatible with tagging syntax, so when colon delimited automatic commands are permitted, tagging is prohibited, and that causes c/XDC's internally generated address expressions to fail.

Propagating to Personal Profiles

Warning! If you have personal profiles (named or default), this change to the factory default will not affect those. So if you need to make this change to those profiles, use the following sequence of commands:

PROFILE READ name
This loads in a fresh copy of a named profile. (For your personal default profile, use the name **XDC.)**

LIST TRACE
This reports the profile's settings relating to breakpoints.

SET TRACE QUOTEONLY
LIST TRACE
If necessary, use the SET command to change COLONANDQUOTE setting to QUOTEONLY. Then use the LIST command again to view the change.

PROFILE SAVE

This command writes the updated profile back to disk.

Wash, rinse, repeat for all of your profiles. (Use LIST PROFILES ALL to see what they are.)