z/XDC®
RELEASE GUIDE

z/XDC® Release z2.1
for z/OS

David B. Cole

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

Generated 09/04/2016, 2:13pm

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 The Product may not be published, reproduced, distributed, or made available to third parties for any purpose without the expressed written permission of Owner or Licensor.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, and any warranty as to accuracy.

WARNING! z/XDC® is a powerful tool for dynamically locating and correcting malfunctions in actively executing user programs and operating system 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
Home Page:	www.colesoft.com

Our Technical Support contacts are:

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

Our Customer Services contacts are:

Phone:	540-456-8210
E-Mail:	support@colesoft.com
Home Page:	www.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. 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 Customers.

TRADEMARKS

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

Extended Debugging Controller®, **XDC**®, and **z/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 is 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® z2.1 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® z2.1 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 hyperlinks to pursue the specific information that is of interest to you.

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

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

For z/XDC command information, turn to the **z/XDC® Commands** manual. Start with the basic commands. The DISPLAY, 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. 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 Z21

z/XDC release **z2.1** includes all maintenance fixes to release **z1.13** and the following additional changes. For detailed information, type an **S** at the left to select directly, or use **HELP *NEXT** to proceed sequentially. Use **HELP *FORWARD** to skip.

↕	**AUTOMAPPING**	- z/XDC can now automatically load maps for modules and csects whenever it first detects that execution has reached a csect.
↕	**AUTOSCROLLING**	- Certain improvements have been made to the management and methods of maintaining a persistent scroll position within Watch Windows.
↕	**BEA**	- A **LIST BEA** command has been added to display the most recent **Breaking Event Address.**
↕	**BUILTINHELP**	- Significant new and changed Built-in Help topics are summarized here.
↕	**COMMANDS**	- Several commands have been added or changed. A few may have been removed.
↕	**EQUATES**	- Some new **built-in** and **automatic equates** have been created.
↕	**FRR**	- An improvement to support for running z/XDC as an FRR.
↕	**HELPCROSSLINKS**	- Cross topic links, present in the Built-in Help, are now also present in the PDF manuals generated from the Built-in Help.
↕	**HOOKREWRITE**	- Hook processing has been entirely rewritten. Hooks can now be set in PC routines, SRB routines and other SVC-unfriendly places.
↕	**MAPLIBSLIST**	- Default profiles no longer define an initial default MAPLIBS list.
↕	**ONLINEHELP**	- "Online Help" is now referred to as **Built-in Help.**
↕	**OPERANDVALUES**	- There is a new command that can be used to display the current values of all the operands referenced by the current (or prior) machine instruction.
↕	**PARSEORDER**	- (For c/XDC) This is a new concept that will be needed once our c/XDC development work is released. This concept is needed to cope with the fact that different programming languages require different syntaxes for

parsing its variables. So when, for example, you use a variable in a FORMAT command's address expression, you can use **Parser Ordering** to control which Language Parsers to use and in what order they are invoked for parsing said address expression.

- **PROFILEMENUINGSYSTEM** - Many new settings have been added to Session Profiles. Also, navigation through the Menuing System has been improved.
- **PROFILERESETSANDDEFAULTS** - The PROFILE RESET command can now select among several Factory Default profiles.
- **SUPPORTIMPROVEMENTS** - We are expanding our Online Presence in ways that we hope are helpful to our customers. We now have training videos (on YouTube) and a support forum (on LinkedIn).
- **SYNTAXCHANGES** - The syntax for providing automatic commands on breakpointing commands is changing!
- **TAGGEDADDRESSEXPRESSIONS** - (For c/XDC) This is a method to force a particular address expression to be parsed by a particular Language Parser. It overrides the current Parse Order.
- **WATCHWINDOWDEFAULT** - The Factory Default Watch Window has changed.

- **MISCELLANEOUS** - Some of z/XDC's **minor changes** are described here.
- **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 Z21 AUTOMapping

z/XDC can now automatically build load module maps, csect maps and source image maps whenever it detects that execution has arrived in a new load module or csect for the first time.

Previously, if maps were wanted, the **MAP** command always had to be used to load them. Now however, whenever z/XDC receives control, it will check the modules into which the error level and retry level PSWs point. If those locations have not yet been mapped, and if z/XDC has not previously tried to map them, then it will do so now.

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

For more information, see HELP MAPS AUTOMAPPING.

Help Whatsnew Z21 AUTOSCrolling

Watch Windows are designed to contain persistent displays of whatever you care to show (as selected by the displaying-type command strings you choose to enter onto their command lines).

It sometimes is the case that you would like to issue a command that generates significantly more messages than can possibly be displayed in any Watch Window, but you are interested in persistently viewing only a small part of that information, located some knowable distance either from the top **or from the bottom** of the generated display.

In order to meet this need, we have significantly improved the code that manages the positional stability of a Watch Window display. Previously, logic to maintain the scroll position was kind of a mess. That has now all been cleaned up. Now when you scroll a Watch Window's content downwards, you can expect the following:

- When you use scrolling commands to arrive at a particular position, a **DOWN** command is generated to represent a **merge** of the net effect of those scrolling actions.

- The generated **DOWN** command is appended to the Watch Window's command line for two reasons:
  - Its presence on the command line is the means by which the display's position is reestablished every time you press ENTER.
  - Being able to see the merged **DOWN** command is what reminds you that autoscrolling is occurring. Also it makes you aware of how far down that autoscrolling distance is.

- It is now possible to establish an autoscrolling position, not just relative to the top of the display, but also **relative to the bottom.** You can now say **DOWN MAX-nnn,** and when you do, autoscrolling will maintain that display position nnn lines relative to the bottom of the generated display.

Autoscrolling pertains only to Watch Windows. It does not occur in the Working Window.

For more information, see HELP FULLSCREEN SCROLLING.

In conjunction with this support of a **MAX-nnn** "adjustment" operand for the DOWN command, it was easy to extend that adjustment operand support to all operands of both the DOWN and UP commands.

So now, for example, a command such as **UP CMD+5** is perfectly legal for the Working Window. It positions the display to start five messages prior to the next preceding logged command string.

Help Whatsnew Z21 BEa

- A new command, **LIST BEA,** has been added that displays the most recent **Breaking Event Address** (BEA). See HELP COMMANDS LIST BEA for more information.
- In addition, a display of the BEA was added to the **LIST BRANCHES** command in z/XDC z1.13 via maintenance Z1D-1210C.
- Also, the **LIST FEATURES** command now shows whether or not your mainframe has BEA support implemented in its hardware.
- Finally, a **LIST BEA** command has been added to the factory default Watch Window that is loaded by the **PROFILE RESET** commands.

The **Breaking Event Address** is IBM-speak for the most recently executed branch-type instruction that (in this case) was executed prior to the abend or breakpoint by which z/XDC received control. **Branch-type instruction** is Dave Cole speak for any instruction that can change the execution path. This includes branching instructions, jumping instructions, LPSW and others.

Sometimes, the BEA will point to a **LPSW** instruction somewhere in the System Nucleus. But usually it will point to a branch-type instruction within your own code. So **LIST BEA** can be very helpful if you need to know how you got to where you are.

Help Whatsnew Z21 BUiltinhelp

PDF Improvement

- A major improvement has been made to the PDF versions of the Built-in Help manuals. Now, the cross referencing hyperlinks that exist in the Built-in Help are finally also activated in the PDFs. See HELP WHATSNEW Z21 HELPCROSSLINKS for more information.

Broken Crosslinks Fixed

Hundreds of broken crosslinks (some broken for decades) have **finally** been fixed.

Recently, Frank wrote a tool for me that pretty easily finds broken crosslinks. Thank you Frank.

Organizational Changes

The following structural changes have been made to the Built-in Help:

- The topic named HELP ADDRESSING ASM has been moved TO **HELP ADDRESSING PARSERS ASM.**
- The topic named HELP CDF has been moved TO **HELP XDCSRVER CDF.**
- The topic named HELP DEBUGGING HOOKS has been moved TO **HELP HOOKS.**
- The topic named HELP LINECMDS has been renamed to to **HELP SHORTCUTCOMMANDS.**
- All of the subtopics of HELP SCRIPTS have been moved to be subtopics of **HELP SCRIPTS OURSCRIPTS.**

These changes affect, of course, all references to the affected subtopics.

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

HELP ADDRESSING PARSERS
Due to the impending advent of C/C++ support, there is a need to support a new (alternative) syntax for parsing address expressions, one that is compatible with C/C++ language constructs.

So now...
- It is possible to establish a default parser for address expressions (HELP COMMANDS SET PARSEORDER).
- It is possible, for individual address expressions, to override the default parser via **tagging** (HELP WHATSNEW Z21 TAGGEDADDRESSEXPRESSIONS),

HELP ADDRESSING PARSERS (and its several subtopics) is a new topic that discusses address expressions in a world of multiple parsers.

HELP DDNAMES
This topic has been broken down into 29 smaller topics with one subtopic for each ddname.

HELP DEBUGGING LOSTLOCKS
Information about the consequences of and coping with Lost Locks has been extracted from HELP DEBUGGING FRR and made into its own topic.

HELP HOOKS
HELP COMMANDS HOOKS
Due to the redesign of z/XDC's Hook Processing Services, these two topics (and their subtopics) have been massively rewritten. It would be well worth taking the time to reread them.

HELP FULLSCREEN SCROLLING
This topic and its new subtopic have been rewritten to clarify:
- The distinctions between scrolling through Watch Windows vs. scrolling through the Working Window.
- The relationship betweeen scrolling through the Working Window's recent history vs. scrolling through storage.

HELP FULLSCREEN WINDOWS
This topic and its new subtopics have been rewritten to better describe the Working Window vs. Watch Windows, their similarities, their differences and their best uses.

HELP PROFILES FACTORYDEFAULTS
This is a new set of topics that describe the several Factory Default Profiles that now come with z/XDC.

HELP SCRIPTS
This topic has been reorganized and rewritten, and two new subtopics have been created:

HELP SCRIPTS RYOSCRIPTS
This is a new subtopic that contains comprehensive information about writing your own Command Scripts.

HELP SCRIPTS OURSCRIPTS
This is a new subtopic that focuses exclusively on the prewritten scripts that come with z/XDC.

HELP SCRIPTS OURSCRIPTS scriptnames
All of the HELP panels documenting the prewritten scripts have been moved to be subtopics of HELP SCRIPTS OURSCRIPTS.

HELP SUPPORT ZONLINE
This is a new set of topics describing our **Online Presence** for z/XDC. In addition to our home page **(colesoft.com),** we now also have a **Facebook** page, a **LinkedIn** discussion group and a **YouTube** channel for training videos.

HELP SUPPORT ZMANUALS
HELP HELP ZMANUALS
These are two new topics that briefly describe the PDF format documentation that's available for z/XDC and from where they can be downloaded.

HELP XDCSRVER
This is a new topic that describes the **XDC/Server** task, what it does and which z/XDC facilities require it in order to function.

Also, the old HELP CDF topic has been moved to be a subtopic under HELP XDCSRVER.

Help Whatsnew Z21 COmmands

The following commands are either new to z/XDC z2.1, changed in z/XDC z2.1 or deleted from z/XDC z2.1.

All breakpointings Commands - Syntax Improvements
ADEFERRED,
AT
ATX
TDEFERRED,
TRACE
TRAP
There are two major changes to the syntax for these commands:

- 1st, The positional placement requirements have been relaxed for the specialty operands:
  - =familyid#
  - =(conditional expressions)
  - 'automatic commands' (i.e. the new quote delimited format for automatic commands)

  These operands can now be placed anywhere within the command... before, after or in between other operands.

  (Note that positional requirements for the old **colon** delimited automatic commands format have **not** been relaxed. When supported, they still must occur at the end of the command.)

- 2nd, The syntax for automatic commands is changing:
  - From **:cmd:cmd:cmd**
  - To **'cmd;cmd;cmd'**

  For detailed information, see HELP WHATSNEW Z21 SYNTAXCHANGES.

DELETE
A new operand, **SILENT=YES,** can be used on any type of DELETE command to suppress

progress and completion messages.

SILENT=YES has been implemented primarily for use by the internally issued DELETE EQUATES commands that are used by those commands (LIST TASKS, LIST RBS, etc.) that generate Automatic Equates (TCB#n, RB#n, etc. See HELP EQUATES BUILTIN AUTOMATIC.) Due to another change in z2.1, without a SILENT=YES operand, the displays produced by such commands would include a message reporting how many equates had been deleted.

DELETE MAPLIBS

The DELETE command's new **SILENT=YES** operand is functionally redundant with the DELETE MAPLIBS' previously existing **[NO]SHOW** and **QUIET** operands. Accordingly, support for these older operands has been deprecated. They will continue to be supported, but their documentation is being removed.

DELETE PROXYTASKS

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

DOWN

The various operand values that this command accepts all can be modified by "adjustment" values. DOWN **MAX-7,** for example, will position the scroll area to seven lines above the DOWN MAX position.

GETMAIN

Some new operands have been implemented that allow you to control both the key of the allocated storage and its residency (24 bit vs. 31 bit).

Also, new keyword= type operands have been added to provide different ways to specify the subpool in which the requested storage is to be obtained.

HOOK

z/XDC's implementation of Hook Support has been entirely rewritten so that it can be used in PC routines, SRB routines and other SVC-unfriendly places. Also, new operands have been created to allow the user to tell a hook what its execution environment will be, and to target common storage hooks to specific address spaces.

LIST AUTOMAP

This command displays z/XDC's current **Automapping** settings.

LIST BEA

This command has been added to display the most recent **Breaking Event Address** in just a one line display.

LIST BRANCHES

This command was updated in release z1.13 by maintenance (Z1D-1210C) to include a display of the most recent **Breaking Event Address** (BEA). See HELP COMMANDS LIST BRANCHES for more information.

LIST FEATURES

The following additional Features are now shown:

- Whether or not your mainframe has **BEA** support implemented in its hardware.
- Whether or not your z/OS retains Lost Locks Information in the SDWA for ESTAE-type recovery routines.

LIST LOCKS

Starting in z/OS R1.12, for ESTAE-type recovery routines, the Recovery Termination Manager (RTM) began providing partial information in the SDWA about what locks were held at abend time. (Previously, such information was available only for FRR-type recovery.) So I have finally gotten around to updating the LIST LOCKS command to display that information even when z/XDC is running as an ESTAE[X].

LIST OPERANDS

This command displays the current storage operand values for a given machine instruction.

LIST PARSEORDER

This command displays the order in which language-specific parsers will be called for resolving a variable name.

LIST READ

The report produced by this command has been redesigned to make it more readable. Also, it displays new information:

- The current default scripts library.

LIST TRACE

The report produced by this command has been redesigned to make it more readable. Also, it displays new information:

- The current Automatic Commands syntax setting.
- Whether or not tagged address expressions are permitted.

LOCATE

A minor change: LOCATE's **nnn** operand can now be prefixed with a **#.** For example: **LOCATE #10** will shift the Working Window's Scroll Area up or down to bring the tenth issued command string into view.

This change was made to make LOCATE's syntax similar to the **#nnn** support already present on the **UP** and **DOWN** commands.

PROFILE RESET

This command now accepts an operand that allows you to select which (among several) Factory Default profile you want to load.

It no longer accepts operands that allow you to assign a new name to the profile upon being loaded.

SET AUTOMAP

This command allows you to change z/XDC's current **Automapping** settings. These settings can be saved in your Session Profile.

SET PARSEORDER

When resolving variable names, different programming languages require different syntaxes for parsing its variables. this command allows you to specify:

- Which language parsers will be called,
- The order in which they will be called,
- And which parser will be used to generate an error message should all parsers fail.

Parse Ordering is a capability that will become useful once we publish our upcoming c/XDC support.

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 PSW
SET PSWE

New support has been added to the **SET PSW** command that (authorization permitting) allows you to change the IO mask, the EXT mask and the PROG masks in the retry level PSW[E]. (Note, this change was retrofitted to z/XDC's prior release via maintenance Z1D-1303A.)

SET TRACE
New operands **(QUOTEONLY** and **COLONANDQUOTE)** have been implemented to allow you to control the syntax that z/XDC will accept for adding Automatic Commands to breakpointing commands (AT, TRAP, TRACE and the like). See HELP WHATSNEW Z21 SYNTAXCHANGES for more information.

UP
The various operand values that this command accepts all can be modified by "adjustment" values. UP **#10+7,** for example, will position the scroll area to seven lines above the UP #10 position.

USING
When you use the requeuing form of the USING command, a message is now displayed confirming that the dsect has been requeued to the top of the dsects search order. Example:

```
USING TCB TCB
TCB REQUEUED TO THE TOP OF THE SEARCH ORDER
```

Help Whatsnew Z21 Equates

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

@BEA
This labels the most recent BEA (Breaking Event Address) location for your program. This is the same location displayed by the **LIST BEA** command.

For more information, see HELP WHATSNEW Z21 BEA.

@PRIOR
This labels the retry level resume address that existed the **prior time** that z/XDC received control from the user program. This points to the machine instruction whose operands are displayed by a **LIST OPERANDS PRIOR** command.

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

Help Whatsnew Z21 Frr

- Starting with update **Z21-1509H,** 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 Z21 HElpcrosslinks

PDF Improvement

It has long been the case that z/XDC's Built-in Help has had cross topic links that you can use to navigate to topics related to what you are currently reading. (Believe it or not, z/XDC has had this support long before the term **hyperlink** came into common use. But I digress...)

But until now, we have not been able to carry these links forward into the PDF manuals that we generate from the Built-in Help. Well now we can!

And now we do! And we're very pleased about this. (Thank you Frank.) The presence of cross-links will help make the PDF manuals far more powerful for those studying to learn more about z/XDC.

Broken Crosslinks Fixed

Hundreds of broken crosslinks (some broken for decades) have **finally** been fixed.

Recently, Frank wrote a tool for me that pretty easily finds broken crosslinks. Again, thank you Frank.

Help Whatsnew Z21 HOokrewrite

z/XDC's hook processing support has been entirely rewritten with the goal of making it usable in code where SVCs cannot be issued. (Although the HOOK SVC remains for legacy support.) So now both dynamic and static hooks can be set into code that is:

- A PC routine
- An SRB routine
- FRR protected
- Locked
- Disabled
- Generally any place without regard to whether or not it is SVC friendly.

For more information, see HELP HOOKS.

For static hooks, the #XDCHOOK macro has been rewritten to remove dependence upon the HOOK SVC. However, z/XDC's HOOK SVC continues to be supported so that existing code containing #XDCHOOK macros does not have to be reassembled. But once such code is assembled, the new #XDCHOOK macro will be used. For more information (including **Legacy Support** details), see HELP HOOKS STATIC.

For dynamic hooks, the footprint has been reduced from eight bytes to just six, and the hook itself has been changed from an SVC (followed by data) to just a JLU instruction. For more information, see HELP HOOKS DYNAMIC.

Generally, the new Hook Support works without having to change its authority level, so it does not have to use either SVCs or PCs to do its work. There is, however, one case where a PC routine needs to be called. That's when a hook has been placed (by z/XDC while running authorized) into non-authorized code that resides in store protected key-0 code. (This can happen when a program is either reentrant [RENT] or refreshable [REFR].) In this one case (and this one case only), when execution actually reaches the hook, a PC routine needs to be called in order for z/XDC to remove the hook and restore the original code.

In z/OS, PC routines must be owned by Server Tasks. In z/XDC's case, the Hook Support PC routine is owned by the **XDCSRVER** task. So when the PC routine is needed, XDCSRVER must be up and running. Otherwise, hook removal will fail, and message DBC548T will be issued. For more information, see HELP XDCSRVER.

For the HOOK command, z/XDC's execution environment at the time that you issue the command is not always a predictor of the execution environment that will exist when the hook is actually reached by execution. For that reason, the HOOK command now has some new operands that you can use to tell a new hook what its execution environment actually will be. For more information, see HELP COMMANDS HOOK.

Also, there are new HOOK command operands that allow you to more easily target common storage hooks to particular address spaces. Previously, users had to control this targeting by setting Foreign Address Space Mode (FASM) prior to issuing the HOOK command. Now it can be done with just a command operand.

Another command operand allows you to target a common storage hook to a particular address space by name (not just by ASID). This means that address spaces can be targeted before they even exist.

Finally, Hook Processing's handling of [E]SPIEs has changed. Previously, if an [E]SPIE intercepted Program Interrupt Codes (PICs) 0001 thru 0007, the [E]SPIE would be changed to exclude those particular codes from being intercepted. Now, however,

when Hook Processing detects such an [E]SPIE, it cancels the [E]SPIE entirely.

Help Whatsnew Z21 MAplibslist

Default MAPLIBS List no Longer Provided

For the past several releases, the Factory Default Profile had provided an initial default MAPLIBS list specifying the library name: DBCOLE.XDCZ21.XDCADATA.

Unfortunately, a larger number of customers than I expected routinely install z/XDC under high level qualifiers other than "DBCOLE.". Well, z/XDC does not have any way to know its libraries' HLQ, so having a hard-coded DBCOLE.--- library in the default MAPLIBS list was causing more trouble than it was worth: Users were seeing **DBC892E** messages every time they started a debugging session! This would continue until they issued the necessary **SET MAPLIBS** commands to fix the default MAPLIBS list. That's something a beginner user in particular is not going to know how to do right away.

Accordingly, to alleviate this problem, I've had to remove the default MAPLIBS list definition both from all Factory Default Profiles and from all Installation Default Profiles.

The need for a pre-created MAPLIBS list has been reduced by the introduction (in release z1.12) of the **ADATALIB=** operand for the **MAP** command. That not only allows you to explicitly identify an ADATA library at MAP time, but it automatically adds that library name to your currently active MAPLIBS list.

You can, of course, continue to create your own MAPLIBS lists and set one of them up as your personal default. It's just that z/XDC will no longer offer a default list of its own.

Help Whatsnew Z21 ONlinehelp

For decades, z/XDC's internal help has been referred to as "Online Help". Well once upon a time, that term may have been good enough, but nowadays, in the age of the Internet, the term Online Help has come to mean something that is on the Internet.

But z/XDC's internal help is not on the Internet, it is built in. So going forward we will use the term **Built-in Help** instead of "Online Help" when referring to our internal help.

This nomenclature change has been made throughout the Online Help... Oops I mean throughout the Built-in Help.

For discussions of other changes made to the Built-in Help, see HELP WHATSNEW Z21 BUILTINHELP.

Help Whatsnew Z21 OPerandvalues

There is a new command that can be used to display the current values of all the storage operands referenced by the current or prior (or other) machine instruction. That command is **LIST OPERANDS.**

This command displays the current values of all of the storage operands of a machine instruction. You just give it the address of the instruction to analyze, and it discerns all of that instruction's storage operands and displays their current values.

LIST OPERANDS can be directed:

- Towards the instruction pointed to by the retry level PSW. In this case, the values displayed are those that exist **prior to** the execution of said instruction.

- Towards the instruction that was about to be executed the **prior time** that z/XDC received control. In this case, for single stepping, this would show the values of that instruction's, operands **just after** it was executed.

- Towards any instruction anywhere in the system. In this case, the displayed information is **highly likely to be wrong!** Whether it is right or wrong depends upon whether the data and execution environments are the same or different between the point in time when said instruction would be executed and the current retry level environment. That, of course, is something that z/XDC **cannot predict.**

Note, the more remote an instruction is from the current point of execution, the more likely it is that the displayed operands will be wrong. For more information, see HELP COMMANDS LIST OPERANDS.

Help Whatsnew Z21 PArseorder

The syntax rules for variables used in different languages differ with each other in conflicting ways. To cope with this, we have developed a concept we call **Parse Ordering** by which a user can select the Language Parsers to be used by z/XDC when parsing address expressions.

z/XDC now has the ability to support multiple Language Parsers, one for each supported programming language. z/XDC also now has **Parser Management.** When parsing an address expression that contains a variable name, Parser Management allows you, the user, to control:
- (a) Which Language Parsers are called for parsing the variable,
- (b) In what order they are called,
- (c) And which Language Parser gets to issue an error message should all parsers fail.

For more information about Parser Management, see HELP ADDRESSING PARSERS PARSEORDER.

Another way in which the parsing of variables can be controlled is through the use of **tagged** address expressions. This is a mechanism by which you can force a

particular address expression to be parsed by a particular Language Parser regardless of the current Parse Order. For more information, see HELP ADDRESSING PARSERS TAGS.

Help Whatsnew Z21 PROFILEMenuingsystem

Certain improvements have been made regarding the PF keys used for navigating within the Profile Menuing System:

- Certain PF keys (PF3, PF4 and PF5) have specific, hard-coded functions within the Profile Menuing System **regardless** of how they are defined by the **KEYS** command outside of the Profile Menuing System. Those meanings are documented in HELP PROFILE MENU.

- Previously, for some of the Profile Menuing System panels, some of those PF keys were behaving differently from their defined functions. That has now been cleaned up.

- The behavior of PF4 (CANCEL) has been changed. Previously, it would cancel all changes but remain within the panel. Now however, it still cancels all changes, but it also closes the panel and pops out to the next higher level. This brings the behavior of PF4 into conformity with its behavior in all other specialty panels.

For more information, see:

- **HELP PROFILES MENU** (the root of all information about the Profile Menuing System)

Help Whatsnew Z21 PROFILEResetsanddefaults

There are now 2 Reset Profiles

Until now, z/XDC provided only one one-size-fits-all Factory Default profile. Now, there are two. (And soon there will be four.)

With the eventual advent of **c/XDC,** it became clear that the default profile used for Assembler debugging simply was not appropriate for C debugging. So we needed to provide two default profiles.

But then I realized that developing support for two default profiles was a good opportunity for dealing with similar lingering issue: Default profiles suitable for 80-column display terminals were not suitable for wider terminals. So presto! Now we are about to have **four** Factory Default profiles. (But for the moment, we'll start with just two.)

The **PROFILE RESET** command now accepts an operand for choosing which factory default profile is to be loaded. Further, the operand can be **generic** such that the

particular Factory Default profile that is loaded is determined **dynamically** by the PROFILE RESET command based upon whether or not c/XDC is being used and upon whether the terminal is wide or narrow. For more information, see HELP COMMANDS PROFILE RESET.

Default Scripts Library

Previously, the default Scripts library (set by the **PROFILE RESET** command) was **none.** Now it is **DBCOLE.XDCZ21.XDCCMDS.**

Further, the default scripts library is now displayed by the **LIST READ** command.

Help Whatsnew Z21 SUpportimprovements

We have increased (and will continue to increase) our Online Presence in ways that will be more supportive to our customers. In addition to our home web site (colesoft.com), we now also have:

- A Facebook page (at **facebook.com/colesoftware)** for random information about ColeSoft people and activities

- A LinkedIn group (named **x/XDC Users Group)** for product announcements, problem reporting and discussion, how-to tips and tricks, pretty much everything that's important to z/XDC customers

- A YouTube Channel (at **youtube/colesoftware)** where you will find our training videos for z/XDC. Currently, there are eleven videos, but we will be making more.

For more information, see HELP SUPPORT ZONLINE.

Help Whatsnew Z21 SYntaxchanges

There are several syntax changes relating to the breakpointing commands.

Counting-Type Conditional Expressions

When a single command is used to set multiple traps, and that command includes a counting-type conditional expression, the hit limit created by the count now applies to each trap individually. (Previously, it applied to the breakpoints collectively, i.e. it applied to the sum total that all of the connected breakpoints were hit.)

Compound Conditional Expressions

Conditional expressions must now always be fully enclosed as a whole, within a pair of parentheses. (Previously, it was sometimes sufficient that such expressions started with an open parenthesis.)

Operand Positioning Requirements Relaxed

The positional placement requirements have been relaxed for all breakpoint command operands. Previously, the =familyid, (conditionalexpression) and 'automaticcommands' operands had specific positioning requirements with respect to each other and to other operands in general. Those requirements no longer exist.

Now, all breakpoint command operands can occur anywhere within the command. Positioning requirements simply do not exist.

Conditional Expression Evaluation

When:

- The **XADDR, XASID** or **XALET** built-in function is used within a breakpoint's conditional expression,
- And its resolved value is to be compared against a given hex constant,

The comparison is now done **right-aligned.** This means that no more than one leading zero needs to be prefixed to the constant in order for the comparison to work correctly. Notes:

- For all other comparisons occurring within conditional expressions against other kinds of objects (PSWs, registers, storage), a left-aligned comparison continues to be done.
- One leading zero may still need to be prefixed to the constant so that the constant contains an even number of digits.
- The width of the comparison continues to be governed by the byte-width of the constant. So any part of the resolved value that is to the left of the comparison is ignored.
- Extra pairs of leading zeros are OK and have the effect that the corresponding portion of the resolved value are compared against zeros.
- So existing command scripts that have leading zeros prefixed to the constant do not have to be changed.

For more information, see HELP COMMANDS SYNTAX BREAKPOINTS CONDITIONS COMPARISONS.

New Associated Commands Syntax

The syntax for assigning automatic commands to breakpoints is changing.

The old syntax was... To append the commands to a breakpointing command, you had to delimit the appended commands with colons (:). Example:

```
              |                      |
              V                      V
    AT .TOPLOOP:ALARM 'TOPLOOP reached':WHERE
```

The **new syntax** is:

(1) - Enclose the automatic commands within quotes (')
(2) - Double up any included quotes ('')
(3) - Append the quoted string to the breakpointing command separated by a blank ( ) or comma (,)
(4) - And within the quoted string, delimit the individual automatic commands from each other by semicolons (;) (not colons)

Example:

```
              ||      ||              |||     |
              VV      VV              VVV     V
    AT .TOPLOOP 'ALARM ''TOPLOOP reached'';WHERE'
```

See HELP COMMANDS SYNTAX BREAKPOINTS AUTOCMDS for more information.

This change was made to accommodate the need to use colons for another purpose - the delimiting of tags from address expressions. Under the old syntax, colons had to be reserved **exclusively** as automatic command delimiters. The problem is, colons are too useful to be wasted for that single esoteric purpose. So in the end, I've had to bite the bullet and make this change.

However, the old syntax is not **yet** gone, only deprecated. Its documentation has been largely removed, but for the time being, z/XDC will continue to accept the syntax (at least until the next release).

BUT! There *is* a catch! It remains true that when colons are accepted in the automatic commands syntax, they **cannot** be used for any other purpose... such as delimiting tags in tagged address expressions. (See HELP ADDRESSING PARSERS TAGS.)

So I have had to implement support for turning on or off z/XDC's recognition of colon-delimited automatic commands. The commands involved are:

SET TRACE QUOTEONLY/COLONANDQUOTE
This controls whether z/XDC recognizes **both** colon-delimited and quote-delimited automatic command strings or **just** quote-delimited commands.

LIST TRACE
A display has been added to this command's report that shows the current QUOTEONLY vs. COLONANDQUOTE status. Also shown is whether or not tagged address expressions are permitted.

PROFILE
The current QUOTEONLY vs. COLONANDQUOTE setting can be displayed and changed in the Profile Menuing System, and it can be saved in your personal Session Profile.

For now, the factory default setting is COLONANDQUOTE, but that **will** change.

Help Whatsnew Z21 TAggedaddressexpressions

In order to support multiple programming languages, z/XDC needs to support multiple, language specific Parsers which it calls in a user specified Parsing Order when it needs to parse a variable name.

Generally, when the context is unambiguous, z/XDC will always use the parser that is right for the context. Examples: when the **LIST RBS, LIST LSTACK,** LIST whatever commands are given an address expression operand, the **ASM** parser will always be used to parse that expression. Likewise, the operands of a **LIST HLLVARIABLES** command will always be parsed with the **CEE** parser.

But when using generic storage referencing commands such as the **FORMAT, SHOW** and **DISPLAY** commands, that's when the Parse Order matters, and that's when you might want to **tag** an address expression to override the global default. For more information, see HELP ADDRESSING PARSERS TAGS.

Help Whatsnew Z21 Watchwindowdefault

The Factory Default Watch Window's command string has changed...
- If your TSO workstation terminal's geometry has 100 or more columns,
- And if you issue a **PROFILE RESET** command,

Then a display layout will be loaded whose Watch Window's command string includes a **LIST BEA** command to show the address of the most recent branch-type instruction executed by your program (or by a support routine) prior to the abend or breakpoint by which z/XDC received control. For more information, see HELP WHATSNEW Z21 BEA.

If you are on a terminal having less than 100 display columns, then a **PROFILE RESET** command loads a profile that does not include a **LIST BEA** command in the Watch Window's command string. There is nothing, however, to prevent you from adding it yourself and then saving that profile as your own personal default.

Help Whatsnew Z21 MIscellaneous

Several additional small changes have been made to z/XDC in this release. Most of them are listed here.

AXDCIS Function Subroutine

Usually when you wish to debug a program with z/XDC, you would invoke it by its normal name: **XDC.** However, if you are, for example, trying out a beta release of z/XDC, or if you are using your own personal copy of z/XDC, that copy will have been

renamed to a different three-letter name other than "XDC". Such a name is called a **clone name.** See HELP XDCCLONES for more information.

Sometimes, the program that is to be debugged may need to know the clone name of the z/XDC that is to do the debugging. So now we provide a function subroutine named **AXDCIS** that will dynamically determine the correct z/XDC clone name and return that name to its caller as the function's value. It does this by scanning the **TIOT** looking for an allocation named **//XDCISxxx** DD DUMMY. The **xxx** part of the ddname is returned as the clone name. See HELP DDNAMES XDCIS for more information.

AXDCIS can be called by Assembler programs. For more information, see HELP XDCCLONES AXDCIS.

DBCMAPS Script

This script has been revised as follows:
- New commentary has been added to provide better usage information.
- DELETE commands have been added so that the script is now rerunnable.

Default Terminal Colors

The factory default display colors setting has changed. Previously, it was RWCY. Now, it is **RWMY** (Red White Magenta Yellow).

With the older setting, the contrast between Cyan and White was pretty low. I believe this new setting improves the contrasts between the various field types.

For more information, see HELP COMMANDS SET COLORS.

DSECT Requeuing

The Old News: When you issue a USING command of the form **USING name samename,** the dsect map is requeued to the top of the Dsects Search Order without affecting the dsect's base address or basing method. See HELP COMMANDS USING for more information.

The New News: Now when you do this, an information message will be displayed telling you that the dsect has been requeued. (Previously, no message whatsoever was displayed.)

Help Whatsnew Z21 THingsfixed

Things Fixed

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 since product release, see HELP * MAINTENANCE.)

DS 0H Formatting Issues

When all of the following were true, formatted displays of code would unexpectedly switch to data formatting starting at the point that a **DS 0H** occurred:

- A csect map has been loaded by the DMAP command as a dsect and then assigned to storage via a USING command.
- The code in the csect map contains **label DS 0H** statements (instead of, for example, "label EQU *").
- The map was built from SYM data.
- A normal data mapping dsect has also been assigned to overlay the same storage containing a **DS 0H** statement.

Normally, a **DS 0H** statement should not affect the FORMAT command's instruction-vs-data decisions, but in this case, it did.

This has now been fixed.

Dsect Maps Loaded with Negative Zero Points

When the ZEROPOINT= operand was used on a DMAP command, and the value of that operand was negative, then when a USING command was issued to assign that map to storage, the fields within the map would not display. This has now been fixed.

Message Construction Buffer Overflow

z/XDC's internal message construction buffer was too short for some messages. This would lead to those messages being truncated and appended with the text:

    <DBC998E - MSGBUF OVERFLOW>

The construction buffer has been significantly enlarged, so this message should not reoccur.

PF Keys 1 and 13 Fixed in Factory Default Profiles

The factory default settings for PF keys 1 and 13 are now as follows:

- PFK1: SET SCREEN DELETE
- PFK13: SET SCREEN CREATE

Previously, these command strings had a dash appended to them. (See HELP FULLSCREEN PFKEYS HYPHEN for why.) Unfortunately, this caused unnecessary usage errors when command lines were non-empty. Removing the dash fixes those problems.

Scroll-Down Excess Lines Issue

Frequently, the FORMAT command and the WHERE command and especially the **DOWN** command would produce more display lines than would fit within the display window. Then a subsequent DOWN command would display those few excess lines, and it would take an additional DOWN command to proceed to displaying the next chunk of storage. This has now been fixed.

XADDR Built-in Function Fixed

When the **XADDR** built-in function was used within a breakpoint's conditional expression, it would never resolve **TRUE.** This has been fixed.

Post-Release Maintenance

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

Help Whatsnew Z21 THingsfixed Maintenance

The following updates have been APPLY'd to z/XDC z2.1 since it was released. They are listed below in reverse chronological order. For detailed information, some of the following topics are selectable.

Z211609B - This update fixes minor spelling and other errors in the Built-in Help.

Also, the z/XDC manuals have been rebuilt and reposted to www.colesoft.com/xdc-release-z2-1.

Z211609A - This update adds commands to create and delete Proxy Tasks independently of whether or not the debugging session is running within an xxxCALLA environment.

This update also restores the default creation of Proxy Tasks by xxxCALLA.

This update implements the following:

- **SET PROXYTASKS** command and a HELP topic for it,
- **DELETE PROXYTASKS** command and a HELP topic for it,
- The ability to create/delete Formal Proxy Tasks in Foreign Address Spaces.

- A rewrite of the **FRR Proxy Tasks** section of the HELP DEBUGGING FRR topic,
- Restoration of XDCCALLA's action of creating Formal Proxy Tasks by default.
- Restoration of the **HELP DDNAMES FPTNN** topic.

Additionally, we have made substantial updates and corrections to the Built-in Help, So we have rebuilt the z/XDC manuals and reposted them to the website: **www.colesoft.com/zxdc-release-z2-1.**

I would like to thank Andre Schoeman of BMC and Charlie Pitts of Software AG for bringing this to my attention.

Z211608A - This update fixes a terminal hang that may occur when pressing the ATTN key in a native-VTAM cdf/XDC debugging session.

I would like to thank Charlie Pitts of Software AG for bringing this to my attention.

Z211606B - This update fixes an sA03 that may occur when debugging an asynchronously scheduled SRB.

I would like to thank Andre Schoeman of BMC for bringing this to my attention.

Z211606A - This update fixes an 0C4 that occurred during the startup of a debugging session under XDCCALL[A]. This failure had the following characteristics:

- The program to be debugged was flagged to run with AMODE64,
- The retry-level PSW would (incorrectly) show the execution location to be the target program's entry point address.
- z/XDC would (incorrectly) state that the retry level and error level were the same,
- The error level PSW would (correctly) show the 0C4 has having occurred on a "BSM 0,R15" instruction located in the CMDLOADR csect of XDCCALL[A].

This 0C4 would not occur for all AMODE64 programs, but it would occur when (for some unknown reason) XDCCALL[A] itself was invoked with RH15<>0.

I would like to thank Hans Schoone of IBM Netherlands for bringing this problem to our attention.

Z211605A - This fixes several issues with the rexx/XDC Interface:

- Attempts to send XDC commands to the "XDC Environment" (within the

rexx/XDC Interface) would fail with an s0C1 at xxxEFMVS+0. That has now been fixed.

However, it remains the case that no new "XDC Environment" support has been written beyond the presence of a stub that issues a DBC414E message (XDC ENVIRONMENT NOT SUPPORTED) and aborts with RC=8.

In any case, I would like to thank Ken Kornblum of BMC Software for bringing this s0C1 to our attention.

- Support for using XDC clones (Z21 for example) in the rexx/XDC Interface did not work. Instead, the Interface would ignore the xxxEFMVS module and always use the XDCEFMVS module instead. That has now been fixed.

- The sample REXX execs (in DBCOLE.XDCZ21.XDCSAMP(RXTSTccc)) did not provide any way to display which clone of the rexx/XDC Interface (load module xxxEFMVS) was in use. So I've updated the **RXTSTENV** sample exec to display that information.

- In reviewing the rexx/XDC doc, it became clear that I had been a bit sloppy with distinguishing the concept of **the REXX Environment** from z/XDC's Interface for REXX. So I have improved the doc regarding this.

- I also have improved the doc regarding z/XDC's non-support for sending z/XDC commands to the XDC Environment within rexx/XDC.

Z211603B - This fixes a bug in HOOK deletion. When the address of the HOOKCBE is specified on the **DELETE HOOKS** command, the HOOKCBE is freed but the actual HOOK is left untouched.

This bug is not expected to affect any users as specifying the address of the HOOKCBE is not typical of what the end user will use. This form of **DELETE HOOKS** was created for internal use by z/XDC.

This update also updates **DELETE HOOKS** to accept the the address of a HKODE. This was created for internal use by z/XDC

Z211603A - This update fixes a minor bug where z/XDC's internal zaps were being subjected to security checks. This has the unintended affect of causing HOOKs to failed to be created when zapping was prohibited by security to subpool 132.

I would like to thank Robert Ngan of CSC for bringing this to our attention.

Z211602C - This update fixes an issue with **LIST LSTACK lseaddress** displays. When

z/XDC is running authorized it is supposed to be possible to use **Z** shortcut commands to manually zap the modifiable area and the registers that are saved in the Linkage Stack Entry being displayed. Unfortunately, that wasn't working. This update fixes that.

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

Z211602B - This update adds logic to XDCCALL (and friends) to store the pointer of the parameter list that is generated by XDCCALL and passed to the user program in the TCBFSA field of the current TCB.

We would like to thank Mikael Nystrom of Skandinaviska Enskilda Banken for bringing this to our attention.

Z211602A - This update adds logic to prevent a zero address from being used in z/XDC's Service SVC. The use of a zero address does not impact z/XDC. The purpose of this fix is to prevent z/XDC from generating hits with any zero address detector.

We would like to thank Greg Abashian of SAS for bringing this to our attention.

Z211601A - This update fixes an error in Z21-1509H.

We would like to thank David Warner of Rocket Software for bringing this to our attention.

Z211601A - This update fixes an error in Z21-1512D.

We would like to thank Clark Hunter of Dynatrace for bringing this issue to our attention.

Z211512D - This update adds support to the xxxCALLA program to optionally debug the target program as a **jobstep** task. This is controlled by the presence or absence of a **//xxxJSTCB** allocation. for more information, see HELP XDCCALL JOBSTEPTASKS.

We would like to thank Don Ebright of Dynatrace for bringing this issue to our attention.

Z211512C - This update fixes a problem with product licensing that occurred when

the **CSVSP252ROUNDUP** setting was specified in the active **DIAGxx** member of PARMLIB. z/XDC would abort with a a **DBC975E** message.

We would like to thank Ilya Gersh of Rocket Software for bringing this problem to our attention.

Z211512B - This update fixes a problem with FRR and SRB debugging. Under certain circumstances, z/XDC may attach a proxy task to a task that contains daughter tasks that are flagged as JOBSTEP tasks. z/XDC will attach the proxy task as a non-JOBSTEP task which is prohibited by the system. This fix updated the logic to detect the JOBSTEP state of the daughter tasks and attach a proxy task to match

I would like to thank Pradeep Kohli and Mike Behne of BMC for bringing this to my attention.

Z211512A - This update fixes a problem with execution tracing. When the current machine instruction was a BXLE, BXH, JXLE, JXH, BXLEG or BXHG, whether or not a branch would occur was not always being correctly predicted for those register values that resulted in arithmetic overflows involving the sign bit.

I would like to thank Ed Boekee of the Royal Bank of Canada for bringing this to our attention.

Z211511A - This update changes the binder compatibility level of the xxxHLL and xxxCSAMP program objects from PM5 to ZOSV1R6. This corrects a failure when attempting to install and apply maintenance to z/XDC on zOS 1.6

Z211510A - This update removes the double job selection when using cs-cdf/XDC to debug programs running on another system in the sysplex.

Z211509H - This update will be appreciated by those developers who run z/XDC as an FRR to debug SRB routines and other programs that run in constrained environments.

This update removes the requirement that Proxy Tasks be pre-created. z/XDC’s FRR support will now create Proxy Tasks on an as needed basis.

In particular:

- The xxxCALL[A] program will no longer create Proxy Tasks by default.
- However, xxxCALL[A] will continue to recognize, accept and act on //xxxFPTnn dd cards if they are present.
- And z/XDC’s FRR support will continue to use xxxCALL[A] created Proxy

Tasks when present.
- However, when Proxy Tasks are not present, the FRR support will now create them as needed.

I would like to thank David Warner of Rocket Software for being the most recent of many people to make this suggestion.

Z211509G - This update fixes an 0C4 that occurs when a **MODIFY** Operator command is issued against the xxxSRVER address space.

Z211509F - This update adds System Symbol Support to server/XDC's processing of its //SYSIN file parameters. When used properly, this can significantly reduce the number of //SYSIN parameter files and VTAMLST members a multi-system Data Center would need for supporting multiple instances and clones of z/XDC across their systems.

I would like to thank Keith Blankenship of Tibco Software for making this suggestion.

Z211509E - Fixes a coding mistake that may generate ASMA044E messages during assembly when using the #XDCHOOK macro with no PFIX defined or defaulted.

Z211509D - When the #XDCHOOK macro is issued with the **RCEQUATES=YES** operand (either specified or defaulted), considerable commentary is produced describing the return and reason codes that can be generated by Hook Support Services.

However, I felt that specific information was missing regarding how the return and reason codes were presented in returned registers R15 and R0. This update adds the missing details.

Z211509C - This update corrects some blunderous changes I made in Z21-1505A to the cs-cdf/XDC section of the Factory Default //SYSIN file **(XSRVPARM)** for server/XDC.

Fortunately, most people ignored the changed XSRVPARM, but one poor guy... Please accept my apologies.

Z211509B - This update fixes a problem with the ZAP command that occurred when both of the following were true:

- z/XDC was running as an ESTAE for which **SDWALOC31=YES** was ***not*** in

effect. (This means that the high halves of the abended program's 64-bit general registers [what z/XDC refers to as the **RHn** registers] are not available to z/XDC.)

- The user wished to zap the lo-order 32 bits of a register [what z/XDC refers to as the **Rn** registers].

Attempts to ZAP an Rn register directly would fail with message DBC152E.

(The workaround was to zap the register via its **RWn** view/name. This would work so long as you made sure that the hi-order 8 digits were zero.)

We would like to thank Peter Smith of Mainframe Cloud for bringing this to our attention.

Z211509A - This update fixes a rare problem that would cause a debugging session to lose its connection to a major internal component (TFS). This would have the following consequences:

- When debugging a program running in the batch, the connection to cs-cdf/XDC would be lost, and the debugging session would terminate (allowing the current abend [s0C1 for breakpoints and #DIE traps] to percolate).

- When debugging a program running within TSO, the fullscreen display would be lost, and the debugging session would drop down to using linemode messages.

We would like to thank Mark Linker of CA Technologies for bringing this to our attention.

Z211507J - This update partially fixes an issue involving the display of disassembled code when no map has been loaded and no equates are present to guide the disassembly process.

In the absence of guidance, it is quite possible that displays produced by the FORMAT and WHERE commands (among others) will misinterpret inline data as being instructions. In fact, this happens quite frequently at the start of typical programs that have a B instruction followed by eye-catcher text. In such situations, z/XDC has no way of knowing where the inline data begins and ends and where the code resumes.

Well, this update won't get rid of the misinterpretations, but when a PSW falls within a misinterpreted machine instruction, z/XDC will now know to display the beginning of that instruction as data and to switch back to code display at the location pointed to by the PSW.

(This problem did not occur when data was being displayed, in fact, as data. In these cases, z/XDC did properly notice the PSW and resumed code formatting appropriately.)

I would like to thank Manfred Schnitzspahn of Software AG for bringing this issue to my attention.

Z211507I - This update fixes a problem that occurred during tracing when z/XDC reached an opcode that is legal but that z/XDC does not know is legal.

Previously, z/XDC was counting on the (incorrect) assumption that an 0C1 would occur. But with the opcode being legal, no 0C1 would occur, and so z/XDC would lose control of the trace.

This update makes it unlikely (but not impossible) that such loss of control would occur: Specifically, unless the unknown instruction is a branching type instruction, control of the trace will be retained.

Of course, if the unknown opcode is truly illegal, then an 0C1 does occur, just as it did previously, and that, naturally, continues to be handled properly.

I would like to thank Gary Bergman for bringing this issue to my attention.

Z211507H - This update changes the criteria upon which environment description messages (DBC830I and friends) will be displayed when z/XDC receives control from the user program.

Generally, the environment description messages are displayed during tracing only when the program's state has changed significantly as compared to when z/XDC relinquished control back to the program. This update removes AMODE from being considered as a significant state change.

I would like to thank Gary Bergman for bringing this issue to my attention.

Z211507G - This update adds partial knowledge of machine instructions created in support of the Runtime Instrumentation Facility. What is published (both on the web and in this update) are the mnemonics and the opcodes. What is not published are the operand layouts.

This update adds sufficient knowledge to z/XDC that it can recognize and disassemble RIF support opcodes, but with the operands represented only as hex strings.

The RIF opcodes are:
- X'AAx0' thru X'AAx4'
- X'EBxxxxxxxx60' thru X'EBxxxxxxxx62'

My information sources are:

- For the mnemonics: US patent #20130246746A1
- For the opcodes: The open source code for the IBM written KVM disassembler

This fix also corrects the formatting of DIAG instruction operands.

Z211507F - This update fixes an s0C4 that occurs when:

- cs-cdf/XDC is starting up.
- Parameters in the //SYSIN file indicate that Cross Sysplex Support is to be activated.
- But only one member is joined to the SYSPLEX.

When all of the above are true, message DBC200I is issued, reporting that the cs-cdf/XDC Server Subtask has failed with an s0C4.

Z211507E - This update fixes an 0C4 that occasionally occurs in XDCCALL under the following circumstances:

- A non-authorized debugging session has been started in the batch via // EXEC PGM=XDCCALL,...
- The debugging session has nearly completed,
- But at the time of completion, no user terminal is actually connected to the debugging session,
- So cs-cdf/XDC has issued its DBC640Q WTOR,
- And someone has replied **GO** to that WTOR.

Under these circumstances, sometimes an 0C4 would occur at a CLC located at or near XDCCALL+1806. This update fixes this failure.

Z211507D - This update fixes a dead trap in the **COMMENTARY** command.

Z211507C - This fixes an issue with cs-cdf/XDC when it is configured to use its Cross Sysplex Support but it is started within a single-SYSPLEX environment or under a VTAM that does not have Generic Resources enabled. Previously, cs-cdf/XDC would refuse to operate until its //SYSIN parameter file was edited to remove certain parameters. Now, cs-cdf/XDC will fall back to coming up with its Cross Sysplex Support deactivated.

I would like to thank Bill Mileski of BMC Software for bringing this to my attention. -Mike Lewis

Z211507B - This update fixes a bug that was introduced by Z21-1505F. The **Z** shortcut command was being aborted with an error message when all of the following were true:

- The display being zapped was generated by the **DISPLAY** command,
- A single, contiguous chunk of storage was to be zapped,

- However, that storage chunk spanned two display lines,
- The zap was directed into the text portions of the display.

When all these conditions were true, the ZAP command would be aborted with error message **DBC711E.**

Z211507A - This update fixes a possible security issue.

Z211506C - This update fixes a bug in Z21-1505F.

Z211506A - This update fixes a couple of bugs in the Vector Register Support implemented via the prior update:
- The mapping between floating point registers and vector registers VR0 thru VR15 was not being properly managed.
- Hilighting of changed vector registers also was not being properly managed.
- And the Built-in Help need tweaking.

Z211505F - This update implements support for **vector registers.** They can now be displayed and zapped and (when appropriate) used in Point-and-Shoot commands. For more information, see HELP * Z211505F.

There also are some minor other issues that are resolved by this update. Again, see HELP * Z211505F for more information.

Z211505E - This update implements support for a **//xxxNOALOC DD DUMMY** JCL statement. When present, z/XDC suppresses all commands that require Dynamic Allocation (SVC 99).

I would like to thank Bob Edmonds of BMC Software for making this suggestion.

Z211505D - This fixes a couple of bugs in Z21-1504E.

Z211505C - This fixes an issue with HOOK processing that would remove all transient breakpoints when the HOOK is hit.

I would like to thank Keith Tidwell of Black Knight Financial Services for bringing this to my attention.

Z211505A - The Cross Domain Facility has been renamed to **cs-cdf/XDC** to commemorate its new capability to connect to debugging sessions located on any processor connected via a Coupling Facility. The **cs-cdf** part of the new name refers to **Cross Sysplex** CDF.

Accordingly, this maintenance adds documentation for activating and using the Cross Sysplex Support, particularly at **HELP XDCSRVER CDF CROSSSYSPLEX.**

It also amends all messaging and all Built-in Help to refer to cs-cdf/XDC by its new name.

Z211504F - This fix adds support to the z/XDC Startup Panel for mixed case PARM data. A new field is added to the Panel allowing the user to control whether or not PARM field data is to be upcased. For more information, see HELP DEBUGGING ISPF PANEL.

This fix changes XDCPANEL, XDCCLIST and several of the XDCPHLPx panels (Help Panels for the Startup Panel). So you may (depending upon your installation methods) have to propagate these changes into your ISPPLIB, ISPMLIB and CLIST libraries **manually.**

Z211504E - This fixes a problem with Dynamic Hook Processing. After restoring the original code back to the hook'd location, processing would internally issue a fairly complex TRAP command to place a temporary breakpoint where the Hook had been located. Under rare circumstances, the logic would lose track of that location and, instead, try to set the trap at a random location (whatever the target code's R1 pointed to).

If the user was lucky, that attempt would immediately fail with an s0C4 abend, causing z/XDC to receive control in an environment that users would find puzzling. On the other hand, if an 0C4 did not occur, than:

- An X'00' would be stored at some random location,
- And the hooked location (having been restored) would be executed without control being transferred to z/XDC.

I would like to thank James Magill of BMC Software for bringing this to my attention.

Z211504D - This maintenance adds a warning message (DBC598W) whenever XDCCALL[A] truncates PARM field data. (Previously, XDCCALL[A] was silent about such truncations.)

Also, commentary was added to HELP XDCCALL regarding use and support of long PARM field data (via the PARMDD=ddname JCL parameter).

Z211504C - This updates XDCCALL (and XDCCALLA) to properly process the **PARMDD=ddname** parameter on the EXEC card, when specified in JCL.

PARMDD=ddname is a way to allow the passing of a PARM field that is greater than 100 bytes. Previously, XDCCALL[A] enforced a max limit of 100 bytes and truncated long PARM fields to fit. This fix relaxes that limitation and accommodates PARM fields of any length up to 32,751 bytes (9 bytes short of the PARMDD= maximum of 32760 bytes).

This fix affects XDCCALL[A] only when it is executed via JCL running in the batch. There is no similar PARMDD= like facility available for programs executed within TSO.

I would like to thank Robert Burchfield of Black Knight Financial Services for bringing this to my attention.

Z211504B - When a macro expansion is shown in a //SYSPRINT listing, often the so called "card images" that are shown are **83** characters long (not the expected 80 characters). This arises because the High Level Assembler tries to fit both a nesting level and the macro name into the card's 8 character sequence number field. This results in a 3 character overflow.

However, the type 0030 ADATA records (which are what z/XDC uses for building source image displays) truncate the images back down to 80 characters.

Well, this particularly bothered one of our customers (you know how programmers are), and he really really wanted to see the full 8 characters of his macro names.

So I discussed it with IBM's HLASM Owner, Sharuff Morsa, and as expected he had some good reasons for not wanting to change the ADATA 0030 records. But then he made a really good suggestion about how I could reconstruct the missing three characters... And that's what this fix is all about.

I would like to thank Sharuff Morsa of IBM for helping with a solution.

Z211504A - This fixes a formatting problem with the **JLU** family of instructions. Jumps in negative directions were being formatted as positive distances.

I would like to thank Mike Skopec of Rocket Software for bringing this to our attention.

Z211503E - This fixes a handshaking problem between z/XDC and LE programs. The result was that z/XDC would be invoked to handle signaling abends that should, instead, have been passed on to LE to handle as a part of its

normal processing.

I would like to thank Deborah Greer of Allen Systems Group for bringing this to our attention.

Z211503D - This fixes a problem with the parsing of an internal command.

z/XDC has an undocumented internal command named QUICK$INIT. It is used internally as part of the processing of the **//xxxQUICK DD DUMMY** keyword ddname. If the user should happen to issue this command accidentally, no harm will occur, but also there will be no apparent result.

Now of course it's not real likely that a user would issue QUICK$INIT by accident... However, it turns out that its minimum abbreviation was **Q,** and that's a much more likely command to issue by mistake. Again, there would be no harm, but the command's silence could be confusing.

Anyway, this fix invalidates all abbreviations of the QUICK$INIT command.

Z211503C - This fixes a rare problem when two XDC clones are used together in a debugging session. An abend loop occurs when an XDC clone (call it "yyy") is used to start the xxxSRVER space of another clone. When shutting down the xxxSRVER space, an sA03 abend loop occurs in the yyyCALL[A] module of the first clone.

Z211503B - GRS management products (such as CA-MIM and probably others) were converting the scope of a cs-cdf/XDC cross-sysplex ENQ from =SYSTEMS to =SYSTEM. This was interfering with cs-cdf/XDC's ability to connect to any debugging session at all. This fix adds RNL=NO to cs-cdf/XDC's ENQ to prevent its scope from being converted.

Z211503A - The **LIST XDC** command was showing a bogus maintenance level.

Z211502I - Corrects an issue where support for c/XDC interferes with FORMAT and WHERE displays of object code. The issue often causes the spurious display of message DBC310I.

We would like to thank Reza Fatemi of Tibco for bringing this to our attention.

Z211502H - Fixes a rare bug in XDCCALL: When ending a debugging session, XDCCALL would fall into a closed loop.

Z211502G - Fixes a minor display problem that would occasionally occur when debugging a background program via cs-cdf/XDC. For certain terminal display geometries, residual text would be left on the display's last row.

Z211502F - Adds disassembly support for 503 new mnemonics for 141 new SIMD support machine instructions.

Z211502E - This maintenance implements an enhancement to the Cross Domain Facility (formerly known as XDC/CDF). **cs-cdf/XDC** is no longer restricted to connecting only to running on the same processor which the user is logged on to. Now the user may connect to debugging sessions running on any processor that is connected via a Coupling Facility.

For more information, see HELP XDCSRVER CDF CROSSSYSPLEX. Also, extensive commentary can be found in the **XSRVPARM** member of the DBCOLE.XDCZ21.XDCSRVER library.

Z211502D - Fixes several potential (but not yet reported) failures in static hook support (aka #XDCHOOK support).

Z211502C - Fixes a problem when using the legacy HOOK SVC on z/OS R1.12 and older systems.

Support for z/OS R1.12 and older systems was accidentally removed from the SVC. This lead to random and unexpected results when using the HOOK SVC.

I would like to thank Reza Fatemi of Tibco and Steven Smith of Rocket Software for bringing this to our attention.

Z211502B - Fixes a problem when using the #XDCHOOK macro in AMODE24 programs. HOOK processing was using 31-bit addresses when running AMODE24. This lead to random and unexpected results.

We would like to thank Sergey Prosvirov and Sergey Igonin of Rocket Software for bringing this to our attention.

Z211501D - Adds some diagnostic code to trap a rare problem when using z/XDC in FRR

mode. A work unit that has been PAUSE'd by z/XDC processing is being RELEASE'd by something other than z/XDC. This is causing our RELEASE to fail and FRR processing to be aborted.

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

Z211501C - A prior fix (Z21-1412D) caused the XDCTFS load module to be bound without AC=1. This caused problems for those customers who are still starting the XDC Server address space with the old XDCCDF PROCs.

This fix changes the XDCMAINT maintenance file to restore the AC=1 attribute to the XDCTFS load module.

We would prefer, of course, that customers discard their old XDCCDF PROCs in favor of the current XDCSRVER proc, but we will continue to support the old XDCCDF for awhile as a courtesy.

I would like to thank Jim McPhillips of CA Technologies for bringing this issue to my attention.

Note, a sample XDCSRVER proc can be found in DBCOLE.XDCZ21.XDCSRVER(XSRVPROC). See the z/XDC Installation Guide for details.

Z211501B - Damn typos! The Hook Support rerewrite is working great for Dynamic Hooks, but was failing for Static Hooks (i.e. #XDCHOOK macros) at a dead trap: DEAD MSG: 2176 DBC527T XDC HOOK PROCESSING FAILURE - Writable Work Area not found

Legacy Hook Support (specifically, older versions of the #XDCHOOK macro) would still work. Only the new Static Hook support failed.

I would like to thank David Kreiss of BMC Software for bringing this problem to my attention.

Also, I noticed Dave Kreiss' comment about not being able to easily identify #XDCHOOK's macro version, so I've added such a message to that macro's expansion.

Z211501A - While shooting a customer issue, we ran into a problem with the **LIST SSCT** command. The command uses a sorting table that we thought was large enough, but this customer had 709 Subsystems installed, and that blew our table. This fix changes the logic to count the SSCTs before creating the sorting table.

I've also taking this opportunity to improve LIST SSCT's syntax a little. Previously, the SSCT name pattern was case sensitive. Well, that turns

out to be a bit of a pain in the neck. (P.I.T.N.?) So the patterns are now case insensitive. But if you really do have SSCT names using lowercase characters, you now can specify the name as a 'quoted string'.

We want to thank John Moore of ASG for bringing the table overflow problem to our attention.

Z211412D - A customer pointed out, the other day, that z/XDC did not recognize the **MSDR** instruction. Well, it turns out that there were around a dozen or so published instructions that were overlooked. This fixes corrects that oversight.

We would like to thank Vit Gottwald of CA Technologies for bringing this issue to our attention.

Z211412C - Fixes several bugs and design problems with the the **HOOK** and **HDEFERRED** commands that our customers have been reporting.

Also, a number of additional related and unrelated small changes are made by this fix. For more information, see **HELP * Z211412C.**

We would like to thank Ken Scott of Trident Services for being the first to bring these issues to our attention.

Z211412B - Addresses a rather annoying problem for people still using 80-character wide display terminals. The Factory Default Profile for 80-column displays (named **A80** in this release) was defaulting to displaying the 64-bit wide registers in the Watch Window. The problem is that displaying wide registers on narrow displays burns **eight** rows of precious display real estate! For more information, see see **HELP * Z211412B.**

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

Z211412A - Fixes a bug: Recent changes to MAP command processing included a feature called **AUTOMAP,** which was overzealously attempting to MAP unmapped CSECTs.

We would like to thank Jim Martin of Fundi Software for being the first to bring this issue to our attention.

Z211411C - Fixes #DEAD trap 8728 when attempting to map a GLOBAL LOAD'd module while using FASM. For more information, see see **HELP * Z211411C.**

We would like to thank Chris Parket of Imperva for bringing this issue to our attention.

Z211411B - Fixes a bug: Recent changes to MAP command processing caused a situation where message DBC119E was suppressed inappropriately.

Z211411A - Fixes a bug: Recent changes to MAP command processing caused a situation where message DBC986S was displayed inappropriately.

We would like to thank Greg Grounds of Imperva for bringing this issue to our attention.

Z211410E - Fixes a bug: Under uncommon circumstances, z/XDC would delete its knowledge of a breakpoint but not clear the breakpoint itself, thus converting the breakpoint into a spurious s0C1 abend. For more information, see see **HELP * Z211410E.**

We would like to thank Greg Grounds of Imperva for bringing this issue to our attention.

Z211410D - Makes a minor correction to HELP SUPPORT CONTACTUS: Removes a FAX# we no longer have.

Also moves this MAINTENANCE topic into a subtopic of THINGSFIXED.

Z211410C - Updates the VTAM login panel for cs-cdf/XDC to accept pass phrases up to 100 characters.

Phrases that are less or equal to 8 characters will be treated as passwords.

Z211410B - Updates the XDCADATA Filtering Program to retain Job Identification Records (Type X’0000’) instead of discarding them. (z/XDC does not make use of Job ID records, but one of our customers wanted them preserved for their own reasons.)

We would like to thank Fred Hoschett of Phoenix Software International for making this suggestion.

Z211410A - Z1D1110A - Unlock z/XDC to permit it to execute.

Help Whatsnew Z21 THingsfixed Maintenance Z211505f

The Z21-1505F update's primary purpose is to implement **vector register** support in z/XDC. But it makes some other "housekeeping" changes as well. Those are described further below.

Vector Register Support

IBM's current incarnation of vector registers was first implemented on its **Z13** Processor. They are part of the new **SIMD** feature which also includes several hundred new machine instructions for manipulating the vector registers. (We've already published SIMD support with the **Z21-1502F** update.)

Vector registers are 16 bytes wide, and there are 32 of them. They can be...

- Displayed collectively by the **LIST VREGS** command
- Displayed individually by the **LIST VRn** command
- Zapped by the **ZAP** command
- Also zapped by the **Z** Shortcut Command

For more information, see HELP COMMANDS LIST VECTORREGISTERS.

Other Things Changed or Fixed

Previously, z/XDC would always save and restore all sixteen floating point registers. Unfortunately, that had the undesirable side effect of notifying z/OS to activate its own logic to define, save and restore the APF floating point registers for the current execution thread (task or SRB). (AFP means Advanced Floating Point and it includes support for Binary Floating Point arithmetic and, sometimes, Decimal Floating Point.)

Well, this has been fixed. If APF support has not yet been activated in the current thread, then:

- z/XDC will not unintentionally cause it to be activated.
- In a LIST FREGS display, the AFP registers will be shown dashed out.
- However, if you use a zap to set values in a dashed out AFP register, then z/XDC will intentionally activate them.

During the development of this code, one of the things that became clear is that I was not properly saving and restoring Floating Point Registers in the situations where z/XDC was being used as an FRR routine (for example, when debugging SRB routines). This update fixes that.

Another is that the distinction between error level and retry level Floating Point

and Control Registers is simply nonsensical. Accordingly, all commands and built-in equates in support of "error level" Control/Floating Point Registers have either been removed or deprecated.

- The following built-in and automatic equates have been removed:
  - @ECHREGS
  - @ECREGS
  - @EFREGS

- The following commands have been deprecated: (They will continue to exist, but they no longer will be documented.)
  - LIST ECREGS
  - LIST ECRHREGS
  - LIST ECRWREGS
  - LIST ECRn
  - LIST ECRHn
  - LIST ECRWn

Help Whatsnew Z21 THingsfixed Maintenance Z211412C

Z21-1412C fixes a number of problems that have been occurring with the **HOOK** and **HDEFERRED** commands. Basically, this fix is a rerewrite of hook support. Our testing shows that there remains problems when trying to hook disabled code, but otherwise, everything now seems to be working as it should.

Things fixed include the following:

- A sB78-5C abend that occurred when a hook located in key-0 code located in common storage was executed by a program running in problem state.

- When a Deferred Hook was defined while z/XDC was running authorized, but the program into which the hook was eventually set was running in problem state, but the hook’d module was loaded into key 0 storage, the removal of the hook would fail with an "unable to remove" message.

- Hook Processing sometimes would fail at a dead trap that contained the message **DBC548T [...] xxx/Server needs to be started,** but the server was already running.

- Under some conditions, deferred hook definitions would be ignored. They would not set hooks.

- Etc.

All these issues (and others) have finally been fixed.

We would like to thank Ken Scott of Trident Services, Mark Ruhe of BMC Software and Mike Puiu of MVS Solutions for bringing these problems to our attention.

IMPORTANT!

Hook Processing now **unconditionally REQUIRES** the xxxSRVER address space be up and running. (Previously, only certain hooks required the server space.) If the HOOK and HDEFERRED commands detect that server space is not up, they will now always abort.

Also, problems remain for setting hooks into disabled code. Those will be addressed via future maintenance.

Additional Small Changes

The following additional small changes were also made via this fix:

- The **LIST XDC** command now shows two additional items of information:
  - It shows whether or not z/XDC currently is in supervisor state. (So this command is now an alternative to **LIST MSGS** for displaying that information.)
  - It shows that our LinkedIn forum can now be reached at **www.xdc.com.**
- Operand consistency checking has been enhanced for the **GETMAIN** command. For example, when obtaining storage from a 31-bit-only subpool, the **RMODE=24** operand is now disallowed.
- Doc for the **HDEFERRED** command previously was folded into the ADEFERRED/TDEFERRED doc. It has now been extracted from there and moved into its own topic: HELP COMMANDS HDEFERRED. This is because the HDEFERRED command has operands **(AMODE=** and **STATE=)** that the other commands do not.
- The **HOOK** command will now disallow specifying RMODE=24 on hooks being set into 31-bit code.
- When **STATE=LOCKED** is specified on the HOOK command, z/XDC now vchecks that the XDC load module is located in page fixed storage. If it is not, then the command is disallowed.
- When a breakpoint was set on top of a Hook's **JLU,** and that location was then displayed via a **FORMAT** command, z/XDC's knowledge of the hook was lost. It would no longer be displayable via a **LIST HOOKS** command.

Help Whatsnew Z21 THingsfixed Maintenance Z211412B

Z21-1412B addresses a rather annoying problem for people still using 80-character wide display terminals. The Factory Default Profile for 80-column displays (named **A80** in this release) was defaulting to displaying the 64-bit wide registers in the Watch Window. The problem is that displaying wide registers on narrow displays burns **eight** rows of precious display real estate!

This fix reverts the A80 profile to displaying only the old 32-bit registers, thus

recovering three (yes, only three) rows. (A trailing blank row is retained for displaying a warning a message when the high halves of the registers are not all zero'd.)

This fix also corrects some misinformation in the HELP PROFILES FACTORYDEFAULTS A80 and HELP PROFILES FACTORYDEFAULTS AWIDE topics.

We want to thank **Dante Ferman** of Innovation Data Processing for bringing this problem to our attention.

Help Whatsnew Z21 THingsfixed Maintenance Z211411c

Z21-1411C fixes #DEAD trap 8728 when attempting to map a GLOBAL LOAD'd module while using FASM.

The #DEAD trap will be hit under the following circumstance:

- z/XDC has been set (via a **SET ASID** command) to Foreign Address Space Mode (FASM).

- A **MAP** command has been issued to map a load module that is owned by that address space but is located in CSA.
- The operand of the MAP command is an address expression (not a pure module name). Examples: MAP +0, MAP 00129867 or MAP MODULEA+100.
- The programs running in the target address space had loaded the module into CSA via the LOAD macro with **GLOBAL=YES** specified.

When the module is loaded into CSA, the System builds a CDE and places it on the target address space's Job Pack Queue **(not** on the System's Link Pack Queue). This effectively hides the module from other address spaces unless it's address is explicitly known by some other mechanism.

Under these circumstances, z/XDC was unable to match the module to an internal control block, leading to a logic error.

We want to thank **Chris Parker** of Imperva for bringing this problem to our attention.

Help Whatsnew Z21 THingsfixed Maintenance Z211410e

Z21-1410E fixes a bug - Under uncommon circumstances, z/XDC would delete its knowledge of a breakpoint but not clear the breakpoint itself, thus converting the breakpoint into a spurious s0C1 abend.

This would occur under the following circumstances:

- Routine A would call (BALR, whatever) routine B.
- The called routine (B) was located in a load module that was:

  - Different from the load module that contained the calling routine (A).
  - Located at a higher storage address than the calling routine.
- A permanent (AT type) breakpoint was contained in the calling routine such that it would be executed at some point following B's return to A.
- Another breakpoint (permanent or transient) was located in the called routine such that it would be executed prior to the breakpoint in the calling routine.

Under the above circumstances, the following would happen.

- Execution would flow into the called routine, where its breakpoint would be executed, thus passing control to z/XDC which would then display the program code via a **WHERE** command.
- Eventually, the user would use a **GO** command to allow execution to resume and eventually return to the calling routine where the permanent breakpoint would be executed thus causing another **WHERE** command to be executed.

There was a bug in the **WHERE** command's logic that would cause z/XDC to purge its knowledge of the permanent breakpoint but would allow the breakpoint itself (a X'00' "opcode") to remain.

We want to thank **Greg Grounds** of Imperva for bringing this problem to our attention.

Help Whatsnew Z21 Incompatibilities

Some changes have been made that are incompatible with prior releases of z/XDC. Brief descriptions are given here.

Built-in Help Changes

Throughout all of Built-in Help, all references to "Online Help" have been changed to **Built-in Help.**

The following structural changes have been made to the Built-in Help:

- The topic named HELP ADDRESSING ASM has been moved TO **HELP ADDRESSING PARSERS ASM.**
- The topic named HELP CDF has been moved TO **HELP XDCSRVER CDF.**
- The topic named HELP DEBUGGING HOOKS has been moved TO **HELP HOOKS.**
- The topic named HELP LINECMDS has been renamed to to **HELP SHORTCUTCOMMANDS.**
- All of the subtopics of HELP SCRIPTS have been moved to be subtopics of **HELP SCRIPTS OURSCRIPTS.**

These changes affect, of course, all references to the included subtopics.

Counting Conditions for Breakpoints

When a single breakpointing command (AT ATX TRAP ADEFERRED and TDEFERRED)...
- Set multiple breakpoints
- And had a counting condition operand,

Then previously the given count value was handled as a collective limit: When any of the breakpoints were reached by execution, a single hit counter would be incremented and tested against the count limit.

Now, however, a separate counter is maintained for each of the breakpoints individually.

So now, a breakpoint is not accepted until it itself is reached by execution the count number of times.

DELETE MAPLIBS: Support Deprecated for [NO]SHOW and QUIET Operands

The DELETE command's new **SILENT=YES** operand is functionally redundant with the DELETE MAPLIBS' previously existing **[NO]SHOW** and **QUIET** operands. Accordingly, support for these older operands has been deprecated. They will continue to be supported, but their documentation is being removed.

Display Colors Defaults

The factory default display color settings have changed. Previously, they were RWCY. Now, they are **RWMY** (Red White Magenta Yellow).

With the older settings, the contrast between Cyan and White was pretty low. I believe these new settings improve the contrasts between the various field types.

For more information, see HELP COMMANDS SET COLORS.

HOOK [E]SPIE Management is Changed

Hook Processing's handling of [E]SPIEs has changed. Previously, at the time a Hook is reached by execution, if:
- The program being hooked is running:
  - In problem state
  - And with HASN=PASN=SASN
- And if an [E]SPIE exists
- And if said [E]SPIE intercepts any Program Interrupt Codes (PICs) 0001 thru 0007

Then the [E]SPIE would be changed to exclude those particular codes from being intercepted.

Now, however, when Hook Processing detects such an [E]SPIE, it cancels the [E]SPIE entirely.

Note, it remains the case that if any of the above conditions are not met, then the

[E]SPIE is left alone. This is because when any of the above conditions are not met, the System ignores the [E]SPIE, and so interference with z/XDC does not occur.

HOOK SVC Support is Deprecated

As noted elsewhere, z/XDC's Hook Support has been entirely rewritten to remove the need for a hook SVC instruction. But also as noted, the hook SVC continues to be provided so that programs assembled with prior versions of the #XDCHOOK macro do not need to be reassembled.

However, it needs to be noted that support for the hook SVC has been deprecated. In the event that problems are discovered with the SVC, we might choose not to fix them. Instead, we might simply recommend you reassemble code containing #XDCHOOK macros so that newer (and supported) logic is generated.

JES2 Update no Longer Provided

For a very long time now the z/XDC Installation Package has included a JES2 update that created a debugging interface within JES2 (for those who are developing and maintaining JES2 mods).

The update has not been changed or even tested in decades! At this point it is highly doubtful that it even works any more. So I've decided to remove it from the product.

If this creates a problem for you, please let us know.

MAPLIBS: Default MAPLIBS List no Longer Provided

z/XDC will no longer provide an initial default MAPLIBS list. It will continue to support default lists that you might create, but it will no longer provide one of its own. For more information, see HELP WHATSNEW Z21 MAPLIBSLIST.

Online Help Changes

Throughout all of Built-in Help, all references to "Online Help" have been changed to **Built-in Help.** So for information about Built-in Help incompatibilities, you're looking in the wrong place. You need to scroll back up to the **Built-in Help Changes** topic.

Profile Menuing System

Certain PF keys (PF3, PF4 and PF5) have specific, hard-coded functions within the

Profile Menuing System **regardless** of how they are defined by the **KEYS** command outside of the Profile Menuing System. Those meanings are documented in HELP PROFILE MENU.

Previously, for some of the Profile Menuing System panels, some of those PF keys were behaving differently from their defined functions. That has now been cleaned up.

The behavior of PF4 (CANCEL) has been changed. Previously, it would cancel all changes but remain within the panel. Now however, it still cancels all changes, but it also closes the panel and pops out to the next higher level. This brings the behavior of PF4 into conformity with its behavior in all other specialty panels.

For more information, see:

- **HELP PROFILES MENU** (the root of all information about the Profile Menuing System)

PROFILE RESET Command Operand Changes

The PROFILE RESET command used to accept up to two optional operands that allowed you to change the name of the profile upon being loaded. (Absent those operands, the profile would be named **XDC** when loaded.) Those operands are no longer supported.

Now, the PROFILE RESET command accepts only one optional operand whose purpose is to select from among several Factory Default profiles. For more information, see HELP COMMANDS PROFILE RESET.

Default Profiles are no Longer Being Distributed with the Product

Because of improvements in the PROFILE RESET command, we feel it is no longer helpful to also distribute **precreated default profiles** with the product. Instead, the **z/XDC Installation Guide** explains how to create local default profiles when your local requirements differ from what is created by the PROFILE RESET command.

Default Profile Name Format has Changed

Previously, the format for true names for the **XDC** default profile was xxxXDCrr. Now it is **xxxDPrrr.** ("DP" stands for "Default Profile".)

If the old format were still being followed, the default profile name for this release (z2.1) would have been xxxXDC21. But instead, following the new format, the default profile name is **xxxDPZ21.**

This format change was made to avoid a future conflict with the default profile name for a very ancient release (XDC x2.2).

Syntax Changes in the Breakpointing Commands

Several syntax and processing changes have been made to the several breakpointing commands. Some changes are compatible, some are not. See HELP WHATSNEW Z21 SYNTAXCHANGES. for more information.

#XDCHOOK Redesign

As noted elsewhere, z/XDC's Hook Processing has been entirely redesigned in this release. This redesign includes substantial changes to the #XDCHOOK macro. In particular, the runtime logic generated by the macro is now significantly different. So if your code has dependencies upon the internals of that logic. you will have to review your code.

However, the redesigned #XDCHOOK retains its same functionality: Starting a debugging session without disturbing the execution environment.

Further, #XDCHOOK is been written with complete legacy support in mind...
- Code assembled with an older #XDCHOOK will still work within the newer z/XDC.
- And vice-versa: Code assembled with the newer #XDCHOOK will still work within older z/XDC's.

For detailed information about this, see HELP HOOKS STATIC #XDCHOOK.

#XDCHOOK Incompatibilities

Previously, #XDCHOOK's logic may have preserved registers RW14-RW1 and RH14-RH1. Now, however, it does not. Some of these registers are now used for returned data, while the contents of the rest are not intentionally preserved.

(If you want 100% register preservation, use Dynamic Hooks.)