Release Notice
CONVEX CXdb V1.1

Document No. 710-014830-004

November 1991

CONVEX Computer Corporation
Richardson, Texas

Release Notice, CONVEX CXdb V1.1

Printed in the United States of America

CONVEX Computer Corporation

1

Release Notice

Introduction

This document describes the V1.1 production release of the CONVEX Visual Debugger, *CXdb*. It supplements the current documentation with information and features that were developed too late to be included in the CXdb documentation. Always refer to this release notice before reporting questions or problems with CXdb; your questions may be answered here. Fixes and workarounds are listed here that may keep you from rediscovering known problems.

The remaining sections in this chapter describe the contents of this release.

- Contents of this distribution.
- Notes and warnings about the product.
- Enhancements.
- Known software problems.
- Known documentation problems.

For instructions on installing CXdb, see *Installation Procedures, CONVEX CXdb V1.1.*

Contents of This Distribution

The distribution package for this release of CONVEX CXdb consists of

- This Release Notice
- CXdb Installation Procedures
- CXdb Reference Manual
- CXdb Quick Reference
- CXdb User's Guide
- Distribution media for the software

The specific contents of the software distribution is described in the following tables:

Table 1-1: Software for USA and International Distribution

Qty	Type	Part Number	Description
1	Mag Tape	710-008215-003	CONVEX CXdb, V1.1

Table 1-2: Documentation Release Package

Qty	Type	Part Number	Description
1	Document	710-014930-004	Installation Procedures, CONVEX CXdb V1.1
1	Document	710-014830-004	Release Notice, CONVEX CXdb V1.1
1	Document	710-015330-001	CONVEX CXdb Concepts, First Edition
1	Document	710-015430-001	CONVEX CXdb Reference, First Edition
1	Document	710-015530-001	CONVEX CXdb User's Guide, First Edition
1	Document	710-015630-000	CONVEX CXdb Quick Reference, First Edition

Notes and Warnings

This subsection contains general useful information and words of caution about the product.

- The debugger is stamped with a unique serial number. At runtime, the debugger checks the serial number of the machine on which it is running. If the serial number does not match the expected one, a user error message is given and execution is aborted.

- This release of CXdb comes with an on-line tutorial. This tutorial only runs under the X Window System. To execute the tutorial, perform the following commands:

  % **cd /usr/lib/cxdb/tutorial**
  % **setenv DISPLAY <*your_display*>:0**
  % **./cxdb_guide**

  Once *cxdb_guide* is running follow the instructions it provides.

  The tutorial can also be accessed from the pull-down menus within the help window when running under the X window system.

- For the Domestic distribution, this release requires the installation of entire contents of the following:

  — CONVEX FORTRAN V7.0 and/or CONVEX C V4.1 or CONVEX C V4.3.0.1. These compilers generate instrumented code for use with CXdb.
  — To work with CONVEX FORTRAN V7.0 you will need ConvexOS V9.1.
  — To work with CONVEX C V4.1 you will need ConvexOS V9.1.
  — To work with CONVEX C V4.3.0.1 you will need ConvexOS V10.0.
  — To use the X window system features of CXdb you will need CONVEX CXwindows V2.1.

Enhancements

This release significantly extends CXdb V1.1's capability to debug optimized code. This includes:

- Source code to object code correlation at the -O2 and -O3 optimization levels enabling full source level debugging of code compiled at any optimization level.
- Synthesized variable tracking which correlates compiler generated variables to user define variables. This allows the values of user defined induction variables to be printed while debugging.
- Support for debugging dynamically loaded object modules. This allows debugging of object modules which are dynamically loaded into an application. See the "load object" reference page for further details.
- The on-line reference pages have been updated to reflect current CXdb functionality. If an inconsistency between the hard copy version of the Reference Manual and on-line version of the reference pages is found, use the on-line version.

For more information on debugging optmized code consult the CXdb User's Guide. Note that this new functionality requires C V4.3.0.1 or later or FORTRAN V7.0 or later, C V4.1 does not support the above functionality.

In addition to these capabilities, CXdb V1.1 includes a significant number of bug fixes and performance enhancements making CXdb a more robust and effective tool.

Known Software Problems

This section contains the known problems with CONVEX CXdb software and compiler instrumentation as of Aug 27, 1991. Any problems reported after this date may not be reflected in this document. Please refer to this list before reporting a problem to ensure that it has not been reported previously. Serious problems include work-arounds if they are known.

- X-21589

  Array slices should work for pointer types within C. For example:

```
(CXdb) p __1currentBuffer ->charPtr
(int*) 0x80d7c600
(CXdb) p __1currentBuffer ->charPtr[0]
(int) 0
(CXdb) p __1currentBuffer ->charPtr[0..11]
ERROR: 334
Invalid Slice Expression, line: 1 col: 20  Left operand of '[]' is not an array type
```

- X-21559

  Find window backward and find window forward are defined to have a default parameter for the window to be searched. The default is said to be the current source window. Unfortunately, nothing happens when a window parameter is not provided.

- X-21463

  The default format for floating point numbers when being 'print'ed needs to be beefed up. For numbers less than 10.0**-4, the default causes the value to be printed to be zero.

- X-21295

  Info break should not provide a message that there are no breakpoints when there are none. Right now, it just contains a blank line...

(CXdb) i bre

(CXdb) i event

No events currently defined

(CXdb)

- X-21291

Dynamic object modules referenced during debug session of SAS programs are not unloaded when CXdb rerun command is used to restart the program.

As execution proceeds dynamic objects previously mapped are mapped a second time (appear twice in info dynamicobject summation).

Any breakpoint events previously defined in dynamic objects remain defined, but are not honored in new invocation.

CXdb often terminates in this senerio.

- X-21240

There is an inconsistency between process/process defaults and eventpoints/eventpoint defaults and what happens when you modify the defaults

- X-21193

Command completion appears to be broken in the following case:

```
i sym isEntryEqua
```

is converted to...

```
i  sym isbols EntryEqua
```

- X-20739

The RELATED COMMANDS section of the on-line help for 'info line' has duplicate entries for 'info sourceunit'.

- X-20736

In batch mode, the output of a process is sometimes echoed to standard out, and sometimes not.

- X-20710

Incorrect information regarding the arguments to ENTRY points is given by the 'info args' and 'print' statements.

```
      subroutine sub1 (a, i, b, j)

10      format (a, i1, a, i1)
20      format (a, f5.2, a, f5.2)

      write (*, 10) 'i is ', i, ' j is ', j
      write (*, 20) 'a is ', a, ' b is ', b
      return
```

```
      entry ints (i, j)
      write (*, 10) 'i is ', i, ' j is ', j
      return

      entry reals (a, b)
      write (*, 20) 'a is ', a, ' b is ', b
      return
      end
```

Alternate entry agruments are not properly evaluated in the alternate entry prolog or if there is a return at the end of the preceding entry.

- X-20404

  Single stepping by statement produces radically different than when single stepping by expression when steping over calls to intrinsics that are realized with callq's.

- X-20443

  Passing a binary file to CXdb for with the -f option causes CXdb to coredump.

- X-20966

  Thread creation within cxdb still isn't consistent. With fixed scheduling on, sometimes a process will spawn threads, sometimes not. An "event spawn" will sometimes catch the event, and sometimes the process will be stopped by an exiting thread and never get caught by the event spawn. This seems to be true whether or not the threads are compiler-created.

- X-20174

  The quick reference guide for CXdb contains an error on the description of the cxdb invocation command line flags. It indicates that the -D flag accepts a colon (:) seperated list of paths. It actually accepts a comma (,) seperated list of paths.

- X-20030

  Thread creation within CXdb isn't consistent. An executable can be run a hundred times under "mpa -f" and give multiple threads every time. Running with fixed scheduling under CXdb, sometimes threads are created, sometimes not.

- X-19921

  When a program segmentation faults in a library routine, the source window gets confused. Even though it knows where the source for the library routine is, it stays in the file that main() is in.

- X-19825

  CXdb does not properly highlight block source units. When in the body of a loop the loop source unit, not the blocks source unit, will get highlighted.

- X-19791

  The "**step over**" command doesn't work properly on nested loops when the stepping granularity is set to "**loop**" and the starting point of execution is not the start of the inner loop.

  discussion:

The "**step over**" command is incorrectly stopping at the same statement of the inner loop on the next trip of the outer loop.

work-around: None.

- X-19660

With a stack window open and an arguments/locals pop-up displayed, printing any expression (via the "**print**" or "**eval**" commands) causes the popup window to close.

work-around: None. Clicking on the appropriate stack window line will bring the pop-up window back.

- X-19528

The eventpoint description pop-ups do not update when an eventpoint handler is added or modified on the eventpoint being displayed.

work-around: None. However, closing the pop-up and redisplaying it will incorporate the correct data.

- X-19055

Under Maryland Windows (CRT mode): The M-z command uses reverse video to highlight the box being resized. When the command is issued, the title bar is also reversed (from an already reversed state) so it appears the box's top dimension is the top of the box above it. Similar for M-m but the reversed title bar gets left there.

Example:
1. Create an Xterm 80x52
2. cxdb -nw
3. help help <position window so geometry is 79x23+0+29>
4. M-z <title bar is no longer highlighted. Looks like top of help window is top of command window.>

work-around: None. This is a problem with Maryland windows.

- X-19047

Under Maryland windows (CRT mode) running in an xterm window, resizing the xterm window smaller than the command window produces incorrect screen updates. This is a bug in Maryland windows.

work-around: None. Resizing the xterm window larger than the command window will restore correct behavior.

- X-19046

Under Maryland Windows (CRT mode): When the xterm has more than 60 rows, Marylnad Windows does not use the rows after the 60th. If you reverse scroll (bound to the keystroke Meta-V) when there are more than 60 lines in the command window, random characters are left in the bottom few rows of the xterm window.

Example:
1. Create an Xterm with 65 lines
2. run cxdb -nw
3. in command window say "info bind"
4. Now scroll backward with M-V
5 info bind output goes to bottom of Xterm instead of stopping at end of

command window.

work-around: None. This is a bug in Maryland Windows.

- X-18887

  When a window with a menu bar is resized to be so short that the menu bar doesn't have very much height, then resized to be large enough, the menu bar still doesn't get drawn any taller.

  work-around: None. This appears to be a bug in Motif.

- X-18886

  When error messages are displayed in the command window due to errors detected in pop-up interaction, the command window prompt is not redrawn.

  work-around: None. However, pressing the return key will produce a new prompt.

- X-18883

  The command window doesn't scroll after resizing to keep the prompt in view.

  work-around: None. However, pressing any key will cause the command window to reposition to the bottom (which will bring the prompt into view). This is a bug in Motif.

- X-17944

  When an exec eventpoint is set with an eventpoint handler, the process receives a segmentation fault. The cxdb prompt then disappears and CXdb emits an error message saying that there is an invalid command in the eventpoint handler (even though the handler is valid). Even though there is no longer a prompt, commands can still be entered.

  work-around: None.

Known Documentation Problems

The following reference pages are not included in the reference guide or the on-line help database:

2

Compiling for CXdb

CXdb V1.1 can be used to symbolically debug FORTRAN source code compiled with CONVEX FORTRAN V7.0 or later. CXdb V1.1 can be used to symbolically debug C source source code compiled with CONVEX C V4.1, V4.3.0.1, or later. CXdb V1.1 can readily be used to debug applications composed of both FORTRAN and C source code. CXdb V1.1 can't be used to symbolically debug Ada source code.

To compile either FORTRAN or C source code for debugging, simply specify the compiler flag, **-cxdb**, in the compilation command. Debugging is supported up to the **-O1** optimization level. Debugging may be attempted at higher optimization levels, but the source code to object code mapping will not be accurate.

CXdb does not support debugging of inlined code in FORTRAN. The **-cxdb** flag cannot be used in conjuction with **-il** or **-is** flag. CXdb requires object modules to be generated directly by the compiler. Objects that are produced via assembly files will not have complete CXdb debugging information. This includes files compiled with **-S** or **-pb** flags or C files that have **asm** statements in them.

FORTRAN Compiler

By default the CONVEX FORTRAN V7.0 compiler resides in */usr/convex/newfc/fc*. Optionally, it may also be available in */usr/convex/fc*. CONVEX FORTRAN V7.0 is an optional product.

C Compiler

By default the CONVEX C V4.1 compiler resides in */bin/cc*. By default the CONVEX C V4.3.0.1 compiler resides in */usr/lib/newcc/cc*. CONVEX C V4.1 is a standard product. CONVEX C V4.3.0.1 is a field test product and is not generally available.

Known C Compiler Problems

There are a number of known problems in the C compiler that will affect debugging. These problems manifest themselves as either incorrect highlighting of source code or as incorrect source code to object code mappings at the expression level. These incorrect source-to-object mappings will affect stepping at the expression level or at optimization levels above **-no**.

- Include files which contain executable source code (not including macro definitions) create source units with incorrect source positions. Such source units either highlight incorrectly (when active) or do not highlight at all.

- Scope blocks for external routines are not globally visible. Ideally, you should be able to specify variables in external routines by qualifying the variable with the routine name of the variable. For example, *routine'var*, would reference variable *var* in routine *routine*. Instead, you must qualify the variable with the scope block of the file, as in

file'routine'var.

- In some situations scope block numbers are not correctly assigned to nested scope blocks.
- Structure definitions in function prototypes will produce incorrect scope block numbering in the code following the prototype.