CONVEX C V5.0
Release Notice

Document No. 720-002030-017

January 1993

CONVEX Press
Richardson, Texas
United States of America

CONVEX C V5.0
Release Notice

Although the material contained herein has been carefully reviewed, CONVEX does not warrant it to be free of errors or omissions. CONVEX reserves the right to make corrections, updates, revisions, or changes to the information contained herein. CONVEX does not warrant the material described herein to be free of patent infringement.

UNLESS PROVIDED OTHERWISE IN WRITING WITH CONVEX COMPUTER CORPORATION (CONVEX), THE PROGRAM DESCRIBED HEREIN IS PROVIDED AS IS WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. SOME STATES DO NOT ALLOW THE EXCLUSION OF IMPLIED WARRANTIES. THE ABOVE EXCLUSION MAY NOT BE APPLICABLE TO ALL PURCHASERS BECAUSE WARRANTY RIGHTS CAN VARY FROM STATE TO STATE. IN NO EVENT WILL CONVEX BE LIABLE TO ANYONE FOR SPECIAL, COLLATERAL, INCIDENTAL, OR CONSEQUENTIAL DAMAGES, INCLUDING ANY LOST PROFITS OR LOST SAVINGS, ARISING OUT OF THE USE OR INABILITY TO USE THIS PROGRAM. CONVEX WILL NOT BE LIABLE EVEN IF IT HAS BEEN NOTIFIED OF THE POSSIBILITY OF SUCH DAMAGE BY THE PURCHASER OR ANY THIRD PARTY.

Printed in the United States of America

1. Introduction

This document is intended to provide users of the CONVEX C Version 5.0 compiler with notice of improvements, new features, and other pertinent changes to the compiler.

The remaining sections in this document describe the following aspects of the CONVEX C Version 5.0 compiler:

- Section 2 describes the software and hardware necessary to run this software.
- Section 3 describes the new features provided in this release.
- Section 4 describes the documentation provided with this release
- Appendix A lists known defects that may produce incorrect results
- Appendix B lists reported defects and enhancement requests that have been resolved in this release
- Appendix C lists remaining open bugs. Open wishes are not listed.

The CONVEX C compiler is both a scalar optimizing compiler (automatically performing local and global scalar optimizations), a vectorizing compiler (automatically performing vectorization of loops), and a parallelizing compiler (automatically performing parallelization of loops). *The capability to perform automatic vectorization and parallelization of code is an optional feature*; ask your CONVEX sales representative for details. The part number for the automatically vectorizing compiler C compiler is 720-000615-222. The part number for the scalar optimizing compiler C compiler is 720-002715-013.

CONVEX C Version 5.0 will consist of the compiler driver `(cc)`, the compiler `(cocc)`, the error message file `(errmsg.cc)`, the runtime system libraries, the man pages, other C-specific utilities, and documentation.

2. Serial Numbers and Prerequisites

This section discusses general information and prerequisite software and hardware for CONVEX C Version 5.0.

SERIAL NUMBERS

The fully optimizing compiler checks the serial number of the machine on which it is running. If the serial number does not match the expected one, a message is printed and execution is aborted.

The scalar optimizing compiler does not perform serial number verification.

PREREQUISITES

CONVEX C Version 5.0 requires the installation of the following software:

- ConvexOS Version 9.1 or later. is acceptable.
- CONVEX Assembler, Loader and Libraries (ALL) Version 2.0.

CONVEX C Version 5.0 is compatible with the following optional program development tools:

- CONVEX CXpa Version 1.3 or higher
- CONVEX CXdb Debugger Version 2.0 or higher
- CONVEX CXmetrics Version 1.0 or higher

3. New Features

This section lists new features included in CONVEX C Version 5.0.

PROCEDURAL POINTER TRACKING

Version 5.0 of the CONVEX C Compiler introduces pointer tracking to the procedure compiler. Pointer tracking reduces aliasing on pointers by keeping track of the possible objects that each pointer could reference.

Pointer tracking is performed at optimization levels -O1 and above.

Refer to the *CONVEX C Optimization Guide* for details.

RESTRICT QUALIFIER

Another mechanism for reducing aliases has been added to this release. The *restrict* qualifier offers the programmer an opportunity to assist the compiler in the reduction of potential aliases. The *restrict* qualifier is used to identify pointers that have exclusive access to objects. Using this information, the compiler is able to perform more optimization.

Refer to the *CONVEX C Optimization Guide* for details.

OPTIMIZATION REPORT ENHANCEMENTS

The optimization report has been enhanced to provide more complete information in a more logical fashion for CONVEX C Version 5.0. The following enhancements have been made:

- Transformation optimizations (loop distribution, loop peeling, loop promotion, and dynamic selection) are now marked with an asterisk; (*).
- Dynamic selection is now clearly labeled as such and allocated a separate line with the other transformation optimizations.
- Loops are now assigned ID numbers, which are indicated in a column of the optimization report and can be referenced in other parts of the report.
- ID numbers of loops created by the compiler are reported in a separate column on the same line as the loops from which they were created.

- The original ordering for loop nests is maintained in the report, with compiler generated loops immediately following the source loops from which they were generated. The only exception to this ordering occurs in interchanged loops, in which case the loops to be interchanged are kept on consecutive lines.

- Long variable names are truncated, footnoted and expanded in a separate table instead of simply truncated.

Optimizer Changes

The compiler will be more selective when vectorizing loops with a compile time determinable trip count. Depending on the complexity of the loop, some loops will no longer vectorize when the compiler determines is is not profitable to do so.

`-except` options

The `-except` options concern the treatment of arithmetic exceptions generated within functions. The command line options are `-except precise` and `-except default`.

The `-except precise` option generates code that ensures that any arithmetic exceptions generated within functions before the return will be received by the program. Without `-except precise`, there is a possibility that the location of an arithmetic exception might be reported incorrectly or lost.

The code generated under `-except precise` is specific to the target architecture for which you are compiling and is only guaranteed to work for that architecture. The target architecture is determined by the `-tm` option. If `-tm` in not used, the default is the machine on which you are compiling.

Use `-except precise` only when absolutely necessary. `-except precise` causes additional instructions to be inserted before every return, which degrades performance.

`-except default` cancels the effects of `-except precise`. This is provided mainly for overriding an `-except precise` supplied in the `CCOPTIONS` environment variable.

Summary Of New Options

The following table contains a summary of the new options available in CONVEX C Version 5.0.

Option	Purpose
`-alias restrict_args`	Treat all pointer and array function arguments as if the `restrict` qualifier were used.
`-except precise`	Ensures that any arithmetic exceptions issued by a subprogram before it returns will be received by the subprogram
`-except default`	Cancel `-except precise`
`-nptr`	Disable procedural pointer tracking

4. Documentation changes

The documentation package for this release of CONVEX C consists of this document, the *Installation Instructions*, and the manuals listed below. The Full Documentation Kit will accompany the fully optimizing version of the compiler. Scalar compiler customers will receive the Scalar Documentation Kit.

The documentation contains one of each item mentioned in the tables below.

Release Package:

C compiler documentation			
	Order number	Part number	Description
1	DSW-085	720-001022-200	CONVEX C Compiler Full Documentation Kit
OR			
1	DSW-044	720-002622-000	CONVEX C Compiler Scalar Documentation Kit
2	*	720-002130-013	CONVEX C Installation Procedures, V5.0
3	*	720-002030-017	CONVEX C Release Notice, V5.0

* Installation Procedures and Release Notices do not have order numbers.

Documentation Kit:

	Order number	Description
1	DSW-086	CONVEX C Guide, Second Edition
2	DSW-087	CONVEX C Quick Reference Guide, Sixth Edition
3		The C Programming Language, Second Edition
4	DSW-089	CONVEX C Optimizing Guide, Third Edition (Full Documentation Kit only)
5	DSW-043	CONVEX Interlanguage Programming Guide, First Edition

Appendix A

Compiler problems which produce invalid results

This appendix includes sample source code or a concise description for compiler problems that are known to produce invalid results.

When this program is compiled at any optimization level with CONVEX C, it produces incorrect results. The problem is with functions that return structures.

If all the following conditions occur.

- There is more than one actual parameter in a parameter list.
- Both actual parameters are calls to the same function.
- The function that is called in the actual parameter expression returns a structure.

Then the wrong arguments get passed. The return values from the structure-returning functions overwrite each other so the one that was called second is passed for both arguments.

WORKAROUND:

Compile using the option `-compat rrf=stack` or assign one of the function return values to a local variable and pass the local variable.

```
struct test { int x, y, z; };

void pass(struct test p1, struct test p2)
{
  printf("p1.x = %d p1.y = %d p1.z = %d\n", p1.x, p1.y, p1.z);
  printf("p2.x = %d p2.y = %d p2.z = %d\n", p2.x, p2.y, p2.z);
}

struct test makeStruct(int p)
{
  struct test result;
  result.x = 0; result.y = p; result.z = 0;
  return result;
}

main()
{
  pass(makeStruct(5), makeStruct(10));
}
```

Appendix B

Fixed bugs

This section lists:

- Bugs which have been fixed
- Reported bugs which were not really bugs
- Wishes which have either been implemented, or are unlikely to ever be implemented.

The (P) column will contain a "P" if this bug or wish was actually resolved in a previous release of the C compiler, but not indicated as such in the release notice for that release.

CPU #	PR #	X #	(P)	Symptom of fixed problem
6	20773	20293	P	Man page error
23	24036	22965		Compiler terminates abnormally
57	28851	26859	P	Compiler terminates abnormally
109	20154	9267		Executable generates wrong answers
147	17488	17477	P	Source debugger interface problem
147	17489	17478	P	Source debugger interface problem
147	17491	17480	P	Source debugger interface problem
147	17492	17481	P	Source debugger interface problem
147	17493	17482	P	Source debugger interface problem
147	17974	17870	P	Source debugger interface problem
147	17979	17872	P	Source debugger interface problem
147	19773	19433	P	Installation procedures problem
147	20447	20056	P	Compiler terminates abnormally
147	20841	20327		Long compile time
147	23591	22559		Installation procedures problem
147	26378	24950	P	Loader interface problem
173	9847	11408		Enhancement request
209	13498	14100		No optimization performed
223	18219	9267		Executable generates wrong answers
223	24054	22982	P	Installation procedures problem
228	18910	18817		Miscellaneous
228	22004	21236		Installation procedures problem
228	25153	22608		Installation procedures problem
303	27427	25798		Compiler terminates abnormally
394	26764	25254	P	Enhancement request
8201	7150	9267		Executable generates wrong answers
8201	15034	15303		Executable generates wrong answers
8201	16157	16334		Compiler terminates abnormally
8201	17013	17092		Compiler accepts invalid program
8201	18426	18218	P	Run-time library problem
8201	19432	19198		Compiler accepts invalid program
8201	19574	19331		C source utilities
8201	19794	19450		Executable generates wrong answers
8201	19935	19578		Compiler terminates abnormally

CPU #	PR #	X #	Symptom of known software problem
8201	15366	15610	Compiler accepts invalid program
8201	19830	19470	Miscellaneous
8201	23363	22322	Miscellaneous
8201	23370	22327	Compiler generates incorrect message
8201	23332	22370	Compiler generates incorrect message
8201	23995	22915	Compiler terminates abnormally
8201	24006	22929	C source utilities
8243	21232	20638	Compiler rejects valid program
8529	26938	20766	Compiler generates incorrect message
8534	16562	20766	Compiler generates incorrect message
8534	22050	21294	C source utilities