Intel Corporation
5200 N.E. Elam Young Parkway
Hillsboro, OR 97124-6497

(503) 696-8080

March, 1994

Dear Paragon™ Customer:

This package contains Release 4.5 of the Paragon™ C compiler for your Paragon™ system.

Before using your system:

- **Read this letter completely.**
- **Verify the contents of this package.**
- **Read the *Paragon™ C Compiler Software Product Release Notes.***

Package Contents

Your Paragon™ C compiler software package should include the items listed in Table 1 (Installation Media) and Table 2 (Documentation). If any items are missing, or if you have any questions, please contact Intel Supercomputer Systems Division as described in the "Comments and Assistance" section.

Your package should contain one (and only one) of the cartridge tapes listed in Table 1 (depending on your host).

Table 1. Installation Media

Description	Order Number
Paragon™ C Compiler Release 4.5 Native and Sun4-Hosted	313009-001
Paragon™ C Compiler Release 4.5 Silicon Graphics-Hosted	313015-001

Table 2. Documentation

Description	Order Number
Paragon™ C Compiler Software Product Release Notes	313008-001
Paragon™ C Compiler User's Guide	312490-002
*Basic Math Library Performance Report**	312936-001

*The *Basic Math Library Performance Report* is included only in the Native and Sun4-hosted compiler package.

Restrictions and Limitations of Compiler Release 4.5

Every effort has been taken to ensure the quality of this release, but at shipping time we are aware of a few problems. Please refer to the *Paragon™ C Compiler Software Product Release Notes* for known limitations and workarounds.

Installation

For directions on how to install your Paragon™ C compiler, refer to Chapter 5 of the *Paragon™ C Compiler Software Product Release Notes.*

NOTE

You must have System Software Release 1.1.4 or greater installed on your system in order to install Compiler Release 4.5.

Comments and Assistance

Intel Supercomputer Systems Division is eager to hear of your experiences with the Paragon™ C compiler. Please call us if you need assistance, have questions, or otherwise want to comment on your Paragon™ system.

U.S.A./Canada Intel Corporation
Phone: 800-421-2823
Internet: support@ssd.intel.com

Intel Corporation Italia s.p.a.
Milanofiori Palazzo
20090 Assago
Milano
Italy
1678 77203 (toll free)

France Intel Corporation
1 Rue Edison-BP303
78054 St. Quentin-en-Yvelines Cedex
France
0590 8602 (toll free)

Intel Japan K.K.
Supercomputer Systems Division
5-6 Tokodai, Tsukuba City
Ibaraki-Ken 300-26
Japan
0298-47-8904

United Kingdom Intel Corporation (UK) Ltd.
Supercomputer System Division
Pipers Way
Swindon SN3 IRJ
England
0800 212665 (toll free)
(44) 793 491056 (*answered in French*)
(44) 793 431062 (*answered in Italian*)
(44) 793 480874 (*answered in German*)
(44) 793 495108 (*answered in English*)

Germany Intel Semiconductor GmbH
Dornacher Strasse 1
85622 Feldkirchen bei Muenchen
Germany
0130 813741 (toll free)

World Headquarters
Intel Corporation
Supercomputer Systems Division
15201 N.W. Greenbrier Parkway
Beaverton, Oregon 97006
U.S.A.
(503) 629-7600 (Monday through Friday, 8 AM to 5 PM Pacific Time)
Fax: (503) 629-9147

If you have comments about our manuals, please fill out and mail the enclosed Comment Card. You can also send your comments electronically to the following address:

techpubs@ssd.intel.com (Internet)

Users' Group

The Intel Supercomputer Users Group promotes the exchange of information among users. Intel strongly supports the Users Group and encourages participation in its activities, which include: Special Interest Groups (SIGs), an annual international users conference, an electronic mail task force, and a "freeware" library of user-contributed software, available electronically to all members of the Intel Supercomputer Users' Group. For membership information contact:

JoAnne Wold (503-629-5322)
joanne@ssd.intel.com (Internet)

Sincerely,

Steve Cannon

Steve Cannon

Product Marketing Manager
Intel Supercomputer Systems Division

March 1994

Order Number: 313008-001

Paragon™ C Compiler

Release 4.5

Software Product Release Notes

Intel® Corporation

286	i386	Intel	iPSC
287	i387	Intel386	Paragon
Concurrent File System	i486	Intel387	ProSolver
Direct-Connect Module	i487	Intel486	
i	i860	Intel487	

WARNING

Some of the circuitry inside this system operates at hazardous energy and electric shock voltage levels. To avoid the risk of personal injury due to contact with an energy hazard, or risk of electric shock, do not enter any portion of this system unless it is intended to be accessible without the use of a tool. The areas that are considered accessible are the outer enclosure and the area just inside the front door when all of the front panels are installed, and the front of the diagnostic station. There are no user serviceable areas inside the system. Refer any need for such access only to technical personnel that have been qualified by Intel Corporation.

CAUTION

This equipment has been tested and found to comply with the limits for a Class A digital device, pursuant to Part 15 of the FCC Rules. These limits are designed to provide reasonable protection against harmful interference when the equipment is operated in a commercial environment. This equipment generates, uses, and can radiate radio frequency energy and, if not installed and used in accordance with the instruction manual, may cause harmful interference to radio communications. Operation of this equipment in a residential area is likely to cause harmful interference in which case the user will be required to correct the interference at his own expense.

LIMITED RIGHTS

Preface

These release notes provide the latest information on Release 4.5 of the Paragon C compiler.

These release notes assume that you are an application programmer proficient in the ANSI C language and the UNIX operating system.

Organization

Chapter 1	Introduces the new features of the Release 4.5 C compiler.
Chapter 2	Describes resolved limitations for this release.
Chapter 3	Describes known limitations and workarounds of this release.
Chapter 4	Provides some hints and suggestions for using the compiler.
Chapter 5	Provides installation instructions for this release.

Notational Conventions

This manual uses the following notational conventions:

Bold	Identifies command names and switches, system call names, reserved words, and other items that must be used exactly as shown.
Italic	Identifies variables, filenames, directories, processes, user names, and writer annotations in examples. Italic type style is also occasionally used to emphasize a word or phrase.

`Plain-Monospace`
: Identifies computer output (prompts and messages), examples, and values of variables. Some examples contain annotations that describe specific parts of the example. These annotations (which are not part of the example code or session) appear in *italic* type style and flush with the right margin.

`Bold-Italic-Monospace`
: Identifies user input (what you enter in response to some prompt).

`Bold-Monospace`
: Identifies the names of keyboard keys (which are also enclosed in angle brackets). A dash indicates that the key preceding the dash is to be held down *while* the key following the dash is pressed. For example:

`<Break>` **`<s>`** **`<Ctrl-Alt-Del>`**

[ ] (Brackets) Surround optional items.

... (Ellipsis dots) Indicate that the preceding item may be repeated.

| (Bar) Separates two or more items of which you may select only one.

{ } (Braces) Surround two or more items of which you must select one.

Applicable Documents

For more information, refer to the *Paragon™ System Technical Documentation Guide.*

Comments and Assistance

Intel Supercomputer Systems Division is eager to hear of your experiences with our products. Please call us if you need assistance, have questions, or otherwise want to comment on your Paragon system.

U.S.A./Canada Intel Corporation
Phone: 800-421-2823
Internet: support@ssd.intel.com

Intel Corporation Italia s.p.a.
Milanofiori Palazzo
20090 Assago
Milano
Italy
1678 77203 (toll free)

France Intel Corporation
1 Rue Edison-BP303
78054 St. Quentin-en-Yvelines Cedex
France
0590 8602 (toll free)

Intel Japan K.K.
Supercomputer Systems Division
5-6 Tokodai, Tsukuba City
Ibaraki-Ken 300-26
Japan
0298-47-8904

United Kingdom Intel Corporation (UK) Ltd.
Supercomputer System Division
Pipers Way
Swindon SN3 IRJ
England
0800 212665 (toll free)
(44) 793 491056 (*answered in French*)
(44) 793 431062 (*answered in Italian*)
(44) 793 480874 (*answered in German*)
(44) 793 495108 (*answered in English*)

Germany Intel Semiconductor GmbH
Dornacher Strasse 1
85622 Feldkirchen bei Muenchen
Germany
0130 813741 (toll free)

World Headquarters
Intel Corporation
Supercomputer Systems Division
15201 N.W. Greenbrier Parkway
Beaverton, Oregon 97006
U.S.A.
(503) 629-7600 (Monday through Friday, 8 AM to 5 PM Pacific Time)
Fax: (503) 629-9147

If you have comments about our manuals, please fill out and mail the enclosed Comment Card. You can also send your comments electronically to the following address:

techpubs@ssd.intel.com

Table of Contents

Product Features 1

NOTE

Report any problems you encounter while using the Release 4.5 C Compiler software to SSD Technical Support at:

1-800-421-2823 (Customer Support Response Center)
Your Local Intel Sales Office (in Europe)
support@ssd.intel.com (Internet address)

Release 4.5 Features

- There are several new compiler switches and some changes to the existing compiler switches. For a description of the new switches and the changes to existing switches, refer to the section *Compiler Switch Changes*.

- A new vectorization switch has been added, **-Mvect=streamlim:***n*, to allow you to control the minimum size of vectors for which data streaming optimizations are performed. For a complete description of this switch, refer to the section *Compiler Switch Changes*.

- New loop splitting switches are available. **-M[no]split_loop_ops** and **-M[no]split_loop_refs** allow you to set thresholds for loop splitting. For a complete description of these switches, refer to the section *Compiler Switch Changes*.

- Some changes have been made to the scope rules for C pragmas. Refer to the section *Scope of C Pragmas* for a complete description of the scope changes.

- The environment variable *MAKECPP* is supported. *MAKECPP* is a colon-separated list of directories that is added to the compiler's search path for include files.

- The **-X** switch is now available to control the level of ANSI C conformance. For a complete description of the **-X** switch, refer to the section, *Compiler Switch Changes*.

- The following predefined macro names have been added:

  MACH

  CMU

  __I860__

  _I860_

  __I860

  _i860_

  OSF1_ADFS

  OSF1AD

- The compiler now supports **signed long** and **unsigned long** as separate data types, instead of as synonyms for **signed int** and **unsigned int** respectively.

- The path names for tools and libraries are no longer partially hardcoded. The C Compiler driver now executes the first **ic**, **ld860**, **as860**, and so on that it finds in your path. To use the R4.5 compilers and tools you must add the directory where they reside to your *PATH* environment variable.

- *libnx.a* is no longer automatically linked in if **-Mperfmon** is in effect. For Unix applications that use any of the nx functions such as **dclock**, **-lnx** should be added to the driver or linker command line when these applications are built.

- The search path for libraries has changed. If *PARAGON_LPATH* or *LPATH* are defined, these directories are searched before any directories added with the **-L** switch.

- This release has been validated by NIST.

- Improvements have been made to the **-Minfo** option to accurately detail the optimizations being performed.

- A global default configuration file is now supported. The compiler searches the following directories in the order listed for the *.icfrc* file.

  1. your current working directory

  2. your home directory

  3. the directory where the compiler driver resides

- The installation instructions for the cross-development environment now include instruction to allow a site to create a new cross-dev directory without having to make copies of system libraries and include files.
- A beta library containing some vector functions that have been optimized for the Paragon X/P has been added. Refer to the section *Beta Vector Library* for complete information.

Compiler Switch Changes

This section outlines the new compiler switches that have been added in this release and changes to the existing switches. The new switches are:

-Msplit_loop_ops=*n*
: Set a threshold of *n* floating-point operations within a loop. Innermost loops whose number of floating-point operations exceeds *n* are split. Each floating-point operation counts as two. The default for *n* is 40 when **-Mvect** is used.

-Mnosplit_loop_ops
: Do not split loops when the floating-point operation threshold is exceeded. When **-Mvect** is specified, innermost loops whose number of floating point operations exceed 40 are split by default. This switch turns the default off.

-Msplit_loop_refs=*n*
: Set a threshold of *n* array element loads and stores within a loop. Innermost loops whose number of loads and stores exceeds *n* are split. The default for *n* is 20 when **-Mvect** is used

-Mnosplit_loop_refs
: Do not split loops when the array element loads and stores threshold is exceeded. When **-Mvect** is specified, innermost loops whose number of array element loads and stores exceeds 20 are split by default. This switch turns the default off.

-Mvect=streamlim:*n*
: This sets a limit for application of the vectorizer data streaming optimization. If data streaming requires cache vectors of length less than *n*, the optimization is not performed. Other vectorizer optimizations are still performed. The data streaming optimization has a high overhead compared to other loop optimizations, and can be counter-productive when used for short vectors. The *n* specifier is not optional. The default limit is 32 elements if **streamlim** is not used.

-nostdinc Equivalent to **-Mnostdinc**.

-X(a | c | s | t | l | o)
Specify the degree of ANSI C conformance.

a	ANSI mode. The compiled language conforms to all ANSI features. _ _STDC_ _ is defined to be zero.
c	Conformance mode. The compiled language conforms to ANSI C, but warnings may be produced about some extensions. _ _STDC_ _ is defined to be one.
s	Pre-ANSI mode. The compiled language includes all features compatible with the C language as defined in *The C Programming Language*, by Kernighan and Ritchie (pre-ANSI C). The compiler warns about all language constructs that differ between ANSI C and pre-ANSI C.
t	Transition mode. This is ANSI C plus pre-ANSI C compatibility extensions without the semantic changes required by ANSI C. Where ANSI C and pre-ANSI C specify different semantics for the same construct, the compiler issues a warning and uses the pre-ANSI C interpretation.
l	Treat [un]signed int and [un]signed long as the same data type. When you use this switch, debug records for [un]signed long are type [un]signed int.
o	Execute the R4.1.1 version of **ic**.

By default _ _STDC_ _ is defined to be one and ANSI conformance is relaxed.

There have also been changes to some of the existing switches.

- **-I-** is accepted but has no effect.
- **-Mnoreentrant** is no longer ignored.
- **-Mnostride0** is now the default.

Default Compiler Switch Settings

The default compiler switch settings are set for ease of porting, safe optimization, and high-speed compilation. Some of the defaults are:

-O1	Optimization level one
-Mnostride0	Do not check for zero stride induction variables.
-Mnodebug	Debugging disabled
-Mperfmon	Performance monitoring enabled
-Mnoframe	Don't include stack frame pointers on stack
-Kieee	Math conforms to IEEE 754 standard
-Mdepchk	Assume that potential data dependencies exist
-Msplit_loop_ops=40	Split innermost loops whose number of floating-point operations exceeds 40 if **-Mvect** is specified.
-Msplit_loop_refs=20	Split innermost loops whose number of array element loads and stores exceeds 20 if **-Mvect** is specified.

For better performance, you may use values other than the defaults, or change your defaults with a configuration file. For example, some appropriate user-defined defaults might be:

-O2	Optimization level two
-Mnoperfmon	No performance monitoring
-Knoieee	Non-IEEE math, if floating point accuracy is not critical
-Mnodepchk	Assume that no potential data dependencies exist

If you use these suggested values as user-defined defaults, then in order to debug the program you have to override several of them. For example, to debug, you would want to use the **-g** command line switch. The **-g** switch is equivalent to the following:

-O0 -Mframe -Mdebug

For best performance you may need to override the suggested defaults with command line switches such as the following:

-O3 or **-O4**

-Mvect

For more information on **-Mnostride0**, **-Knoieee**, **-Mvect**, and other switches, refer to the *Paragon OSF/1 C Compiler User's Guide.*

NOTE

If your application contains a loop with an induction variable whose increment (stride) is zero, you should add the **-Mstride0** switch to the compiler command line. **-Mstride0** is no longer the default.

Scope of C Pragmas

The scope of C pragmas has changed in this release. For pragmas that allow **loop**, **routine**, and **global** scope, the following rules apply:

loop	Indicates the pragma applies to the next lexical loop. The pragma does not apply to any loops that are enclosed by the next loop. Loop-scoped pragmas are only applied to **do**, **for**, and **while** loops.
routine	Indicates the pragma applies to the code that follows the pragma until the end of the function.
global	Indicates the pragma applies to the code that follows the pragma until the end of the file.

For pragmas where **loop** scope is not allowed, the scope rules fall into two groups.

The following rules apply to pragmas **func32**, **frame**, **opt**, and **safe**:

routine	Indicates the pragma applies to the current function, if it is in a function. If it is not in a function, it applies to the next function.
global	Indicates the pragma applies to all functions that follow it.

The following rules apply to pragmas **bounds**, **fcon**, and **single**:

routine	Indicates the pragma applies to the code that follows the pragma until the end of the function.
global	Indicates the pragma applies to the code that follows the pragma until the end of the file.

For a complete description of C pragmas and their scope rules, refer to the *Paragon™ C Compiler User's Guide.*

Beta Vector Library

When you use the **-Mvect** switch, the compiler usually generates calls to hand-coded library routines to perform certain vector operations. Some of these functions have been optimized for the Paragon system and have been placed in *libvbeta.a*. The amount of improvement you see by using functions from this library depends on the number of times the functions are called in your application. In particular, single-precision applications that use stride one array references should show improvement when you use this library. Some applications of this type had better performance with only software pipelining of loops than with both pipelining and vectorization.

Since these new versions have not undergone rigorous testing, they are provided as a beta library. To use this library, add **-lvbeta** to the driver command line. If you invoke **ld860** directly, insert it before **-lic**.

Resolved Limitations 2

The following problem reports are fixed in this release. The number in brackets following each description is the problem report number.

- Initializing char arrays in their declarations no longer causes core dumps. [5571]
- Pointers to doubles now work properly when declared on the stack. [5819]
- **nm860** no longer fails for libraries containing a zero-length object. [7006]
- Null characters in source code no longer cause a fatal compiler error. [4118]
- **ar** now handles "ar cur archive.a file.o ..." [4930]
- **ld860** now identifies a module for an undefined symbol error. [5019]
- The compiler no longer produces faulty code when old-style arg declarations are used. [7007]
- A global compiler default file is now possible. [6917]
- Using **-Mvect** no longer results in internal compiler errors. [4823]
- **cpp** no longer corrupts or truncates certain comment lines. [4898]
- *.def* now matches the static function (label) being defined. [5353]
- The compiler now reports an error if an argument is declared multiple times. [5426]
- The unmatched #endif (*/usr/include/mach3/mach/mach_traps.h*: 187) is fixed. [5446]

- #ifdef constructs now work within a macro expansion. [6243]
- nx function references are now resolved correctly. [6659]
- **ld860** no longer fails with the error: Bad object module x.o contains undefined symbol. [6995]
- Programs linked with the -contig switch no longer core dump. [5626]
- **cpp** no longer lists *STDIN.c* as the source file for a null file. [6999]
- The **-l** switch now picks up non-default libraries in */usr/lib*. [7083]
- **cosh**(**x**) and **sinh**(**x**) now return the correct value when $709.783 < x < 710.471$. [6365]
- The function **dcos** no longer gives incorrect answers when a large negative input value is specified while pipelining is turned on. [7275]
- Bessel functions no longer core dump when *arg* is between 0 and 298156826. [5794]

Limitations and Workarounds 3

This chapter contains a list of known limitations and workarounds. This list is available in the file */usr/share/release_notes/icc_buglist.ps* on the Paragon XP/S system.

The *buglist* file is updated just before shipment. The file included with your software may not be the latest version. Please contact the SSD Customer Support Response Center for information about how to get the latest version.

PostScript Copies of the Manuals

PostScript copies of the Paragon compiler manuals are available in the directory */usr/share/ps.docs* on the Paragon system. Postscript copies of the release notes are available in the directory */usr/share/release_notes*. The hardcopy version may be more up-to-date than the online version included on the release tapes, but the very latest online version can be obtained by contacting SSD Customer Support.

The Buglist File

The rest of this chapter lists the outstanding bugs. The number on the left side of the first line for each bug is the bug number. Use this number when communicating with SSD Customer Support about the bug. An ASCII version of the *buglist* file is available in the directory */usr/share/release_notes*. The hardcopy version may be more up-to-date than the online version included on the release tapes, but the very latest online version can be obtained by contacting SSD Customer Support.

3264	Line numbers are not generated with **-Mdebug** and default optimization. **icc -Mdebug** will not put line number information in the COFF file for each line number if you use anything over the **-O0** optimization level. **-Mdebug** should cause line numbers to be generated, at least through **-O1**, the default optimization level.
6842	No error message results when a pointer is redeclared as an array of the same type.
6939	No error message results when the "&" operator is applied to an object declared **register**.
7636	The compiler removes unreferenced strings. In a future release, a compiler option will be available to force the compiler to emit the initializations for unreferenced but initialized static variables. For the current release, you can add **-Mx,119,8** to the **icc** command line as a workaround.
7845	**cpp** does not process a string that contains the character '.

Hints and Suggestions 4

This chapter provides some hints and suggestions for making the best use of the compiler.

1. The compiler may occasionally generate internal compiler messages. If they are of severity **W** (Warning) or **I** (Informational), the generated code is correct. However, please report all internal messages to SSD.

2. When using pipelining (**-O4**), the **-Mnodepchk** switch generally increases pipelining opportunities. If the program does not produce correct results with this switch, then it must be omitted. Use the switch only if you are sure no data dependencies that inhibit vectorization exist.

3. The compiler does not check to see that the address of a variable declared **register** is not taken.

4. The compiler conforms to the ANSI Standard, with some minor deficiencies and extensions. All known deficiencies and extensions are exercised in the regression test suite. However, be advised that the following functions behave differently than the ANSI Standard specifies:

   clock　　**system**　　**mktime**

5. Functions that are declared externally within an inner scope are visible to all following code at outer scopes.

6. The **-Mvect=unroll** switch is no longer supported or documented. For backwards compatibility, it is silently ignored if you use it. This switch results in the following warning:

   ```
   icc - Warning - mvect = unroll not implemented
   ```

7. No features are currently enabled by the **-Mbeta** switch.

8. The **-Mstride0** compiler switch should be used if a loop may contain an induction variable whose increment (stride) is zero. For example:

   ```
   is = 0;
   j = 0;
   for (i=1; i<=N; i++) {
      a[j] = b[i]+1.0;
      j = j+is;
   }  /* end for */
   ```

   This switch may degrade performance so should only be used if zero-stride induction variables are possible.

9. The **-Knoieee** switch can give a substantial performance improvement. Division that does not conform to IEEE is several times faster than IEEE division, and some benchmarks run about twice as fast overall with the **-Knoieee** switch set. The penalty you pay for this performance is up to three low order bits of accuracy on certain division operations, and denormals are flushed to zero. The majority of division operations give identical results, whether or not IEEE math is used.

10. If your application runs slower when you use **-Mvect -O4**, try **-Mvect=streamlim:999 -O4**. The additional overhead of streaming in and streaming out data to and from cache could result in decreased performance if the vectors are short.

11. If your application uses only stride 1 array references, you may see increased performance if you use **-Mvect -Mstreamall** and link in *libvbeta.a*.

12. For applications with array references that are not stride 1, you may see increased performance if you use **-Mvect=streamlim:999**.

Installation 5

NOTE

If you must re-install your system software, you must install the compilers at the end of the system software installation. If you installed compilers earlier in the system software installation, you must re-install them after all other system software is installed.

Installing the Native Compilers

Installation Time:	Approximately 45 minutes.
Installation Media:	One 0.25-inch QIC 150 cartridge tape labelled "Paragon™ C Compiler Release 4.5 Native, Sun4, and Solaris-Hosted (313009-001)." The tape contains an installation **tar** file for the native compiler and another for the cross-compiler for Sun-4 workstations. The files also contain the examples and PostScript copies of the manuals.

These instructions assume you are reading the tape on the Paragon diagnostic station, which is running SCO Unix. The tapes were written on a diagnostic station, therefore the *tar* format and the physical characteristics of the tape are compatible with that system. You can, if you prefer, try reading the tape on some other networked system, such as a workstation. If that system does not successfully read the files from the tape, you are advised to read it on the diagnostic station.

1. Log in to the diagnostic station as *root.*

2. Copy the installation **tar** files from the release tape into */u/tmp* on the diagnostic station. Each compressed **tar** file is about 4M bytes. After installation each compiler requires about 8M bytes.

   A. First make */u/tmp* your working directory. Then perform the following steps.

   ```
   DS# cd /u/tmp
   ```

   B. Insert the release tape into the cartridge tape drive on the diagnostic station.

   C. Issue the command,

   ```
   DS# tar xvf /dev/rStp0 native_install nat_c.tar.Z
          icc.doc.tar.Z
   ```

   D. After the file has been copied, remove the tape from the cartridge tape drive.

3. Log in to the Paragon system as *root.*

4. If you have already installed the native Fortran compilers and */tmp/native_install* still exists on your system, you can proceed to step 5.

   Establish an **ftp** connection with the diagnostic station and transfer the following file:

native_install	This file copies the compiler and documentation files from the diagnostic station and installs them in / or an alternate directory.

   On the Paragon system, issue the following commands:

   ```
   # cd /tmp
   # ftp diagnostic station IP address
   ftp> cd /u/tmp
   ftp> get native_install
   ftp> bye
   # chmod 544 native_install
   ```

5. Execute the installation script.

```
# cd /
# /tmp/native_install C
```

The following is displayed. The distribution information is read from */etc/defaults/install* if the file exists:

```
==============================================================
                  Native Compiler Installation
==============================================================

Install C? [y/n]:                                  y
Root directory for compiler installation [path]: /
Distribution Node:                                 myhost
Distribution IP Addr:                              myhost_ip_addr
Distribution Path:                                 /my_default_path
Is this correct? [y/n]:
```

To change any of these values, answer "n" to the "Is this correct?" prompt and enter the desired value when prompted to do so. If you enter <CR> at a prompt, the value is not changed. If you change the value of the root installation directory, and the directory does not exist, you are asked if you want to create it. When you are satisfied with all the values displayed, enter "y" in response to the "Is this correct" prompt.

The files are copied to the Paragon system and installed. The following is an example of the output seen when installing in the directory /.

```
Username for FTP'ing files from roadkill: [anonymous] myname
                  .
                  .
                  .
       FTP output from file transfers
                  .
                  .
                  .
221 Goodbye.
Uncompressing nat_c.tar.Z...
Uncompressing icc.doc.tar.Z...
Installing Native C compiler...
```

```
Native C compiler has been installed
Installing C manual pages...

Installation complete
```

6. Verify that your path is set correctly.

   If the root directory for the install was not /, set *PARAGON_XDEV* to be the root directory you entered, and add *$PARAGON_XDEV/usr/bin* to the beginning of your execution path. You must also add *$PARAGON_XDEV/usr/man* to the beginning of your *MANPATH* environment variable to access the R4.5 manual pages.

   The following should display when you use the compiler **-VV** switch. If it does not, examine your *PATH* environment variable and make any needed corrections.

✓

```
#icc -VV

icc/Paragon Paragon Version 4.5
Copyright 1994, Intel Corporation and The Portland Group Inc.
All Rights Reserved

View $PARAGON_XDEV/usr/share/IFC/release_notes/icc_4.5_release_notes.ps
for a list of new features for Release 4.5
```

✓

7. Execute the installation verification test.

```
# cd root_installation_directory/usr/testinstall
# ./testinstall_c
Installation successful
```

8. Remove the *testinstall* directory.

```
    # cd ..
    # rm -rf testinstall
```

Installing the Cross-Development Compilers

Installation Time:	Approximately 45 minutes.
Installation Media:	One 0.25-inch QIC 150 cartridge tape labelled "Paragon™ C Compiler Release 4.5 Native, Sun4, and Solaris-Hosted (313009-001)." The tape contains an installation **tar** file for the native compiler and another for the cross-compiler. The files also contain the online manual pages, examples, and PostScript copies of the manuals.
Installation Media (SGI):	One 0.25-inch QIC 150 cartridge tape labelled "Paragon™ C Compiler Release 4.5 Silicon Graphics Hosted (313015-001)."

The cross-development tools and compilers are installed by reading in a set of **tar** files from the installation tape onto the diagnostic station. You then **ftp** one or more compressed **tar** files to your workstation or workstation server, where you untar them. Do not install the cross-development compilers on the Paragon.

1. Log in to the diagnostic station as *root.*

2. Copy the installation **tar** files from the release tape into */u/tmp* on the diagnostic station. Each compressed **tar** file is about 4M bytes. After installation each compiler requires about 8M bytes. The total space for all compiler-related files can be as much as 51M bytes.

   A. First, make */u/tmp* your working directory, Then perform the following steps.

   ```
   DS# cd /u/tmp
   ```

B. Insert the release tape into the cartridge tape drive on the diagnostic station.

C. Extract the compressed tar files and installation script. The installation script and documentation files are the same for each host and can be extracted only once.

If you are copying the cross-compiler for a Sun4 workstation, issue the command:

```
DS# tar xvf /dev/rStp0 cross_install sun_c.tar.Z
      icc.doc.tar.Z
```

If you are copying the cross-compiler for an SGI workstation, issue the command:

```
DS# tar xvf /dev/rStp0 cross_install sgi_c.tar.Z
      icc.doc.tar.Z
```

If you are copying the cross-compiler for a Sun4/Solaris workstation, issue the command:

```
DS# tar xvf /dev/rStp0 cross_install sol_c.tar.Z
      icc.doc.tar.Z
```

D. After the files have been copied, remove the tape from the cartridge tape drive.

3. If you have installed new system software, you must copy the system libraries and include files to the cross-development environment. If you have not installed new system software, proceed to step 4.

Log in to the Paragon system as *root*. Then:

```
# cd /tmp
# /usr/bin/mksysfiles
# exit
```

This step may take up to 30 minutes to complete. You will also see the following message. This message should be ignored.

```
rm: sysfiles.tar No such file or directory
```

4. If you have already installed the Fortran cross compiler(s) and */tmp/cross_install* still exists on your system, you can proceed to step 5.

Establish an **ftp** connection with the diagnostic station and transfer the following file:

cross_install	This file copies the compiler(s) and documentation files from the diagnostic station and installs them.

On your workstation, issue the following commands:

```
CROSS# cd /tmp
CROSS# ftp diagnostic station IP address
ftp> cd /u/tmp
ftp> get cross_install
ftp> bye
CROSS# chmod 544 cross_install
```

5. If you do not need to create a new directory for the R4.5 compilers, you can proceed to step 6.

```
CROSS# mkdir directory
```

6. Make the directory in which the cross compiler(s) will be installed your current working directory and execute the installation script.

```
CROSS# cd directory
CROSS# /tmp/cross_install C
```

The following is displayed.

```
===============================================================
                  Cross Compiler Installation
===============================================================

Install C? [y/n]:                                    y
Install Sun4? [y/n]:                                 y
Install Solaris? [y/n]:                              y
Install SGI? [y/n]:                                  y
Root directory for compiler installation [path]:  $PARAGON_XDEV
Location for compressed tar files [path]:            /tmp
Distribution host name:                              unknown
Distribution host user name for ftp:                 anonymous
Distribution Path:                                   /u/tmp
Install system files? [y/n]:                         n
Create links for system files? [y/n]:                n
Is this correct? [y/n]:
```

If you respond with "y" to the "Install system files" prompt, you are asked for the following information:

```
Enter name of paragon system where sysfiles.tar.Z was created:
Enter user name for ftp from your_system: [anonymous]
Enter path of sysfiles.tar on your_system: [/tmp]
```

If you respond with "y" to the "Create links for system files" prompt, you are asked for the following information:

```
Enter root directory of actual files:
```

To change any of these values, answer "n" to the "Is this correct?" prompt and enter the desired value when prompted to do so. If you enter <CR> at a prompt, the value is not changed. When you are satisfied with all the values displayed, enter "y" in response to the "Is this correct" prompt.

All of the compressed tar files needed for your installation selections are copied to the location you specified. Each file is uncompressed and installed, and the uncompressed tar files are deleted. If any of the uncompress or tar commands fail, the installation is aborted. The most likely cause for a failure is lack of disk space. If this occurs, you may need to install one compiler at a time.

In the following example, the user installs the Sun4 compiler, the Sun4/Solaris compiler, the SGI compiler, and the system libraries and header files.

✓

```
CROSS# /tmp/cross_install C

===================================================================
                  Cross Compiler Installation
===================================================================

Install C? [y/n]:                                     y
Install Sun4? [y/n]:                                  y
Install Solaris? [y/n]:                               y
Install SGI? [y/n]:                                   y
Root directory for compiler installation [path]: /vol/scratch/install
Location for compressed tar files [path]:        /vol/scratch/tmp
Distribution host name:                          fred
Distribution host user name for ftp:             root
Distribution Path:                               /u/tmp
Install system files? [y/n]:                     y
   Paragon system name:                          ethel
   Paragon user name for ftp:                    root
   Path for compressed system tar file:          /tmp
Create links for system files? [y/n]:            n
Is this correct? [y/n]:                          y

Connected to fred.
220 roadkill FTP server (SunOS 4.1) ready.
331 Password required for root.
```

```
Password:
230 User root logged in.
200 Type set to I.
Local directory now /vol/scratch/temp
200 PORT command successful.
                  .
                  .
                  .
```

ftp files from distribution system

```
                  .
                  .
                  .
221 Goodbye.
Connected to ethel.
220 ethel FTP server (OSF/1 Version 5.60) ready.
331 Password required for root.
Password:
230 User root logged in.
                  .
                  .
                  .
```

ftp sysfiles.tar.Z from Paragon system

```
                  .
                  .
                  .
221 Goodbye.
Uncompressing sysfiles.tar.Z...
Installing system libraries and header files...
Uncompressing icc.doc.tar.Z...
Installing C manual pages...
Uncompressing sun_c.tar.Z...
Installing Sun C compiler...
Uncompressing sol_c.tar.Z...
Installing Sun4/Solaris C compiler...
Uncompressing sgi_c.tar.Z...
Installing SGI C compiler...

Installation complete
CROSS#
```

7. Verify that your path is set correctly.

Set *PARAGON_XDEV* to be the root directory you entered, and add *$PARAGON_XDEV/paragon/bin."arch"* to the beginning of your execution path. For example, on Sun4/Solaris systems you would add *$PARAGON_XDEV/paragon/bin.solaris.* You must also add *$PARAGON_XDEV/paragon/man* to the beginning of your *MANPATH* environment variable to access the R4.5 manual pages.

The following should display when you use the compiler -VV switch. If it does not, examine your *PATH* environment variable and make any needed corrections.

```
#icc -VV

icc/Paragon "host" Version 4.5
Copyright 1994, Intel Corporation and The Portland Group Inc.
All Rights Reserved

View $PARAGON_XDEV/paragon/release_notes/icc_4.5_release_notes.ps
for a list of new features for Release 4.5
```

✓ 8. Execute the installation verification test. The *testinstall_c* script requires the name of the paragon system where the test will be executed as an argument. You must be able to execute **rcp** and **rsh** commands on the Paragon system you specify.

```
# cd $PARAGON_XDEV/paragon/testinstall
# ./testinstall_c paragon_system
Installation successful              ✓
```

9. Remove the *testinstall* directory.

```
# cd ..
# rm -rf testinstall
```