Release Notes
Harvest C, Shareware version
Release 1.1b2[1*]
24 November 1991

Harvest C is a shareware C compiler for the Macintosh. An integrated development environment, Harvest C is intended to be compatible with the MPW C compiler. Harvest C compiles (almost) all the same extensions to ANSI C which are supported by MPW C. Harvest C also generates object files which are compatible with MPW.

NOTICE

Throughout this notice, the "SOFTWARE" refers to **this version** and **only this version** of Harvest C, (including all accompanying documentation, notices, source files, executables, and release notes) and no other releases or products. The following statements reflect the legal conditions of the use of this SOFTWARE. If the terms expressed below are unacceptable to you, then you should not use the SOFTWARE.

1. The SOFTWARE is Copyright 1991 Eric W. Sink, All Rights Reserved. Harvest C is NOT public domain.

2. You are explicitly granted the right to use and distribute the SOFTWARE freely and publicly, as long as all copyrights, notices, release notes, documentation, source files and executables are included and unchanged from the original release made by the author. You may not distribute modified copies of the SOFTWARE.

3. You may not falsely represent the author or copyright holder of the SOFTWARE, either by omission or misrepresentation. The author of this SOFTWARE is Eric W. Sink. Currently the copyright holder is also Eric W. Sink.

4. The SOFTWARE is referred to as "shareware". Because the term "shareware" does not have a strict legal definition, the following clarifications make the terms more specific :

a. You may register yourself (and are encouraged to do so) as a user of the SOFTWARE, by sending a fee of US $25 to :

Eric W. Sink
1014 Pomona Drive
Champaign, Illinois 61821

b. You are under no legal obligation to register. In general, it is recommended that you register and pay for any shareware which you use. This notice is merely intended to make the legal aspect of shareware registration more specific for this SOFTWARE.

5. The SOFTWARE is provided with absolutely NO WARRANTY, under any conditions. In no case will the author, copyright holder, or any distributor be in any way liable for any damages or difficulties of any kind, either directly or indirectly caused by any kind of use of this SOFTWARE. The author, copyright holder and/or distributor of this

[1*] The version number 1.1a1 indicates that this is a test version of Harvest C. This means that this release is considered Freeware. No shareware registration fee is requested, but the copyright remains identical. You areencouraged to report any problems, and use this version at your own risk.

FUTURE DIRECTIONS

This shareware version of Harvest C is being released to fill a perceived need for a freely distributable C compiler for the Macintosh. I intend to maintain it, and make subsequent releases with bug fixes and new features. It is my intent to offer a commercially available version in the future. The differences between that commercial version and the shareware version have not been chosen, but it is my intent to maintain both versions. The shareware version will continue, and will aim to be a usable C development environment for Macintosh users. The following features are under consideration. Some of them are already implemented. Some of them will find their way into the shareware release. Some will be available only in the commercial version.

- 68020, 68881 code generation
- support for > 32k of global data
- precompiled headers
- SADE support
- optional 16/32 bit ints
- inline assembler
- more complete documentation
- increased speed
- reporting of software metrics
- support for object oriented programming
- full source code to the compiler included
- better code generation
- better linking
- full System 7 friendliness

REGISTRATION

You are encouraged to register yourself as a user, as described above. Registered users will receive a discount on purchase of the commercial version when it is released. In addition, registered users will obtain my attention much faster with respect to questions, fixes, desired features and the like. You may register yourself by supplying the following information :

Name:
Address (city, state, zip, and country):
Phone (optional):
email address (if you have one):
Age (optional):

Suggestions for improvements:

Enclose a check for US $25 to :

Eric W. Sink

1014 Pomona Drive
Champaign, Illinois 61821

I will acknowledge all registrations.

CONTACTS

You are encouraged to contact me with any questions, comments, suggestions or complaints. You may reach me by electronic mail (USENET,Internet) at

e-sink@uiuc.edu

or by US mail at

Eric W. Sink
1014 Pomona Drive
Champaign, Illinois 61821

I am not currently on AppleLink or CompuServe.

USING HARVEST C

Harvest C includes a built-in text editor. This editor offers MINIMAL functionality, and is not intended to be used as a principal programmer's editor. The Harvest C editor does not support, tabs, undo, multiple fonts, searching, macros, or files longer than 32000 characters. (For general purpose editing, I recommend the use of Alpha, an excellent shareware programmer's editor, widely available at ftp sites.) In any case, you may use the text editor in the normal way, in conjunction with the File and Edit menus.

The Options menu allows control over some of the features of Harvest C. Some of the options are not enabled in the shareware version. Harvest C stores its configuration in a file called "Harvest C Options", in the System Folder. It is necessary to inform Harvest C of the locations of your include files, and libraries (see below). You may specify these using the entries under the Options menu. Harvest C maintains the notion of an "Application directory". This is the directory in which you do most of your work. This may be set as an option as well. You may (and should) save these choices as the defaults by using the Save Defaults entry. Harvest C reads the "Harvest C Options" file from the System Folder when the application is opened. If it has trouble reading the file, it initializes all options to a set of reasonable defaults, and beeps. This beep is to let you know that the current option defaults are not yours, and that you should select the defaults you prefer and save them. Harvest C should beep the first time you run it.

The Options menu has three hierarchical menus which allow a number of different settings. The most important for the shareware version is the Warnings menu. This menu allows you to turn each of the various warnings on or off. A check mark appears next to the warnings which are active.

The Project menu allows compilation. You may compile a single file using the Compile entry. Execution of a Harvest C build script is done with the Build entry. User specified linking can be accomplished with the Link entry.

Compile...
This option simply allows you to compile a single file. Harvest C presents a file dialog, asking

you to specify which file you wish to compile. The corresponding object file is created, and user information is displayed in a new text window.

Build...
Build scripts are the equivalent of Makefiles in MPW, and Project files in Think C. A build script describes an entire project by listing the files and libraries which are to be included in the project. Harvest C build scripts are very flexible, easily allowing any combination of compiler options for any file or set of files. Upon selecting this option, Harvest C presents a file dialog, asking you to specify a build script. You must create your build script by hand, but fear not, it is not difficult, and a future version will have the ability to create it for you.

The Build... menu item does a full rebuild of the program specified. The Make... menu item is identical, except that it rebuilds only that which is necessary, based on the modification dates of the files. For example, say that your build script contains 5 source files, and you build your program. Then, you modify one source file. Issuing the Make... command here will recompile only that source file. Selecting the Build... command will recompile all 5 source files. Both commands require that the program be relinked.

Format of Build files

Each line of a build file may contain zero or more options, and zero or more filenames. Filenames are assumed to be things which should be linked into the final application. Filenames ending in '.c' are compiled. Options which appear on the same line as one or more filenames are in effect for only those files. Options which appear alone (except for preprocessor definitions) are in effect for the remainder of the build. Options are specified in an MPW like format. On a given line, all characters after a cross hatch character (#) are ignored (for commenting). Furthermore, any blank line causes execution of the script to cease (i.e. blank lines are not allowed in the middle of the file). Legal options are (for the shareware version)

-nolink	Compiles but does not link
-WonXXX	Turn warning number XXX on
-WoffXXX	Turn warning number XXX off
-progress	Turns progress reporting on
-signedchar	Toggles the sign of char types
-w	No warnings
-U$$$$	#undefs $$$$
-D$$$$=@@	#define $$$$ @@
-o name	Sets output file name (no spaces allowed)
-c ????	Sets output file creator signature (no quotes, i.e. no spaces)

When executing a build script, Harvest C looks in the standard application directory for C source files needing compilation. For object files needed for the link, Harvest C looks first in the application directory, then in the standard library directory.

Link...
Using this option you may link any group of files you like. Harvest C displays a dialog for selection of object files for the link. Select as many as you like, clicking the Add button after each one. After the last file, click the Done button. Then Harvest C asks you (with a dialog) for the destination of the link output (the application you will be generating). User information from the link appears in a new text window.

Warnings

This version of Harvest C supports the following warnings. Warnings appearing in bold are turned on by default.

1	Empty expression stmt
2	Multi-character constant
3	Redundant cast
4	**Equivalence test of floating type**
5	Discarded function result
6	**Assignment as if conditional**
7	Non-void function has no return statement
8	Constant expression as if condition
9	Comparison of pointer and integer
10	**Assignment of nonequivalent type to a pointer**
11	Return of nonequivalent type to a pointer
12	Pass of nonequivalent type to a pointer
13	Constant expression as switch condition
14	Constant expression as while loop condition
15	Constant expression as do-while loop condition
16	Constant expression as for loop condition
17	Switch expression not of integral type.
18	Volatile is not handled by this compiler
20	**Unused variable:**
21	**Dead code**
22	**Implicit decl :**
23	Re#definition :
24	Possible nested comment
26	**goto statement found**
27	**Multiple function returns**
28	**Empty compound statement**
29	Missing function return type - default to int
30	Trigraph found
32	pascal keyword found
34	Semicolon after function body

Miscellaneous

It is possible to abort compilation by pressing and holding Command-period.

Harvest C supports Balloon Help under System 7.

Pragmas

Harvest C supports two pragmas from MPW C, `parameter` (for use by the include files) and `segment`. In other words, it does not support `once`, `trace`, `processor`, `force_active`, `push`, or `pop` (or any others which I missed), yet.

The `parameter` pragma should be of no use to normal programmers.

The `segment` pragma is used for code resource segmentation :

```
#pragma segment Foo
```

means that all subsequent code goes in the segment named Foo.

Required Memory

Harvest C likes to have a lot of memory. This is due in part to some inefficient memory management, and should be improved in the future. Meanwhile, try to keep your C source files small. Keep in mind, that even a tiny C source file effectively becomes huge if it #includes several standard header files.

Extensions to ANSI C

Harvest C attempts to be an ANSI C compiler. The various ANSI specific features are all supported, including but not limited to function prototypes, trigraphs, stringization, token concatenation, string literal concatentation, and so on.

Harvest C attempts to support the same extensions to ANSI C which are supported by Apple with their MPW C compiler. Specifically, the following features are implemented :

Harvest C supports the keywords `extended`, `pascal`, and `comp`. The `extended` keyword is taken as synonym for `long double`. The `comp` keyword is a 64 bit integer data type. Harvest C accepts `comp` declarations but does not currently support the `comp` data type.

The `pascal` keyword is used for defining functions with pascal calling conventions. For example :

```
pascal int foo(short bar) {
 return bar * 5;
}
```

Harvest C may reserve the keyword `asm` as well. This will be used in the future to support inline assembly code.

Harvest C supports pascal style string literals. For example,

```
char *s = "\pWelcome";
```

will result in a string which begins with a length byte of 7 and **is** null terminated.

Harvest C predefines the following preprocessor symbols for conditional compilation :

```
    MC68000
    mc68000
    m68k
    macintosh
    applec
```

These are essentially the same such symbols as are defined by MPW C.

Enumerated types are always treated as `int`. Harvest C sizes all enumerated types at the size of an `int` instead of adjusting their size based on the number of member constants.

Harvest C accepts but ignores the keyword `volatile`.

Harvest C accepts `register` class declarations of automatic variables, and will attempt to honor such requests. This behavior is not guaranteed for future versions.

Harvest C supports direct functions like MPW C :

```
pascal void PenSize(short width, short height) =
0xA89B;
```

Sizes of data types in bytes are as follows :

char	1
short	2
int	4
long	4
pointers	4
float	4
double	8
extended	10

Registers D0,D1,D2,A0, and A1 are scratch registers.

The total size of all global variables, static variables, string constants, and floating point constants may not exceed 32K bytes. Floating point constants are of type `extended`.

Harvest C accepts multiple character constants, like `'APPL'`.

Interfaces & Libraries

In order for your program to be linked to form an application, you must link with some standard libraries. These libraries allow access to Toolbox routines, handle initialization of global variables, and things like that. Furthermore, use of these libraries will generally require that you use the interface (or header) files which accompany them.

You must obtain Apple's headers and libraries in order to use Harvest C. I do not yet have legal permission to distribute them. They are sold from the Apple Programmers and Developers Association, and are called "MPW Toolbox Interfaces and Libraries v. 7.0", product number M0615LL/B, at a cost of $40. You may reach APDA at :

APDA
Apple Computer, Inc.
20525 Mariani Avenue
Mail Stop 33G

Cupertino, CA 95014-6299

You may also obtain the necessary headers and libraries by anonymous ftp from ftp.apple.com. You need the file `cincludes-mpw3-2.hqx`, in directory `/dts/mac/system7/7.0.mpw/interfaces.` This is a Binhexed, Stuffit archive containing the header files. In addition, you need the files `clibraries.hqx` and `libraries.hqx,` stored in directory `/dts/mac/system7/7.0.mpw/libraries`. Obtaining the files by anonymous ftp is not difficult. You must first connect with Apple's computer, by typing the following at the shell prompt of your UNIX system.

```
ftp ftp.apple.com
```

You should see something like

```
220 bric-a-brac.apple.com FTP server (IG Version 5.90 (from BU,
from UUNET 5.51)...

Name (bric-a-brac.apple.com:yourlogin):
```

Enter your name as `anonymous` (don't misspell it), and enter your full email address as the password. Then change to the proper directory by typing

```
cd /dts/mac/system7/7.0.mpw/interfaces
```

and obtain the interfaces by entering

```
get cincludes-mpw3-2.hqx
```

Then,

```
cd /dts/mac/system7/7.0.mpw/libraries
```

and obtain the interfaces by entering

```
get clibraries.hqx
get libraries.hqx
```

Enter `quit` to terminate your ftp connection.

Harvest C will **not** work with the MPW 3.1 headers and libraries.

Sample Programs

This release of Harvest C shareware comes with 2 sample programs. Sample1 is truly trivial. It simply does some math calculations and beeps three times.

Sample2 is still not very complex, but it does create and manage a menu, allowing the user to quit or use desk accessories. Sample2 has two C source files, called Sample2a.c and Sample2b.c. Sample2a.c is very large, and Harvest C may require a large amount of memory to compile it. See the problems section below for comments about hogging memory.

"I'm lost - what does all this mean ?"

This informational file is not intended to be full documentation. I have made countless assumptions about the experience of the reader. If you are totally lost and do not understand the terms of this document, be advised that I am preparing better documentation. In addition, feel free to contact me by electronic mail if you have any questions. I will accept questions from anyone, but I will respond to my registered users first.

Keep in mind that even the finest documentation for this product would not be a tutorial for the C language. There are some good books on C available. Better documentation would include a list of them here.

If you are a relatively experienced C user and want to know specific details about how Harvest C handles a certain situation, the only answer I have for you now is : "It works just like MPW C". This is undoubtedly not always true, but it is the ideal I want to strive for. I have written Harvest C to be compatible with MPW in almost every way. This does NOT make Harvest C just as good as MPW C, just compatible. More specifically, MPW C is currently faster and more reliable, produces better code, handles the `comp` data type, does precompilation of headers, and supports the "32 bit everything" architecture. My goal is to make Harvest C a highly competent system in its own right, but I do not plan on attempting to eliminate any of existing products, which are of excellent quality.

CURRENT PROBLEMS

As no software is bug free, and compilers are particularly prone to problems, I will not be so foolish as to claim that Harvest C is perfect. I **do** want to hear of any problems you may have. Please report difficulties to me so I can make bug fixes. I plan to make subsequent releases to fix problems. The following items are already known, and I plan on correcting them :

1. The compiler is slow.

2. Documentation is poor, almost nonexistent.

3. Handling of errors is poor.

4. The editor is very poor. I have no plans of improving this, but I do plan to integrate Harvest C with another editor, using System 7 IAC.

5. Needs System 7 friendliness (more Balloon help and Apple Events). Compatibility with virtual memory or 32 bit mode is unknown.

6. Harvest C currently does not support the MPW `comp` data type

7. The reliability of SANE floating point math is not fully known. It has not been extensively tested. Floating point math generated for the 68881 (not available in the shareware version) has been tested more extensively.

8. Harvest C does not support printing text files from within the compiler application.

9. The Harvest C Linker is not as smart as the MPW linker. It produces bloated applications.

10. Harvest C is a memory hog. The efficiency of its memory usage will be cleaned up as the program matures. This should speed up compilation as well.

11. Harvest C sometimes produces code which is truly inane. I know about it, and I am working on it. I certainly welcome technical suggestions from any users (many of which I know are far wiser than I). I don't have too much difficulty looking at the Harvest C output and seeing obvious problems, but I still welcome your ideas.

12. The Harvest C user interface is rather poor. This has been pointed out to me by a number of testers, and I have been convinced to pay more attention to improving it. The most glaring problem is the hierarchial warnings menu, which is really an abuse of the menu manager.