# Compiling Version 8 of Icon for MS-DOS

Ralph E. Griswold

Department of Computer Science, The University of Arizona

## 1. Background

The implementation of the Icon programming language is large and complex [1]. It is, however, written almost entirely in C, and it is designed to be portable to a wide range of computers and operating systems. This document concerns the compilation of Version 8 of Icon for MS-DOS.

Version 8 of Icon runs on computers with 8086/88/186/286/386-family processors. IBM hardware compatibility is not required. Either MS-DOS or PC-DOS, Version 2.0 or higher, is needed. Specific C compilers may impose more stringent requirements. Version 8 of Icon is a large-memory-model program that requires at least 512KB of RAM to perform satisfactorily.

As of the date of this document, Version 8 of Icon for MS-DOS has been successfully compiled a large-memory model program with the following C compilers:

Microsoft C 5.10 (MS-DOS and OS/2)
Turbo C 2.0
Lattice C 6.01
Mark Williams Let's C 4.0.12

Icon may compile under some earlier versions of these compilers, but details are not known.

Icon built under Microsoft C 5.10 supports all features of Icon and has no known bugs. Icon compiler under Turbo C 2.0 support all features of Icon, but co-expressions do not work properly. Icon compiled under Lattice C 6.01 supports all features of Icon except large-integer arithmetic (the module is too large to compile), and co-expressions are not implemented. Icon compiled under Mark Williams Let's C 4.0.12 runs some programs, but there are numerous problems. More detailed information is provided in status files that accompany the distribution.

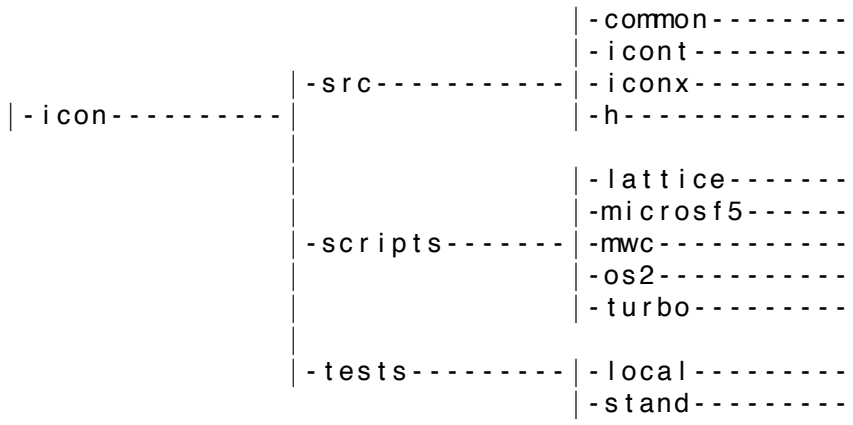
The use of another C compiler will certainly involve some work, since there is some code that is conditional on the characteristics of specific C compilers. Compiling Icon is beyond the capability of many C compilers — be forewarned. For some C compilers, it also may be necessary to make some compromises in the facilities that Icon supports. If you plan to try compiling Icon on a compiler other than one of those listed above, request technical report TR 88-9 from the Icon Project.

The Microsoft macro assembler (MASM) is needed for the optional assembly-language component of Icon — co-expressions. This feature can be omitted if you do not have MASM.

## 2. Organization of the Implementation

The source code for Icon is organized in a hierarchy. To facilitate file transfer, files for various directories are packaged in *arc* format. Instructions for unloading the files are provided on the distribution diskettes.

If the Icon hierarchy is rooted in \icon, the directories following unloading are:




The distribution diskettes also contain documentation and some tools that may be useful in building and testing Icon. See README on the distribution diskettes.

### 2.1 Source Files

The four source-code sub-directories under src contain the following components of Icon:

| | |
|---|---|
| common | files common to different components of Icon. |
| h | header files used by files in the other directories. |
| icont | source code for the translator and linker that converts a source-language program into an *icode* that is executed by iconx. |
| iconx | source code for the executor for icode, including a run-time system that supports the operations of the Icon language. |

### 2.2 Configuration Directories

In order to simplify the process of compiling Icon under different C compilers, files that are compiler-specific, including batch and linker files, are provided in subdirectories of the scripts directory. There are presently five of these *configuration* directories:

| | |
|---|---|
| lattice | Lattice C 6.01 |
| microsf5 | Microsoft C 5.10 for MS–DOS |
| mwc | Let's C 4.0.12 |
| os2 | Microsoft C 5.10 for OS/2 |
| turbo | Turbo C 2.0 |

The use of these configuration directories is described in Section 3.

## 3. An Overview of the Compilation Process

Before starting to compile Icon, be sure your C compiler is properly installed and that any paths that it needs are properly set.

### Configuration

The first step in the compilation process is to configure the source code. If you are using one of the C compilers described above, there is a .bat file in the top level of the icon hierarchy (e.g. \icon) whose name corresponds to the configuration directory. Executing the .bat file performs the configuration. For example, if you want to configure Version 8 of Icon to compile under Microsoft C 5.10, just type

```
microsf5
```

These batch files first erase files that may be left over from a previous configuration ("File not found" is normal at

this point), and then they copy in compiler-specific scripts and source files.

**Assembly-Language Matters**

The assembly-language component of Icon requires macro files that are provided with the C compiler (not as part of the Icon source). For the Microsoft C compiler, the files cmacros.inc and version.inc are needed. These files are included in the Microsoft C distributions, but to the best of our knowledge, their location or purpose is not documented. You may have to search for them on the Microsoft distribution diskettes. cmacros.inc should be installed in \mac. version.inc should be installed at \mac\l (ell).

If you do not have MASM or do not want to include the assembly-language component of Icon for some other reason, add the following line to src\h\define.h:

```
#define NoCoexpr                    /* disable co-expressions */
```

See also the remark at the end of the following section about compilation without the assembly-language component of Icon.

**Extra Functions for MS-DOS**

There are a few functions specially designed for using Icon under MS-DOS that are not part of Icon's standard function repertoire. The functions are described in [2] (ipd132.doc on the distribution diskettes). These functions normally are included in the compilation process. If you wish to eliminate them (which decreases the size of iconx by a few thousand bytes), remove

```
#define DosFncs
```

from src\h\define.h.

**Large-Integer Arithmetic**

Icon has facilities for large-integer arithmetic, but these facilities are disabled by default in MS-DOS Icon because they increase the size of iconx substantially (20-30KB). If you have enough RAM and wish to enable large-integer arithmetic, remove the following line from src\h\define.h:

```
#define NoLargeInts
```

**Compilation**

First go to the directory src\common and compile the source files there.

Then go to src\icont and src\iconx and compile and link the source files there to produce the executable files icont.exe and iconx.exe, respectively.

Finally, copy the executable files to their desired location.

A public-domain version of a UNIX-style make utility is provided on the distribution diskettes. Makefiles for each subdirectory are copied into place during configuration. This make utility is different from (and more powerful) than the make utilities provided by present MS-DOS C compilers. If you do not want to use it, there is a build.bat file in each source subdirectory that compiles and links all files without using make.

If you disabled the assembly-language component of Icon, you will need to change the makefile or build.bat file in src\iconx to remove the assembly steps.

### 3.1 Testing

A suite of test programs is provided with the distribution. The tests range from a variety of simple programs to batteries of expressions. The test programs themselves are in the directory tests. The expected output of the test programs is in tests\stand; tests\local is provided for local output.

The directory tests contains several files of the form *name*.lst, which consist of the names of test programs. Testing can be done with these files and the program runtests provided with the distribution. The form is:

runtests *name*.lst > *name*.res

where *name* is one of the files mentioned above. As a result, *name*.res contains a list showing differences.

*Note:* Local output may differ in some cases from the output in stand. This may be due to compiler or system differences. Also, in a few cases, recent changes to Icon may produce output somewhat different from that in stand. See README on the distribution diskettes for the latest information.

## 4. The Implementation Book

If you are interested in the larger view of the implementation of Icon, or if you are interested in modifying or extending Icon, you may want to acquire the book *The Implementation of the Icon Programming Language*. This book concentrates on the run-time system and covers data structures, the virtual machine, the interpreter, the implementation of generators, and storage management. It also contains material specifically related to making modifications to the source code.

The publication information is: *The Implementation of the Icon Programming Language*, by Griswold and Griswold, Princeton University Press, ISBN 0-691-08431-9, hardbound, 336 pages, $44.50. The book may be ordered from the Icon Project, a local bookstore, or directly from the publisher:

Princeton University Press
3175 Princeton Pike
Lawrenceville, NJ 08648

(609) 896-1344

The implementation book corresponds to Version 6.2 of the Icon source code. There have been several changes in the source code between Version 6.2 and the present version. A supplement describing these changes is available free of charge from the Icon Project [3]. Ask for IPD112.

## 5. Trouble Reports and Feedback

If you run into problems, contact the Icon Project:

Icon Project
Department of Computer Science
Gould-Simpson Building
The University of Arizona
Tucson, AZ 85721
U.S.A.

(602) 621-4049

icon-project@cs.arizona.edu (Internet)
… {uunet, allegra, noao}!arizona!icon-project (uucp)

We cannot promise to solve your problems, but we will try. We also may be able to place you in contact with other persons who are compiling Icon and who may have similar problems.

Please also let us know of any suggestions for improvements to the compilation process or its documentation.

**Acknowledgements**

Many persons have been involved in the implementation of Icon. Cheyenne Wills did most of the work to adapt Icon for use under MS-DOS. He also provided most of the tools that are included in the distribution.

**References**


1. R. E. Griswold and M. T. Griswold, *The Implementation of the Icon Programming Language*, Princeton University Press, 1986.
2. R. E. Griswold, *Version 8 of Icon for MS-DOS*, The Univ. of Arizona Icon Project Document IPD132, 1990.
3. R. E. Griswold, *Supplementary Information for the Implementation of Version 8 of Icon*, The Univ. of Arizona Icon Project Document IPD112, 1990.