Kermit 95 Compiler Options

Edward Berner

August 31, 2011

Compiler and linker options used in the `ckoker.mak` makefile included in the Kermit 95 source code released by Columbia University in July 2011. (`http://www.columbia.edu/kermit/k95sourcecode.html`)

Kermit 95 runs on IBM's OS/2 and various versions of Microsoft Windows. Microsoft Visual C++ was used to build Kermit 95 for Windows platforms, and IBM VisualAge C++ was used to build Kermit 95 for OS/2 platforms.

1 Microsoft

According to Frank da Cruz, "K95 was last built with Microsoft Visual Studio 6.0" (`http://www.columbia.edu/kermit/k95sourcecode.html`).

The information below is taken from "Compiler Reference, Visual Studio 6.0" on Microsoft's MSDN website (`http://msdn.microsoft.com/en-us/library/aa236704`). Some of the descriptions are copied verbatim from the website.

1.1 Compiler Options

Note: the various `/O` options are optimization related and are only supported in the Professional and Enterprise editions of Visual C++.

`-c` Compile without linking.

`/F65536` "`/F`*number*" set the program's stack size to the given number of bytes (the linker might round up to the next multiple of four bytes). The default stack size is 1 MB.

`-Fe` "`/Fe`*filename*" rename the executable file.

`/Fm` "`/Fm`*filename*" create a map file.

`/G4` Optimize code to favor the 486 processor.

`/G5` Optimize code to favor the Pentium processor.

`/GA` Optimize code for Windows application. (Speeds up access to data declared with the `__declspec(thread)` attribute.)

`/GF` Pool strings and place them in read-only memory.

`/Gn-` This is not listed as a Visual C++ compiler option, but the makefile seems to use it for both the IBM and Microsoft compilers. Perhaps it is a makefile bug, or perhaps I've overlooked something.

`/GX-` Enable synchronous exception handling.

`/J` Change the default `char` type from signed to unsigned.

`/MD` Create a multithreaded DLL using `MSVCRT.LIB`.

`/nologo` Suppress display of some startup and informational messages.

`/Ob1` Consider only functions marked with `inline` or `__inline` for inline expansion. It is up to the compiler to decide whether to actually inline such functions. The default option is `/Ob0` which disables inlining. The option `/Ob2` is like `/Ob1` but allows the compiler to also select other functions for inlining.

`/Og` Enable several types of local and global optimization. See the MSDN website for details.

`/Oi` Generate intrinsic functions. See the MSDN website for details.

`/Ot` Optimize for speed as opposed to size.

`/Ox` Full optimization. Equivalent to "`/Ob1gity /Gs`". (`/Oy` omits frame pointers and `/Gs` has to do with controlling stack probes.)

`/W2` Set the warning level. Higher numbers display more warnings.

`/Ze` Enable C language extensions. The opposite is `/Za` which forces strict ANSI C.

`/Zp4` Pack structure members on 4 byte boundaries.

1.2 Linker Options

`/align:0x1000` Specifies the alignment of each section of the program within its linear address space.

`/DEBUG:full` Create debugging information.

`/FIXED:NO` Controls whether a relocation section is included in the program. `/FIXED:YES` does not create a relocation section. `/FIXED:NO` creates a relocation section and is likely necessary for some older Win32 operating systems. See the MSDN website for more information.

`/MAP` Causes the linker to create a mapfile.

`/nologo` Suppress display of some startup and informational messages.

`/OPT:REF` Eliminate functions and data that is never referenced.

`/PROFILE` Enable profiling. (Only available in the Professional and Enterprise editions of Visual C++.)

`/SUBSYSTEM:console` Indicates that the program being linked is a Win32 character-mode application.

`/SUBSYSTEM:windows` Indicates that the program being linked is a normal Windows application. (That oversimplifies bit. In addition to `console` and `windows`, there are also (at least) `native`, `posix`, and `windowsce` options.)

`/WARN:3` Set the warning level. Higher numbers display more warnings.

2 IBM

I don't know what version of VisualAge C++ was used to build Kermit 95, but VisualAge C++ Professional 4.0 was contemporary with Microsoft Visual C++ 6.0, and was the last version of VisualAge C++ released for OS/2.

(IBM withdrew VisualAge C++ Professional 4.0 for OS/2 from marketing on April 27, 2001 and ended support on September 28, 2001. The actual withdrawal announcement is available at the following URL: `http://www-01.ibm.com/common/ssi/rep_ca/3/897/ENUS901-013/ENUS901-013.PDF`)

Regarding the relative dates of the compilers:

> The article "Visual Studio opens window onto the Web" (*Infoworld*, August 17, 1998, font page) says that Visual Studio 6.0 will be delivered in September 1998.
>
> The February 1, 1999 issue of *Infoworld* contains a review of VisualAge C++ Professional 4.0 ("IBM VisualAge for C++ boosts tools, speeds compiler", pages 81 – 82).

I was not able to find online versions of the manuals for IBM VisualAge C++ 4.0, so the following information is from the *VisualAge C++ for OS/2 User's Guide* available in the *VisualAge C++ for OS/2, V3.0 Bookshelf* (publication number GC09-2215-00) on IBM's website (`http://publibfp.dhe.ibm.com/cgi-bin/bookmgr/Shelves/cppvac00`). Some of the descriptions are copied verbatim from the website.

2.1 Compiler Options

`/B` "`/B"`*options*`"`" passes *options* to the linker.

`-c` Compile without linking.

`-Fe` "`/Fe`*filename*" sets the name of the output executable file.

`-Fi+` Enable creation of precompiled header files.

`-G5` Optimize for Pentium processors.

`-Gd` Dynamically link to the runtime library.

`/Gd-` Statically link to the runtime library.

`/Ge-` Build a `.DLL` file.

`-Gh` Enable code to be run by Performance Analyzer. See IBM's documentation for more details.

`-Gi+` Use fast integer execution.

`/Gl+` Remove unreferenced functions. (Causes the `/FUNCTIONOPT` option to be passed to the linker.)

`-Gm` Link with the multithreaded runtime library.

`/Gn-` Do not hide default library information from the linker.

`-Gs` Remove stack probes from the generated code.

`-Gt` Enable variables to be passed to 16-bit functions. See the IBM documentation for details.

`-Gt-` Do not enable variables to be passed to 16-bit functions.

`-J` Set the default `char` type to unsigned.

`-O` Optimize code.

`-Oi25` "`/Oi`*value*" causes all functions marked with `_Inline` or `inline` and all functions shorter than *value* "abstract code units" to be considered for inlining. Whether they are actually inlined is up to the compiler.

`-q` Suppress display of the compiler logo.

`-Si+` Use precompiled header files if they exist and are current.

`-Sm` Ignore unsupported 16-bit keywords such as `near` and `far`.

`-Sp1` Align structure and union members on 1 byte boundaries.

`-Ti` Generate debugger information.

`-Ti+` Same as `-Ti`.

`-Tm+` Use debug versions of memory management functions.

`-Tx+` Provide a complete machine state dump when an exception occurs.

`/Wcmp` Generate warnings for possible redundancies in unsigned comparisons.

`/Wcnd` Generate warnings for possible redundancies or problems in conditional expressions.

`/Wcns` Generate warnings for operations involving constants.

`/Wdcl` Not listed in the manual I was consulting. Another resource on the web describes it as "Check for declaration consistency" (`http://svn.netlabs.org/repos/fat32/trunk/src/icc.opt`).

`/Weff` Generate warnings for statements with no effect.

`/Wenu` Generate warnings for consistency of enum variables.

`/Wext` Generate warnings for unused external definitions.

`/Wgnr` Generate warnings for generation of temporary variables.

`/Word` Generate warnings for unspecified order of execution.

`/Wpar` Generate warnings for unused parameters.

`/Wppc` Generate warnings for possible problems with using the preprocessor.

`/Wpro` Generate warnings for missing function prototypes.

`/Wrea` Generate warnings for code that cannot be reached.

`/Wret` Generate warnings for consistency of return statements.

`/Wtrd` Generate warnings for possible truncation or loss of data or precision.

`/Wund` Generate warnings for casting of pointers to or from an undefined class.

`/Wuni` Generate warnings for uninitialized variables.

`/Wuse` Generate warnings for unused auto and static variables.

2.2 Linker Options

`/align:16` Set the alignment factor of an `.EXE` or `.DLL` file. Pages within the file are aligned on a byte boundary that is a multiple of the given number.

`/base:0x10000` Set the preferred load address of a `.DLL` file or the default base address of an `.EXE` file.

`/dbgpack` Eliminate redundant debug type information.

`/debug` Include debug information.

`/nologo` Suppress display of some startup and informational messages.

`/noi` Cause the linker to be case sensitive. (Short for `/NOIGNORECASE`.)