Programming Environments - Intel C++ Compiler

Miguel Fiandor Gutiérrez[1] Manuel Macías Córdoba[2]

[1] *Department of Computing, Universidad Politécnica de Madrid*
Montegancedo sn, 28660 Boadilla del Monte, Spain
mfgutierrez@gmail.com
[2] manuelmaciascordoba@gmail.com

1. Introduction

When somebody is interested in taking part at the software development world at any programming language, first, he should choose a programming environment that makes possible to create and run programs. There are two main types of programming environments: a graphical user interface (IDE) that also might includes debugger, interface builder, and project management tools. Or a command-line environment - a collection of commands typed in to edit files which you can compile source code and run programs as well.

Command-line environments are recommended for beginners as they help to understand the fundamentals of programming. Because we could say that such task in IDEs usually is transparently to users. Deal with IDEs should come at the time the user needs to manage big projects, better debugging tools, and so on.

Since exists several programming languages imagine how many different programming environments we can find. They can be shorted by the system the program will run on, for example, for FORTRAN language we found 8 recommended environments for Sun, DEC and IBM Systems[2].

We mention some of the most important or popular programming environments:

- For C++: UNIX programming environment, Visual C++, Dev-C++ and Borland C++ Builder.
- For C: UNIX programming environment, Dev-C++ and Borland C++ Builder.
- For Java: J2SE, J2EE, J2ME and Eclipse.
- For XML: Dreamweaver, Oxygen and UltraEdit32.

2. Intel C++ Compiler and its Programming Environments

Intel C++ Compiler describes a group of C/C++ and Fortram compilers from Intel. Intel supports compilation for its IA-32, Intel 64, Itanium 2, and XScale processors. The Intel C++ Compiler for x86 and Intel 64 feature an

automatic vectorizer that can generate SSE, SSE2, and SSE3 SIMD instructions, the embedded variant for Intel Wireless MMX and MMX 2. Since its introduction, the Intel C++ Compiler for IA-32 has greatly increased adoption of SSE2 in Windows application development.

The intel c++ compiler can be run in windows, linux, and MAC operating systems. As it has not a programming environment of its own, it can use different ones depending on the system. In order to be run in linux it requires a Linux system with glibc 2.2.2 and kernel 2.4. The compiler has been validated with Red Hat Linux 7.1.

The compiler can be invoked from the command line of the shell or using a make file. The command to be use in the command line is as follows:

```
prompt> icc [options] file1 [file2...] [linker options]
```

The options and linker options can be found in the user manual[3] provided by intel for further detail.

The intel c++ compiler 9.1 for Windows provides Visual Studio 2005 support for the first time, allowing the development of applications for IA-32 and processors that support Intel® EM64T, as well as Itanium® 2 processors using visual studio programming environment. The intel c++ compiler for windows provides optimization support for the intel multi-core processors. Software compiled using the Intel C++ Compilers for Windows benefits from advanced optimization features, a few of which are explained briefly here:

- Multithreaded Application Support, including OpenMP and auto-parallelization for simple and efficient software threading.
- Interprocedural Optimization (IPO) dramatically improves performance of small- or medium-sized functions that are used frequently, especially programs that contain calls within loops.
- Profile-guided Optimization (PGO) improves application performance by reducing instruction-cache thrashing, reorganizing code layout, shrinking code size, and reducing branch mispredictions.
- Automatic Vectorizer parallelizes code and aligns data, including loop peeling to generate aligned loads and loop unrolling to match the prefetch of a full cache line.
- High Level Optimization (HLO) delivers aggressive optimization with loop transformation and pre-fetching.
- Optimized Code Debugging with the Intel® Debugger improves the efficiency of the debugging process on code that has been optimized for Intel® architecture.

3. Intel C++ Compilier vs GCC

We have found a good and interesting test[4] between both compilers, firstly we show the results and they will be study below.

Good.
- Performance gains of up to approximately 30
- Optional Fortran compiler vs. g77, which translates into C and compiles with GCC.
- GCC compatibility mode.
- 64-bit support (Intel Itanium only).

Bad.
- Limited support for glibc (7.0 supports 2.2.4 or 2.2.5 only).
- Requires RPM for install.
- Debugger does not include a GUI, command line only.
- Official support for Intel Pentium/Xeon or Intel Itanium processors only (no support for cross platform development).
- GCC compatibility limited with open-source code tested.

The performance talked in first point is between Intel C++ Compiler 7.1 and GCC 3.2, but latest versions are 9.1 and 4.1.2 respectively. Obviously, later versions add better improvements, a comparison between Intel 7.1 and gcc 3.3 are quite similar over Intel processors. So as we do not have a study between these two, we cannot decide which is better. Nevertheless, exist a common result in these tests - programs ran over non Intel processor (Athlon ones for example) have not same different improvement between the compilers as when is run over a Intel one.

Intel C++ Compiler actually does not support all extensions that GCC implements, but it accepts most of them, that is what is said 'compatibility'.

The requirements have considerably changed, now they ask for complete system versions as latest versions of Red Hat, SUSE, Fedora, Windows 2000, XP or Server 2003, but not about which libraries should be previously installed. For more detail [5] .

References

[1] Programming Environment Definition, `http://www-plan.cs.colorado.edu/doerr/teaching/csci3308/javaforc/math.hws.edu/javanotes/progenv.html`
[2] Fortran recomended programming environments, `http://www.llnl.gov/icc/lc/asci/fpe/fpe.recommend.html#DEC`
[3] Intel C++ Compiler User's Guide, `http://www.osc.edu/hpc/manuals/ia64/docs2/c_ug/index.htm`
[4] Intel C++ compiler and comparative with GCC, `http://www.linuxjournal.com/article/6766`
[5] Intel C++ Compilers, `http://www.intel.com/cd/software/products/asmo-na/eng/compilers/284132.htm`
[6] Intel C++ Compiler tutorial for enhancing performance, `http://www.osc.edu/hpc/manuals/ia64/docs2/training/optimize/index.htm`