Intel® Integrated Native Developer Experience 2015 Build Edition for OS X*
Installation Guide and Release Notes

22 January 2015

Table of Contents

1 Introduction

This document describes how to install the product, provides a summary of new and changed product features and includes notes about features and problems not described in the product documentation. For the most current update to these release notes, see the release notes posted at the Intel® Software Development Products Registration Center where you downloaded this product.

Due to the nature of this comprehensive integrated software development tools solution, different Intel® Integrated Native Developer Experience Build Edition for OS X* components may be covered by different licenses. Please see the licenses included in the distribution as well as the Disclaimer and Legal Information section of these release notes for details.

1.1 Change History

This section highlights important from the previous product version and changes in product updates. For information on what is new in each component, please read the individual component release notes.

1.1.1 Changes in Update 1

- OS X* 10.10 now supported
- Xcode* 6.0 and 6.1 now supported
- Support for Intel® Advanced Vector Extensions 512 instructions for IA-32 and Intel® 64 architectures in 15.0.1
- New string conversion functions added to libistrconv in 15.0.2
- -xCOMMON-AVX512 and -axCOMMON-AVX512 options added in 15.0.2
- -auto-p32 supported on OS X* and both –auto-p32 and –auto-ilp32 require –no-pie

- Intel® C++ Compiler 15.0.2
- Intel® Integrated Performance Primitives 8.2 Update 1
- Intel® Threading Building Blocks 4.3 Update 3

1.2 Product Contents

Intel® Integrated Native Developer Experience 2015 Update 1 Build Edition for OS X initial release* includes the following components:

- Intel® C++ Compiler 15.0.2 for building applications that run on Intel-based Mac systems running the OS X* operating system
- Intel® Integrated Performance Primitives 8.2 Update 1
- Intel® Threading Building Blocks 4.3 Update 3
- Integration into the Xcode* development environment
- On-disk documentation

1.3 System Requirements

- A 64-bit Intel®-based Apple* Mac* system host (development for 32-bit is still supported)
- 2GB RAM minimum, 4GB RAM recommended
- 3GB free disk space
- One of the following combinations of OS X*, Xcode* and the Xcode SDK:
  - OS X 10.10 and Xcode* 6.0 or 6.1
  - OS X 10.9 and Xcode* 5.0 or 5.1
- If doing command line development, the Command Line Tools component of Xcode* is required

Note: Advanced optimization options or very large programs may require additional resources such as memory or disk space.

1.4 Documentation

Product documentation can be found in the `Documentation` folder as shown under Installation Folders.

Optimization Notice

Intel's compilers may or may not optimize to the same degree for non-Intel microprocessors for optimizations that are not unique to Intel microprocessors. These optimizations include SSE2, SSE3, and SSSE3 instruction sets and other optimizations. Intel does not guarantee the availability, functionality, or effectiveness of any optimization on microprocessors not manufactured by Intel. Microprocessor-dependent optimizations in this product are intended for use with Intel microprocessors. Certain optimizations not specific to Intel microarchitecture are reserved for Intel microprocessors. Please refer to the applicable product User and Reference Guides for more information regarding the specific instruction sets covered by this notice.

Notice revision #20110804

1.5 Samples

Samples for each product component can be found in the `Samples` folder as shown under Installation Folders.

1.6 Technical Support

If you did not register your compiler during installation, please do so at the Intel® Software Development Products Registration Center at http://registrationcenter.intel.com. Registration entitles you to free technical support, product updates and upgrades for the duration of the support term.

For information about how to find Technical Support, Product Updates, User Forums, FAQs, tips and tricks, and other support information, please visit:
http://www.intel.com/software/products/support/

Note: If your distributor provides technical support for this product, please contact them for support rather than Intel.

2 Installation

The installation of the product requires a valid license file or serial number. If you are evaluating the product, you can also choose the "Evaluate this product (no serial number required)" option during installation.

If you will be using Xcode*, please make sure that a supported version of Xcode is installed. If you install a new version of Xcode in the future, you must reinstall the Intel C++ Compiler afterwards.

The Command Line Tools component, required for command-line development, is not installed by default. It can be installed using the Components tab of the Downloads preferences panel.

You will need to have administrative or “sudo” privileges to install, change or uninstall the product.

To begin installation, double-click the downloaded .dmg file.

Follow the prompts to complete installation.

Note that there are several different downloadable files available, each providing different combinations of components. Please read the download web page carefully to determine which file is appropriate for you.

You do not need to uninstall previous versions or updates before installing a newer version – the new version will coexist with the older versions.

2.1 Intel® Software Manager

The installation now provides an Intel® Software Manager to provide a simplified delivery mechanism for product updates and provide current license status and news on all installed Intel® software products.

You can also volunteer to provide Intel anonymous usage information about these products to help guide future product design. This option, the Intel® Software Improvement Program, is not enabled by default – you can opt-in during installation or at a later time, and may opt-out at any time. For more information please see http://intel.ly/SoftwareImprovementProgram.

2.2 Using a License Server

If you have purchased a "floating" license, see http://intel.ly/pjGfwC for information on how to install using a license file or license server. This article also provides a source for the Intel® License Server that can be installed on any of a wide variety of systems.

2.3 Silent Install

For information on automated or “silent” install capability, please see http://intel.ly/1gcW0Bl

2.4 Installation Folders

The compiler installs, by default, under `/opt/intel` – this is referenced as `<install-dir>` in the remainder of this document. You are able to specify a different location.

The `inde_2015.<n>.<pkg>` directory contains the following directories that reference a specific update of the Intel® Integrated Native Developer Experience 2015 Build Edition :

- `bin` – all executables
- `pkg_bin` – symbolic link to `bin` directory
- `compiler` – shared libraries and header files
- `Documentation` – documentation files
- `man` – symbolic link to the `man` directory
- `ipp` – Intel® Integrated Performance Primitives libraries and header files
- `tbb` – Intel® Threading Building Blocks libraries and header files

- `Samples` – Product samples and tutorial files

2.5 Relocating Product After Install

The Xcode integration is relocatable simply by dragging and dropping the Xcode directory tree to another location.

2.6 Removal/Uninstall

It is not possible to remove the compiler while leaving any of the performance library components installed.

1. Open Terminal and set default (`cd`) to any folder outside `<install-dir>`
2. Type the command:
   `<install-dir>/inde_2015.<n>.<pkg>/uninstall_ccinde.sh`
3. Follow the prompts

If you are not currently logged in as `root` you will be asked for the `root` password.

3 Intel® C++ Compiler

This section summarizes changes, new features and late-breaking news about the Intel C++ Compiler.

3.1 New and Changed Features

The following features are new or significantly enhanced in this version. For more information on these features, please refer to the documentation.

- New string conversion functions added to libistrconv in 15.0.2
- -xCOMMON-AVX512 and -axCOMMON-AVX512 options added in 15.0.2
- In Intel® C++ Compiler 15.0.2, -auto-p32 supported on OS X* and both –auto-p32 and –auto-ilp32 require –no-pie
- Support for Intel® Advanced Vector Extensions 512 instructions for IA-32 and Intel® 64 architectures in 15.0.1

3.1.1 New string conversion functions added to libistrconv in 15.0.2

The following functions have been added to Intel's Numeric String Conversion Library (libistrconv) for the Intel® C++ Compiler 15.0.2. For more details, see the User's Guide.

- `__IML_f_to_str`
- `__IML_d_to_str`
- `__IML_f_to_str_f`
- `__IML_d_to_str_f`
- `__IML_f_to_str_e`
- `__IML_d_to_str_e`
- `__IML_i_to_str`
- `__IML_u_to_str`

- `__IML_ll_to_str`
- `__IML_ull_to_str`

3.1.2 Support for Intel® Advanced Vector Extensions 512 (Intel® AVX-512) instructions for IA-32 and Intel® 64 architectures in 15.0.1

The Intel® Compiler 15.0.1 now supports Intel® AVX-512 instructions for processors based on IA-32 and Intel® 64 architectures that support that instruction set. The instructions are supported via inline assembly, intrinsics (using the zmmintrin.h include file), and/or the /Q[a]xCORE-AVX512 (Windows*) or –[a]xCORE-AVX512 (Linux*/OS X*) compiler options. This is in addition to the current support for Intel® AVX-512 instructions for Intel® Many Integrated Core Architecture.

3.2 New and Changed Compiler Options

For details on these and all compiler options, see the Compiler Options section of the on-disk documentation.

3.2.1 New and Changed in Intel® C++ Compiler 15.0

- -xCOMMON-AVX512
- -axCOMMON-AVX512
- -auto-p32
- -xCORE-AVX512
- -axCORE-AVX512

For a list of deprecated compiler options, see the Compiler Options section of the documentation.

3.2.2 -xCOMMON-AVX512 and -axCOMMON-AVX512 options added in 15.0.2

The COMMON-AVX512 selection has been added for the -x and -ax compiler options. This allows generation of the subset of Intel® Advanced Vector Extensions 512 (Intel® AVX-512) that are supported on both Intel® Many Integrated Core Architecture and on Intel® Core™ Microarchitecture. This includes Intel AVX-512 Foundation instructions and Intel AVX-512 Conflict Detection instructions.

3.2.3 In Intel® C++ Compiler 15.0.2, -auto-p32 supported on OS X* and both –auto-p32 and –auto-ilp32 require –no-pie

Starting in the Intel® C++ Compiler 15.0.2, -auto-p32 is supported on OS X* as well as Linux*. Also, on OS X*, both the -auto-p32 and -auto-ilp32 options require Position Independent Executable code to be disabled. This is done with the -no-pie option. If -no-pie is not specified, -auto-p32 and -auto-ilp32 will be ignored.

3.3 Notes

3.3.1 Environment Setup Script

The `compilervars.sh` script is used to establish the compiler environment.

The command takes the form:

`source <install-dir>/inde.<n>.<pkg>/compilervars.sh argument`

Where *argument* is either `ia32` or `intel64` as appropriate for the architecture you are building for. Establishing the compiler environment also establishes the environment for the Intel® Performance Libraries.

3.4 Known Limitations

3.4.1 Incompatible with the default libc++ library

Some applications are incompatible with libc++, which is currently enabled as default C++ library in the Intel® C++ Compiler 15.0. For example, the 435.gromacs and 447.dealII from SPEC CPU2006 suite.

Please use `-stdlib=libstdc++` to compile such applications.

3.4.2 "Build" or "Run" in Xcode* 5 do not accurately detect and rebuild modified dependencies

When the Intel® C++ Compiler is selected for a project in Xcode* 5, there is an issue where if a file that other files/projects depend on is modified, for example a header file, then the dependent files or libraries will not be automatically rebuilt with the "Build" or "Run" commands. Instead, a "Clean" needs to be done first to do a full rebuild. This issue is being investigated.

3.4.3 Creating new project in Xcode* causes hardcoding of –stdlib=libc++

A new project created in Xcode* causes the hardcoding of a setting for –stdlib=libc++ even for projects that have the Intel® C++ Compiler toolset added. So setting the Intel® C++ Compiler field for the C++ Standard Library setting to libstdc++ is ineffective because libc++ overrides the setting. To change this, do the following:

1. Select the project row in the navigator area at the left of the workspace window
2. In the project editor that appears, select the row that represents the project level of build settings
3. You should see the `C++ Standard Library` setting in bold, indicating that it has a custom value in this project
4. Select that row and press the Delete key to remove the customized value
5. The `C++ Standard Library` build property should now have the value `Compiler Default`

Note that you may have to follow the above steps before adding the Intel® C++ Compiler toolset to your project build rules.

4 Intel® Integrated Performance Primitives

This section summarizes changes, new features and late-breaking news about this version of Intel® Integrated Performance Primitives (Intel® IPP).

The latest information on Intel® IPP 8.2 can be found in the product release notes under `<install dir>/inde_2015.x.xxx/Documentation/<locale>/ipp/ReleaseNotes.htm`.

For detailed information about Intel IPP see the following links:

- **New features**: see the information below and visit the main Intel IPP product page on the Intel web site at: http://intel.ly/OG5IF7; and the Intel IPP Release Notes at http://intel.ly/1uj984p.
- **Documentation, help, and samples**: see the documentation links on the IPP product page at: http://intel.ly/OG5IF7.

4.1 Intel® IPP Cryptography Libraries are Available as a Separate Download

The Intel® IPP cryptography libraries are available as a separate download. For download and installation instructions, please read http://intel.ly/ndrGnR

5 Intel® Threading Building Blocks

For information on changes to Intel® Threading Building Blocks (Intel® TBB), please read the file `CHANGES` in the Intel® TBB documentation directory found in `<installdir>/inde_2015.x.xxx/Documentation/<locale>/tbb`.

6 Disclaimer and Legal Information

The products described in this document may contain design defects or errors known as errata which may cause the product to deviate from published specifications. Current characterized errata are available on request.

Contact your local Intel sales office or your distributor to obtain the latest specifications and before placing your product order.

Copies of documents which have an order number and are referenced in this document, or other Intel literature, may be obtained by calling 1-800-548-4725, or go to: http://www.intel.com/design/literature.htm

Intel processor numbers are not a measure of performance. Processor numbers differentiate features within each processor family, not across different processor families. Go to:

http://www.intel.com/products/processor%5Fnumber/

The Intel® C++ Compiler, Intel® Integrated Performance Primitives, and Intel® Threading Building Blocks are provided under Intel's End User License Agreement (EULA).

Please consult the licenses included in the distribution for details.

Celeron, Centrino, Intel, Intel logo, Intel386, Intel486, Atom, Core, Itanium, MMX, Pentium, VTune, Cilk, and Xeon are trademarks of Intel Corporation in the U.S. and other countries.

* Other names and brands may be claimed as the property of others.