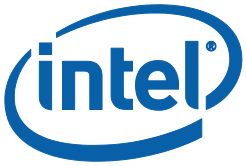

Intel® C++ Compiler v13.0 for Android* OS
Installation Guide and Release Notes

Document number: 327050-001US

04 June 2013

Table of Contents

1 Introduction

This document describes how to install the product, provides a summary of new and changed features and includes notes about features and problems not described in the product documentation.

Intel® C++ Compiler v13.0 for Android * OS is a component in Intel® System Studio for Android*. It is based on the compiler in Intel® C++ Composer XE 2013.

1.1 Change History

This section highlights important changes from the previous product version 13.0.0.006

- New option '-xatom_sse4.2' enabling optimizations for Silvermont target architecture.
- Support for Android NDK r8e.
- Compatibility and stability fixes for AOSP build.
- Performance improvements.

1.2 Product Contents

Intel® C++ Compiler for Android* OS includes the following components:

- Intel® C++ Compiler 13.0 for building applications that run on X86 architecture systems running the Android* operating system
- On-disk documentation

1.3 System Requirements

For an explanation of architecture names, see http://intel.ly/q9JVjE

- A PC based on an IA-32 or Intel® 64 architecture processor supporting the Intel® Streaming SIMD Extensions 2 (Intel® SSE2) instructions (Intel® Pentium® 4 processor or later, or compatible non-Intel processor)
    - Development of 32-bit applications is supported on either 32-bit or 64-bit versions of the OS
    - Development for a 32-bit on a 64-bit host may require optional library components (ia32-libs, lib32gcc1, lib32stdc++6, libc6-dev-i386, gcc-multilib) to be installed from your Linux distribution.
- For the best experience, a multi-core or multi-processor system is recommended
- 1GB of RAM (2GB recommended)
- 2GB free disk space for all features
- The following Linux distribution (this is the list of distributions tested by Intel; other distributions may or may not work and are not recommended -please refer to Technical Support if you have questions):
    - Ubuntu* 10.04 LTS, 11.04
- Linux Developer tools component installed, including gcc, g++ and related tools
- Library libunwind.so is required in order to use the –traceback option. Some Linux distributions may require that it be obtained and installed separately.

- The compiler does require an Android* build environment to run. You need to install either Android* NDK or an Android* source tree (AOSP)

Notes

- The Intel compilers are tested with a number of different Linux distributions, with different versions of gcc. Some Linux distributions may contain header files different from those we have tested, which may cause problems. The version of glibc you use must be consistent with the version of gcc in use. For best results, use only the gcc versions as supplied with distributions listed above.
- The default for the Intel® compilers is to build IA-32 architecture applications that require a processor supporting the Intel® SSE2 instructions -for example, the Intel® Pentium® 4 processor. A compiler option is available to generate code that will run on any IA-32 architecture processor.
- Compiling very large source files (several thousands of lines) using advanced optimizations such as -O3 and -ipo may require substantially larger amounts of RAM.
- The above lists of processor model names are not exhaustive -other processor models correctly supporting the same instruction set as those listed are expected to work. Please refer to Technical Support if you have questions regarding a specific processor model
- Some optimization options have restrictions regarding the processor type on which the application is run. Please see the documentation of these options for more information.

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 Technical Support

For information about how to find Technical Support, Product Updates, User Forums, FAQ, tips and tricks and other support information, please contact your Intel representative.

2 Installation

Please see the 'Getting Started' document for more information on installation.

The installation of the product requires a valid license file (default location: /opt/intel/licenses) or a serial number. If you received the product as a downloadable file, first unpack it into a writeable directory of your choice using the command:

```
tar –xzvf name-of-downloaded-file
```

Then change the directory (cd) to the directory containing the unpacked files and begin the installation using the command:

```
./install.sh
```

Follow the prompts to complete installation.

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 Silent Install

For information on automated or 'silent' install capability, please see http://intel.ly/ngVHY8.

2.2 Known Installation Issues

If you have enabled the Security-Enhanced Linux (SELinux*) feature of your Linux distribution, you must change the SELINUX mode to permissive before installing the Intel C++ Compiler. Please see the documentation for your Linux distribution for details. After installation is complete, you may reset the SELINUX mode to its previous value.

2.3 Installation Folders

The compiler installs, by default, under /opt/intel/CCAndroid13.0.x.nnn – this is referenced as <install-dir> in the remainder of this document. You are able to specify a different location, and can also perform a "non-root" install in the location of your choice.

Under <install-dir> are the following directories:

- bin – contains executables for the installed version
- lib – the lib directory for the installed version

- include – the include directory for the installed version
- Documentation
- man
- perf_headers

2.4 Removal/Uninstall

Removing (uninstalling) the product should be done by the same user who installed it (root or a non-root user). If sudo was used to install, it must be used to uninstall as well.

1. Open a terminal window and set default (cd) to any folder outside <install-dir>
2. Type the command: `<install-dir>/bin/uninstall.sh`
3. Follow the prompts

3 Intel® C++ Compiler

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

3.1 Compatibility

The IA-32 architecture default for code generation assume that Intel® Streaming SIMD Extensions 2 (Intel® SSE2) instructions are supported by the processor on which the application is run. This is inherited from the classic Intel® C/C++ Compiler XE 13.1 for Linux*. See below for more information.

3.2 New Features

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

- New option '-xatom_sse4.2' enabling optimizations for Silvermont target architecture.
- Support for Android NDK r8e.
- Compatibility and stability fixes for AOSP build.
- Performance improvements.
- Features from C++11 (-std=c++0x)
  - Additional type traits
  - Uniform initialization
  - Generalized constant expressions (partial support)
  - noexcept
  - Range based for loops
  - Conversions of lambdas to function pointers
  - Implicit move constructors and move assignment operators
  - Support for C++11 features in gcc 4.6 and 4.7 headers

- 64-bit long double type support (for compatibility with new NDKs)

- Intel® Cilk™ Plus runtime support is enabled for Android as a technology preview.

3.3 New Compiler Options

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

3.3.1 New options

- **-xatom_sse4.2**
- -vec-report6
- -f[no-]defer-pop
- -f[no-]optimize-sibling-calls
- -mmic
- -fextend-arguments=[32|64]
- -guide-profile=<file|dir>[,[file|dir],…]
- -opt-prefetch-distance=N[,N]
- -debug [no]pubnames
- -debug [no]profiling
- -grecord-gcc-switches
- -fno-merge-constants
- -check-pointers=<arg>
- -check-pointers-dangling=<arg>
- -std=c++11 (same as –std=c++0x)
- -[no-]check-pointers-undimensioned
- -no-]check-uninit functionality expanded to –check=<keyword>[,<keyword>…]. Use –
- check:[no]uninit for original functionality.
- -w3
- -W[no-]unused-parameter
- -W[no-]invalid-pch
- -noerror-limit removed
- -watch=<keyword>
- -nowatch
- -fimf-domain-exclusion=classlist[:funclist]
- -ipp-link={static|dynamic|static_thread}
- -fms-dialect=11
- -static-libstdc++
- -[no-]pie

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

3.4 Other Changes

3.4.1 Full support for latest Atom™ Processor instruction set

The compiler does support the instruction set of the latest Atom™ Processor, if the option -xatom_sse4.2 is set.

3.4.2 New Warning Level –w3 and Changes to Warning Levels in Intel® C++ Compiler

Here are the new warning levels as listed in “icc –help”:

-w<n> control diagnostics
n = 0 enable errors only (same as -w)
n = 1 enable warnings and errors (DEFAULT)
n = 2 enable verbose warnings, warnings and errors
n = 3 enable remarks, verbose warnings, warnings and errors

Previously, remarks were listed under –w2. This has been changed so that remarks are now enabled under the new warning level –w3.

3.4.3 Binary compatibility change with __regcall functions and elemental functions (i.e. __declspec(vector))

Intel® C++ Compiler v13.0 for Android * OS introduces an incompatibility with previous compiler versions in the way the __regcall calling convention is handled. Starting from this version, the RBX register is considered to be a callee-save register for __regcall routines, whereas in previous versions it was true only for IA32 targets. This could cause run-time fails if binaries built with different versions of the compilers are used together, so the compiler changes the name decoration scheme for __regcall routines using a new __regcall2__ prefix for mangling __regcall routines (previously the prefix was __regcall__). Binaries built with different versions of the compiler will therefore not link together successfully.

If you have functions declared with the __regcall interface or with the __declspec(vector) elemental function interface, code with these functions built with Intel® C++ Compiler v13.0 for Android * OS will not link with code with these functions built with earlier compilers. If you use these declarations, make sure to rebuild all necessary code with Intel® C++ Compiler v13.0 for Android * OS.

3.4.4 New libirng library for vectorizing random number generator functions added to Intel® C++ Compiler

The compiler can now automatically vectorize the drand48 family of random number generator functions provided by the C standard library. A new library, libirng.a and libirng.so, has been added to implement this support.

3.4.5 Intel® Cilk™ Plus “scalar” Clause removed

The “scalar” clause used optionally with Intel® Cilk™ Plus elemental functions is removed in this release. Please use the functionally equivalent “uniform” clause instead.

3.5 How to use Intel® C++ Compiler for building Android* Application

3.5.1 Required environment

Following versions of the OS and utilities are required:

- OS: Ubuntu 10.04, 11.04
- Version 3.81 or newer of make

3.6 Other Notes

3.6.1 Establishing the Compiler Environment

This version of the compiler has been tested with android-ndk-r8e from March 2013.

You must set the environment variables to point the Intel C++ Compiler to the GNU toolchain. The correct method depends on whether you use the NDK or the Android* Open Source Project (AOSP) environment.

Using the NDK environment:

export NDK= #set to the NDK root directory (example: /work/android-ndk-r8e/)

```
export ANDROID_SYSROOT=$NDK/platforms/android-14/arch-x86
export ANDROID_GNU_X86_TOOLCHAIN=$NDK/toolchains/x86-4.6/prebuilt/linux-x86
```

Using the AOSP environment:

export TOPDIR= #set to the AOSP root directory (example: /export/JB_WW11)

```
export ANDROID_SYSROOT=$TOPDIR/prebuilts/ndk/8/platforms/android-14/arch-x86
export ANDROID_GNU_X86_TOOLCHAIN=$TOPDIR/prebuilts/gcc/linux-x86/x86/i686-linux-android-4.6
```

You must set the PATH environment variable to include a path to Intel C++ Compiler binaries. One method for setting the environment variables quickly is to source the environment script, as shown below:

```
source /opt/intel/CCAndroid13.0.Y.XXX/bin/iccvars.sh
```

3.6.2 Instruction Set Default require Intel® Streaming SIMD Extensions 2 (Intel® SSE2)

This is inherited from the classic Intel® C/C++ Compiler XE 13.1 for Linux*.

When compiling for the IA-32 architecture, -msse2 (formerly -xW) is the default. Programs built with –msse2 in effect require that they be run on a processor that supports the Intel® Streaming SIMD Extensions 2 (Intel® SSE2), such as the Intel® Pentium® 4 processor and some non-Intel processors. No run-time check is made to ensure compatibility – if the program is run on an unsupported processor, an invalid instruction fault may occur. Note that this may change floating point results since the Intel® SSE instructions will be used instead of the x87 instructions and therefore computations will be done in the declared precision rather than sometimes a higher precision.

To specify generic IA-32, specify –mia32.

3.6.3 Features not carried forward from Intel® C/C++ Compiler XE 13.1 for Linux*

The following features in the Intel® C/C++ Compiler XE 13.1 for Linux* are not supported in current release:

- SSA – Static Security Analyser and OpenMP* support.
- Native compilation for MIC.
- Eclipse integration.
- Pointer Checker feature.

3.7 Known Issues

3.7.1 Guided Auto-Parallel Known Issues

Guided Auto Parallel (GAP) analysis for single file, function name or specific range of source code does not work when Whole Program Interprocedural Optimization (-ipo) is enabled.

3.7.2 Intel® Cilk™ Plus Runtime Support Limitations

The Intel® Cilk™ Plus runtime support for Android has the following limitations:

- Inter-operability with OpenMP, TBB is not supported
- Inter-operability with Intel® Vtune™ Amplifier is not supported
- Code which uses Cilk Plus should be linked with shared GNU C++ library from the Android NDK
- libcilkrts.so library need be linked explicitly with -lcilkrts

4 Disclaimer and Legal Information

INFORMATION IN THIS DOCUMENT IS PROVIDED IN CONNECTION WITH INTEL PRODUCTS. NO LICENSE, EXPRESS OR IMPLIED, BY ESTOPPEL OR OTHERWISE, TO ANY INTELLECTUAL PROPERTY RIGHTS IS GRANTED BY THIS DOCUMENT. EXCEPT AS PROVIDED IN INTEL'S TERMS AND CONDITIONS OF SALE FOR SUCH PRODUCTS, INTEL ASSUMES NO LIABILITY WHATSOEVER AND INTEL DISCLAIMS ANY EXPRESS OR IMPLIED WARRANTY, RELATING TO SALE AND/OR USE OF INTEL PRODUCTS INCLUDING LIABILITY OR WARRANTIES RELATING TO FITNESS FOR A PARTICULAR PURPOSE, MERCHANTABILITY, OR INFRINGEMENT OF ANY PATENT, COPYRIGHT OR OTHER INTELLECTUAL PROPERTY RIGHT.

A "Mission Critical Application" is any application in which failure of the Intel Product could result, directly or indirectly, in personal injury or death. SHOULD YOU PURCHASE OR USE INTEL'S PRODUCTS FOR ANY SUCH MISSION CRITICAL APPLICATION, YOU SHALL INDEMNIFY AND HOLD INTEL AND ITS SUBSIDIARIES, SUBCONTRACTORS AND AFFILIATES, AND THE DIRECTORS, OFFICERS, AND EMPLOYEES OF EACH, HARMLESS AGAINST ALL CLAIMS COSTS, DAMAGES, AND EXPENSES AND REASONABLE ATTORNEYS' FEES ARISING OUT OF, DIRECTLY OR INDIRECTLY, ANY CLAIM OF PRODUCT LIABILITY, PERSONAL INJURY, OR DEATH ARISING IN ANY WAY OUT OF SUCH MISSION CRITICAL APPLICATION, WHETHER OR NOT INTEL OR ITS SUBCONTRACTOR WAS NEGLIGENT IN THE DESIGN, MANUFACTURE, OR WARNING OF THE INTEL PRODUCT OR ANY OF ITS PARTS.

Intel may make changes to specifications and product descriptions at any time, without notice. Designers must not rely on the absence or characteristics of any features or instructions marked "reserved" or "undefined". Intel reserves these for future definition and shall have no responsibility whatsoever for conflicts or incompatibilities arising from future changes to them. The information here is subject to change without notice. Do not finalize a design with this 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_number/

BlueMoon, BunnyPeople, Celeron, Celeron Inside, Centrino, Centrino Inside, Cilk, Core Inside, E-GOLD, Flexpipe, i960, Intel, the Intel logo, Intel AppUp, Intel Atom, Intel Atom Inside, Intel Core, Intel Inside, Intel Insider, the Intel Inside logo, Intel NetBurst, Intel NetMerge, Intel NetStructure, Intel SingleDriver, Intel SpeedStep, Intel Sponsors of Tomorrow., the Intel Sponsors of Tomorrow. logo, Intel StrataFlash, Intel vPro, Intel XScale, InTru, the InTru logo, the InTru Inside logo, InTru soundmark, Itanium, Itanium Inside, MCS, MMX, Moblin, Pentium, Pentium Inside, Puma, skoool, the skoool logo, SMARTi, Sound Mark, Stay With It, The Creators Project, The Journey Inside, Thunderbolt, Ultrabook, vPro Inside, VTune,