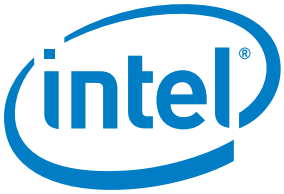

Intel® System Debugger 2019

Release Notes for Linux host*

5 November 2018

Contents

1 Introduction

This document covers release specific information of all components Intel® System Debugger 2019 Initial Release for Linux* host including

- Intel® System Debugger - System Debug
- Intel® System Debugger - System Trace

2 Supported Host Operating Systems

Intel® System Debugger 2019 Initial Release for Linux* host supports the following operating systems:

- Fedora 27
- Ubuntu 16.04 LTS

3 New in This Release – 2019 Initial Release

This section lists new features of the Intel® System Debugger 2019 Initial Release for Linux* host.

3.1 New Features and Bug Fixes

- Support 8th Gen Intel Core Processor i7-8565U, i5-8265U, i3-8145U (Whiskey Lake-U)

3.1.1 Intel® System Debugger – System Trace

- TDE runtime error is fixed
- Issues regarding starting traces for several platforms are fixed
- Error seen during capturing traces in Trace Hub Memory is fixed

3.2 Known Issues and Workaround

- Intel® System Debugger tools have currently issues with Centos 7.4 and RHEL 7.5. Please contact to support in case using a host machine with Centos 7.4 and RHEL 7.5

3.2.1 Intel® System Debugger – System Trace

- No trace messages collected, packets dropped
  - ***Issue:*** No trace messages collected while running system trace on a Linux host and the following error message is shown in the console when stopping the trace: *"[WARN] [npk_config_api] Packets were dropped during trace! Lost X packet(s) with overflow"*
  - ***Workaround:*** Lower the reading speed by adding the following parameter in "Streaming Decode Preferences" (accessed by clicking the menu button at the top right corner of the Trace Capture window, next to "Start Tracing" button). In the “Decoder Parameters” section add the following entry:
    - Decoder: IPC_Connector
    - Name: bytes-per-read
    - Value: 1024

4 Known Issues

This section lists the known issues and the corresponding workarounds for each tool of Intel® System Debugger.

4.1 Intel® System Debugger – System Debug

4.1.1 Note for Probes

- Intel® Silicon View Technology (Intel® SVT) Closed Chassis Adapter (CCA) and Intel® In-Target Probe (Intel® ITP) - XDP3 are supporting hot plug/unplug from the target, whereas Intel® Direct Connect Interface (Intel® DCI) USB Debug Class (DbC) debug cables are bidirectional. In case of losing connection with the probe debugger will post Power Loss event. If target was running, probe disconnect would have no effect on target, please reconnect probe to continue debugging. In case target was halted, debugger will lose debug Context, which is crashing target.
- BE AWARE: Any accidental probe removals during halt would crash the target. Please reboot target and restart debugging session.

4.1.2 Handling of Target Access Issues

- Please ensure that the target system is running a recent firmware version allowing JTAG debugging. If a "no threads found" error message appears, and applying the restart recipe (see below) doesn't help the situation, please contact customer support for that specific platform.
- If target access issues occur, please apply the following recipe for a complete reset of all the components that take part in JTAG debugging, in this order:
  1. Close the debugger.
  2. Unplug both the USB and power connectors from the Intel ITP-XDP3 probe at the same time, so no cable is connected to the Intel ITP-XDP3 probe anymore.
  3. Using the task manager, kill the process MasterFrame.HostApplication.exe if it's running.
  4. Power off the target system.
  5. Plug the USB and power connectors back into the Intel ITP-XDP3 probe.
  6. Power on the target system.
  7. Start the debugger.

4.1.3 Troubleshooting Target Stability issues

- **Considerations**

  The Intel® System Debugger requires a great deal of data to construct a full source-level view of the target state. For a simple operation such as step, there can be dozens to hundreds of individual accesses to target state. In the case where one of these accesses crashes the target, it can be difficult to identify the exact root cause.

- **Techniques for isolating problems**
  1. Close as many GUI panels as possible: Each panel in the debugger (e.g., the registers panel, MSR panel, etc.) is self-updating. If many panels are open, then there are many target accesses. Closing panels is one way to isolate which functional group is causing the problem.
  2. Use the TCI_LOG flag to turn on API logging: The debugger can log all transactions made to the communication backend. Although this information is primarily intended for developers, it may also be useful for identifying root causes of stability issues. Enable logging by setting the environment variable TCI_LOG=1. For example, you can edit the startup script to include the command *set TCI_LOG=1*. On request you may additionally include the command *set ENGINE_LOG=1*, which will however slow down debugger startup considerably to provide very detailed logs. This technique causes a log file to be created in the folder *C:\users\<name>\.sysdbg_2019*, where <name> is the user login.

4.1.4 Troubleshooting Simics Simulated Target

- The error *Connection fails with error message: "E-2201 TCP/IP Socket initialization or connection failed: Connection to host localhost:9123 failed, please verify server is running, and that the address and port are correct"* indicates that the Intel® System Debugger could not connect to the Simics software. This can be caused by:
  - There is a local firewall running which permits TCP accesses to localhost:9123

4.1.5 Supported targets on Red Hat Enterprise* Linux* is limited

- The following target connections are not working on Red Hat Enterprise* Linux:
  - Intel DCI USB 3.x DbC
  - Simics (Demo Mode)

4.1.6 Debugger hangs during startup on Ubuntu*14.04

- After starting the debugger, the GUI is not render properly and seems frozen. Please uncomment the following lines in xdb.sh to solve the issue:
  - #export LD_LIBRARY_PATH=${IPCAPIBINDIR}${LD_LIBRARY_PATH:+:}${LD_LIBRARY_PATH:-}
  - #export XDB_UFC_DISABLE=1

4.1.7 Problems connecting to the targets with OpenIPC on Ubuntu* 14.04

- On Ubuntu 14.04, if user encounters GLIBC related problems while connecting to targets with OpenIPC please uncomment the following lines in xdb.sh to make it work:

- #export LD_LIBRARY_PATH=${IPCAPIBINDIR}${LD_LIBRARY_PATH:+:}${LD_LIBRARY_PATH:-}
- export XDB_UFC_DISABLE=1

4.1.8 Debugger start silently fails on Fedora* 21 with default OpenJDK under Gnome/Unity/KDE

- After starting a debugger script, a splash screen is only displayed, but the debugger exits silently. No diagnostic is issued.
- To overcome this issue you may apply one of the following workarounds:
  - **Install and use JRE from Oracle:** Download jre-8u25-linux-x64.tar.gz from www.oracle.com and unzip the package and add the /jre1.8.0_25/bin to PATH
  - **Change GTK theme to "Raleigh":**
    1. On KDE use lxappearance: install, goto "SystemSettings->Application Appearance->GTK+Appearance" and change widget style to "Raleigh"
    2. On Unity/Gnome use gtk-theme-switch2: Install and launch the gui, switch to "Raleigh"
  - Both utilities can be installed from the packet manager.

4.1.9 Debugger start fails on Fedora* 23 and Fedora*24

- Using the debugger on Fedora* 23 and 24 requires a downgrade of java and to run it using sudo. To downgrade Java to 1.7 follow these steps:
  1. Download http://download.oracle.com/otn-pub/java/jdk/8u60-b27/jre-8u60-linux-x64.rpm
  2. Install with: sudo dfn install jre-8u60-linux-x64.rpm
  3. Add to alternatives: sudo alternatives --install /usr/bin/java java /usr/java/jre1.7.0_79/bin/java 1
  4. Choose 2nd option from alternatives: sudo update-alternatives --config java
  5. This should let you select the newly installed 1.7 java as option 2. Verify that it switched by doing: java –version. This should display 1.7 instead of 1.8
  6. Run the debugger using sudo: sudo ./xdb.sh

4.1.10 Functional differences of 60-pin vs. 10-pin JTAG

- Some platforms do not implement the full 60-pin debug port that is traditionally used on Intel systems, in this case the functionality of the debugger will be limited, especially in the following areas:
  - Detection of reset by the debugger
  - Initiation of reset by the debugger
  - Halting the target at the reset vector

4.1.11 Platform reset policy may inhibit debugger operation

- Some platforms implement reset in such a way that the debugger may not be able to gain control of the target immediately after reset. This impacts the debugger operation in the following ways:
    - The debugger may not be able to restore breakpoints after reset; the breakpoints may appear "enabled" in the GUI but will not in fact be enabled in the target.
    - The debugger will not halt automatically at the reset vector; if the user wishes to debug early in the boot process there will be no way to manually initiate a halt quickly enough.

4.1.12 Platform security policy may inhibit debugger operation

- In some platforms the security policy may disable JTAG access to the CPU, this is intended to prevent reverse-engineering. In this case the Intel® System Debugger will successfully connect to the target, however it will not be able to discover any CPUs on the JTAG bus, and will warn the user that no CPUs are available. To resolve this issue please ensure that that platform firmware has enabled access to the CPUs via JTAG, this is typically done by flashing a special "debug" firmware into the target.
- Also note that in some cases the CPU or CPU module may have physically disabled JTAG access, especially in production or near-production versions. In this case please work with the platform business unit to obtain JTAG-enabled hardware.

4.1.13 Target power management and platform power policy on tablet systems may inhibit debugger operation

- On some tablet designs the platform architecture includes aggressive power-management of the CPU, this will impact debugger operation in that a CPU in a low-power state cannot be accessed via JTAG. This will manifest in a variety of ways:
    - Error messages indicating that "no threads are available"
    - Error messages indicating that "target could not halt"
    - No threads displayed in the hardware threads window
- In general these problems will be mitigated by doing one or more of the following:
    - Ensure that the CPUs are in an active state when the debugger is first started (e.g. in early boot firmware where no power management is present)
    - Ensure that the OS has a workload that will inhibit low power states (e.g. play a video, run animated wallpaper, etc.)
    - Disable low-power states in the platform when possible (generally a BIOS setting)

4.1.14 Platform reset implementation may limit debugger-initiated reset:

- Debugger-initiated reset is not an industry standard feature, it is implemented using sideband signals on the 60-pin Intel ITP-XDP3 port, and it is subject to the reset implementation on the target system. Some targets may not reset reliably via the

debugger's reset/restart feature, this will typically result in a message such as "WARNING: target did not halt after reset, forcing a halt" being displayed in the debugger console, followed by additional error messages. In this case the user may need to manually initiate a reset on the target via buttons, debug card, etc.

4.1.15 Platform reset implementation may limit ability to halt at reset vector:

- Halting the CPU at the reset vector (first instruction fetched) is a CPU/platform dependent feature and may be limited due to target implementation details. The main impact to the user is:
    - **inability to debug early platform boot code due to a runaway target**. In this case it may be necessary to build a special firmware with a hard-coded infinite loop early in the boot flow.
    - **inability of the debugger to re-apply breakpoints after target reset**. In this case the debugger may show breakpoints as "enabled" in the GUI, but they will not be installed in the target, the user should manually halt the target, disable, re-enable breakpoints to ensure they are applied correctly.

4.1.16 Intel Atom® Processor Z3xxx and E3xxx specific issues

- **Platform power management policy may limit debugger control of the target:** The platform power-management policy may include power management of the CPUs, this may limit availability of debugger features when the threads are in a low-power state.
- **Launching the debugger when the target is off or in a low-power state may cause unexpected behavior:** The debugger needs to see all threads when connecting to the target in order to correctly initialize both the debugger state, and the debug resources in the target. The Debugger will report the number and type of threads observed during initialization (e.g. "INFO: Connected to Processor type: <name> (4 threads)") the user should confirm that this matches the expected configuration. Resetting the target (with the debugger running) should clear the issue and cause all expected threads to show up in the hardware threads window.
- **Hardware Threads window may show no threads/partial threads/disabled threads:** Hardware Threads window may show no threads/partial threads/disabled threads due to the CPUs appearing/disappearing from the JTAG scanchain, and/or showing up in the JTAG scanchain as "disabled". This condition should only happen when the target is in "run" state, when halted (e.g. from hitting a breakpoint) the debugger should correctly show all threads. *if the debugger halts and all threads are not shown* then the debugger is in an incoherent state and you may need to restart your debug session.
- **User-initiated Halt may occasionally return errors, especially if the target is in a low-power state:** Errors include "E-2201: Target has no active threads, this operation is not permitted." and "E-2201: Target did not halt execution." This condition occurs when the debugger attempts to halt the target, but the CPUs are asleep and therefore unresponsive

to debugger commands. Workaround: manually bring the CPUs out of sleep (e.g. by fiddling with the tablet) *or* try repeatedly to halt via the debugger, after 2-3 tries the target typically wakes up.

- **Kernel module load configurations may be unreliable:** Due to limitations in silicon debug features, kernel module load notifications may not function correctly on Intel Atom® Processor Z3xxx and E3xxx based platforms. The following should be observed to work around this:
    - Software breakpoint in the target must be set prior to using xdbntf. The breakpoint should be in an unused/unreachable code location, it is not necessary that the breakpoint is ever hit, its purpose is to enable the Intel® System Debugger redirection logic in the Silicon, which will allow the kernel module notifications to function correctly.
    - In some cases the notification will only partially work. This will manifest as a hung system, with the Intel® System Debugger indicating a "running" state. In this case the user should manually halt the target, at which point the debugger will detect the notification, consume it, and resume target execution.

4.1.17 Opening Help content on system with Java* JRE 1.7.x can lead to Java SEGV

- Opening "Help > Help Contents" can lead to a Java segmentation fault triggered by the Intel® System Debugger user interface. Workaround is to access the Intel® System Debugger User's Guide at https://software.intel.com/en-us/xdb_2017_ug instead.

4.1.18 Debugger cannot be launched in user mode once it has been launched as root

- The Intel® System Debugger root installation requirement implies that if the debugger is launched as root instead of regular user only once, access permissions will not permit launch as regular user afterwards.
- Message: "daltempinfo.txt" owned by root:root => "Cannot update registration of MasterFrame: Access denied for writing to '/usr/share/Intel/DAL/MasterFrame'"
- Workaround: Install as root and exclusively run debugger as normal user (or exclusively run debugger as root, but do not switch root/normal user)
- Workaround: (after 1st invocation of XDB as root) run this command in root shell: "chmod -R ugo+rw /usr/share/Intel/"

4.1.19 PCI bus scan in PCI tool very slow on Linux* host

- PCI bus scan in the PCI tool add-on is very slow on Linux* host and interrupting the scan can lead to debugger hang. It is recommended to only scan small bus ranges (e.g. 0-10) and to not Help contents loading from within debugger not supported on Ubuntu* 14.04 with Java* JRE 1.7.x

4.1.20 Help contents loading from within debugger is not supported on Ubuntu* 14.04 with Java* JRE 1.7.x.

- Invoke help from terminal shell with the following command:
  $ firefox <INSTALLDIR/system_studio_2019/documentation_2019/en/debugger/system_studio_2019/system_debugger/system_debug/cl/index.htm

4.1.21 Only installation as "root" or "sudo" user supportd

- Currently, only the "root" and "sudo" installation types are supported using the Linux* hosted Intel® System Studio installer. The "user" installation type should not be selected on Linux hosted systems.

4.1.22 Support for Intel Atom® Processor bitfield editor register views

- To receive information on how to access bitfield editor chipset register views for Intel Atom® Processor s, please send an email to IntelSystemStudio@intel.com for details.

4.1.23 Locals Window updates can be slow

- The local window updates may be slow in many cases where it is evaluating many large structs or in scopes with many locals. If the slowness is noticed, it is recommended to close the locals window.

4.1.24 Kernel Threads Window Population Slow

- The Linux* OS awareness plug-in for the Intel® System Debugger includes a Kernel Threads Window, that displays all current kernel threads and information about their state. When the Kernel Threads Window is opened it can take several seconds before the actual content is displayed. The initial window content of "No data." will disappear once kernel thread data is available. This can take up to 20 seconds.

4.1.25 Debugger puts a file system lock on symbol files

- Currently when a symbol file is loaded in the debugger a file system lock is placed on this file to prevent other processes from deleting or modifying this file. If this lock is preventing you from recompiling your program, simply use the Unload feature found in the Load Dialog. Unloading a symbol file will release the file system lock and allow you to modify or delete the symbol file without exiting the debugger.

4.1.26 Function and file information not listed for watchpoints in breakpoint window

- When setting a data breakpoint (watchpoint) the breakpoint listing in the breakpoint window does not contain file and function information for the data breakpoint.

4.1.27 Local variables and evaluation windows do not display multi-dimensional arrays correctly

- Multi-dimensional arrays are displayed as vectors in the debugger's local variables window and evaluations window suppressing one dimension of the array.

4.1.28 Evaluation window for global variables may be missing type information

- When evaluating a global variable in a debugger evaluation window the variable name and it's value are displayed, but type information may not be displayed.

4.1.29 Writing to a non-writable vector registers may incorrectly update register value display

- In the vector register window it is possible that a write to a vector register seems to have been successful, when a new value was entered from within said window, despite the register being non-writable at the time and not actually having been updated.

4.1.30 Root or Sudo Access required for Intel® System Debugger Install

- To be able to install the Intel® System Debugger it is necessary to either launch the tool suite installation script install_GUI.sh with root privileges or to select "install as root" or "install as root using sudo" during the Intel® System Studio installation process. Installation without root privileges will not be successful.

4.1.31 Memory Writes to Uninitialized Memory

- Memory writes to uninitialized or read-only memory (this includes setting software breakpoints or accessing memory mapped registers) can lead to a crash of the target or a loss of the target control. The debugger will not prevent these memory accesses when requested by the user (e.g. changing instructions in the disassembly window).

4.1.32 Flash Writer disables pre-existing Breakpoints

- Flashing the BIOS will disable all code breakpoints and data breakpoints you may have had set prior to using the flash writer.

4.1.33 Master Flash Header Read/Write not supported for Intel Atom® Processor CE4200

- On the Intel Atom® Processor CE4200 the Master Flash Header serves as a road map for the contents of flash that are processed by security and host firmware. It contains the location and size of each element in the flash, as well as a list of host firmware images that the security processor will attempt to boot. Currently the flash writer plug-in for the Intel® System Debugger does not support writing or modifying the Master Flash Header.
- It is of course possible to use the terminal Master Flash Header commands mfhlist mfhinfo and mfhinit in conjunction with the Intel® System Debugger flash writer plug-in.
- mfhlist provides the location of the Master Flash Header entries and where the current platform boot configuration expects the various flash images to be put. It's output can be used as a guidance for setting the start address when using the flash writer plug-in.
- For NOR Non-Trusted Boot and NAND/eMMC Non-Trusted boot can be configured such that target boot is possible even if no Master Flash Header is present on the platform.
- eMMC Trusted Boot does require the presence of a Master Flash Header and requires that the actual memory layout does match its contents.

- Please read the Platform User Guide closely for further details on the Master Flash Header and its usage.

4.1.34 ERROR: E-2201: Invalid thread/device number selected

- Running the Intel® System Debugger on 64-bit Fedora Linux requires using a correct java version. Download and install Oracle Java jdk-7u75-linux-x64 (or jre-8u40-linux-x64.rpm) and point the path to those versions in start_XXXX.sh directly:
    - - /usr/java/jdk1.7.0_75/bin/java -jar ... or
    - - /usr/java/jre1.8.0_40/bin/java -jar ...

4.1.35 Opening the web help ("Help"=>"Help Contents") on a Linux host OS might cause the debugger to crash.

- This can be worked around by uninstalling the package libwebkitgtk-1.0-0 using the Linux distribution's package manager. Alternatively, the documentation can be opened from the "Intel® System Debugger Documentation Index" page.

4.1.36 Failure to break on init of Linux driver with the kernel version greater than 3.13

- The System Debug cannot find where the "init" section has been loaded in memory when the Linux kernel version is greater than 3.13. It causes the initial break malfunction.

4.2 Intel® System Debugger – System Trace

4.2.1 Installation into an user-provided Eclipse

- To install Intel® System Debugger – System Trace into an user-provided Eclipse, the necessary prerequisites must be installed manually by running the following commands on the command-line from within the user-provided Eclipse installation – this requires an Internet connection:
- **Eclipse Mars:**
    - `eclipsec.exe -nosplash -application org.eclipse.equinox.p2.director -installIUs org.eclipse.jdt.feature.group -repository http://download.eclipse.org/releases/mars`
    - `eclipsec.exe -nosplash -application org.eclipse.equinox.p2.director -installIUs org.eclipse.jetty.websocket.api -repository http://download.eclipse.org/jetty/updates/jetty-bundles-9.x/9.2.13.v20150730/`
    - `eclipsec.exe -nosplash -application org.eclipse.equinox.p2.director -installIUs org.eclipse.jetty.websocket.client -repository http://download.eclipse.org/jetty/updates/jetty-bundles-9.x/9.2.13.v20150730/`

- **Eclipse Neon**
  - `eclipsec.exe -nosplash -application org.eclipse.equinox.p2.director -installIUs org.eclipse.jdt.feature.group -repository http://download.eclipse.org/releases/neon`
  - `eclipsec.exe -nosplash -application org.eclipse.equinox.p2.director -installIUs org.eclipse.pde.feature.group -repository http://download.eclipse.org/releases/neon`
  - `eclipsec.exe -nosplash -application org.eclipse.equinox.p2.director -installIUs org.eclipse.jetty.websocket.api -repository http://download.eclipse.org/jetty/updates/jetty-bundles-9.x/9.3.9.v20160517/`
  - `eclipsec.exe -nosplash -application org.eclipse.equinox.p2.director -installIUs org.eclipse.jetty.websocket.client -repository http://download.eclipse.org/jetty/updates/jetty-bundles-9.x/9.3.9.v20160517/`

4.2.2 Event Distribution View not showing all data under certain circumstances

- When doing a target power-off/power-on cycle, the Event Distribution View might only show events after the power on.

4.2.3 Message View Display issues with horizontal scrolling

- The Message View has display rendering issues related to horizontal scrolling. The selected cell contents in the table may get painted at wrong positions, hiding the contents underneath it. This issue is caused by a following unresolved Eclipse platform bug entry for Linux GTK based systems: https://bugs.eclipse.org/bugs/show_bug.cgi?id=383492
- A solution for this issue is under development and will be deployed using a product update.
- The current workaround is avoid the horizontal scrolling of the Message View. Most rendering issues disappear if the horizontal scrollbar remains at its leftmost position. This can be achieved by maximizing the Eclipse window, reducing the width of columns or by moving required columns to the left side of the view.

4.2.4 BIOS and CSME checkboxes not fully functional

- Some BKC images program the SWDEST registers for BIOS and CSME after a target reset automatically. This causes BIOS and CSME traces to be always switched on regardless of the selection made for the BIOS/CSME configuration checkboxes in the configuration area

4.2.5 The installation must be done as root to guarantee correct functionality of System Trace

- For connection functionality to hardware, files need to be install in /lib/udev and /opt/intel locations, which can only be done with super-user rights.

4.2.6 Target platform re-connect not reliable

- Disconnecting and re-connecting to the target may work unreliable, which causes the capture process not to work.

4.2.7 Previous System Trace feature workspace data not supported

- If you used an older version of the Intel® System Studio System Trace feature, the workspace used previously will no longer be usable with the current update. Either delete the previously used workspace (e.g. $HOME/workspace) or ensure to use a different workspace together with the System Trace feature.

4.2.8 Ordering of time stamps during live decode and file decode may differ

4.2.9 Power states problems

- When the target transitions into a low power state configuration (attempts), starting and stopping trace fails. If attempted, the GUI may fail to detect this and end up in an inconsistent state where no further target interaction is possible.

4.2.10 Incorrect target connection status after reset

- In some rare cases it may happen that after a target reset the target status shown in the "Target Connection" view is incorrect and still reports that the target is in reset state. In this case please reset the target again, which triggers an update of the target status

4.2.11 Trace Viewer may become unresponsive after workspace upgrade

- After workspace upgrade you may encounter something like the following error:

```
1. 16:07:36 [ERROR] Cannot send message, target connection server is not
   running.
2. 16:07:36 [ERROR] Server is unresponsive
3. 16:07:36 [ERROR] Unable to restore API state. Target may be
   incompatible. See server logfile for details.
```

- To overcome the issue, just re-select the current target to perform live trace capture.

4.2.12 Incorrect Target Power status indication

- For some Targets, upon successful connection “No Power” status is indicated in Target Connection Panel and hence tracing operations fail. To overcome this issue, please set option "Allow target access without power indication" in Window/Preferences/System Trace.

5 Change History

5.1 Intel® System Debugger 2019 Beta

- Support 8th Gen Intel® Core™ Processors (Coffee Lake-S) / Intel® H370 Chipset, Intel® H310 Chipset, Intel® B360 Chipset for Consumer (Cannon Lake PCH)

5.1.1 Intel® System Debugger – System Trace

- Upgrade to support Eclipse Oxygen
- Search result visualization is added

6 Legal Information

No license (express or implied, by estoppel or otherwise) to any intellectual property rights is granted by this document.

Intel disclaims all express and implied warranties, including without limitation, the implied warranties of merchantability, fitness for a particular purpose, and non-infringement, as well as any warranty arising from course of performance, course of dealing, or usage in trade.

This document contains information on products, services and/or processes in development. All information provided here is subject to change without notice. Contact your Intel representative to obtain the latest forecast, schedule, specifications and roadmaps.

The products and services described may contain defects or errors known as errata which may cause deviations from published specifications. Current characterized errata are available on request.

Intel technologies' features and benefits depend on system configuration and may require enabled hardware, software or service activation. Learn more at Intel.com, or from the OEM or retailer.

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

Intel, the Intel logo, Xeon, and Xeon Phi are trademarks of Intel Corporation in the U.S. and/or other countries.

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

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