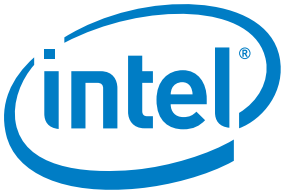


Intel® System Debugger 2019

Release Notes for Windows host*

8 October 2018

Contents

1 Introduction

This document covers release specific information of all components of Intel® System Debugger 2019 Initial Release for Windows* host which includes following tools

- Intel® System Debugger - System Debug
- Intel® System Debugger - System Trace
- Intel® Debug Extensions for WinDbg* (for Windows* targets)

2 Supported Host Operating Systems

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

- Windows 10 & SPs

3 New in This Release – 2019 Initial Release

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

- N/A

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 at the same time, so no cable is connected to the Intel ITP-XDP3 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.
  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:
  1. There is a local firewall running which permits TCP accesses to localhost:9123
  2. The license for Simics is expired. Either update to the latest version of Intel® System Debugger or contact technical support.

4.1.5 Symbol offset issue with GCC compiled firmware

- When debugging GCC compiled firmware the automatic symbol load feature loads the symbols to the wrong offset. Symbols need to be reloaded to the correct offset manually. Command "efi loadthis" shows the path to loaded symbol file.
- Example:
  - > efi loadthis
  - INFO: Searching backwards from 0x0000000077AC7000 to 0x00000000779C7000 for PE/COFF header (semantics=MEM align=0x00001000 range=0x00100000)
  - INFO: Found PE/COFF module at 0x0000000077AC7000 - 0x0000000077AD72A0 Entrypoint: 0x0000000077AC72AF (size: 66208 bytes)
  - INFO: Successfully loaded debug symbols found at offset 0x0000000077AC6FC0 instead of 0x0000000077AC7000:

    - C:\uefi_64_gcc\Build\Vlv2TbltDevicePkg\DEBUG_GCC46\X64\Vlv2TbltDevicePkg\SmBiosMiscDxe\SmBiosMiscDxe\DEBUG\MiscSubclass.debug
- Use eval command to get the module entry point address.
- Example:
    - > eval _ModuleEntryPoint
    - _ModuleEntryPoint [at address 0x0000000077AC728F] : type procedure is not evaluatable [_MODULEENTRYPOINT]
- NOTE: The default module entry point function name may change from module to module. Check the module code to find the correct entry point function name.
- If evaluated entry point doesn't match the one shown by efi loadthis command, the symbols need to be reloaded to an adjusted address.
- Reload the module symbol file with the correct offset:
    - LOAD /NOLOAD /OFFSET = <offset> OF <filename>
- Where <filename> is the path to the module symbol file from efi loadthis command result print.
- Where <offset> is original load address + (original entry point address - evaluated entry point address)
- Example: To get the adjustment subtract evaluated address from the original entry point address: 0x0000000077AC72AF - 0x0000000077AC728F = 0x20. Add the adjustment to the original module load address = 0x0000000077AC6FC0 + 0x20

4.1.6 Intel Atom® Processor Z35xx Support Limitations

- IO breakpoints are not working
- Reset break is currently not supported
- No power management support due to hardware limitations. Ensure that all OS power management is disabled, including low power C-states, and sleep states (S0iX and S3)
- Only supported on Windows* host
- The target needs to have the the eXtended Debug Port enabled (unlocked).

4.1.7 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.

4.1.8 Intel Atom® Processors N4200, N3350, x7-E3950, x5-39xx (Apollo Lake)

- **Re-connection to the target is not reliably working:** The connection to the target is sometimes failing if a previous established connection was disconnected. Power cycling of the target usually resolves the issue.
- **Resuming the target after hitting software breakpoint is not reliably working:** Executing "run" command doesn't work reliably after software breakpoint is hit. The way to solve it is to either step after the breakpoint or remove the breakpoint.

4.1.9 Problems with setting breakpoints for Intel® Processor code-named "Coffee Lake" from disassembly view

- Setting breakpoint by double clicking on an instruction in disassembly window may cause an error in Intel DAL and the breakpoint might not be set correctly. Please use "Create Breakpoint" command to set it.

4.1.10 3rd generation Intel® Core™ Processor support only available on request

- Intel® System Debugger support for the 3rd generation Intel® Core™ Processor code-named "Ivy Bridge" is currently only available on request. Please contact Intel® System Studio support via https://premier.intel.com or IntelSystemStudio@intel.com to request the patch necessary to enable it.

4.1.11 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

4.1.12 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.

4.1.13 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.

4.1.14 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)

4.1.15 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.

4.1.16 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.

4.1.17 There could be loss in control while doing stepping around reset

- On some targets, there could be loss in control while doing stepping around reset.

4.1.18 Intel® System Debugger use in conjunction with PVT product

- In order to use Intel® System Debugger in conjunction with a PVT product, please update OpenIPC and Intel DAL files inside <INSTALLDIR>/env.d folder.

4.1.19 Sharedinfo.txt not writable

- The debugger is dependent on writing to a file called "sharedinfo.txt" which is normally installed at "C:\ProgramData\Intel\DAL\MasterFrame\sharedinfo.txt". Default security policy on the host system may cause this file to be read-only after Intel® System Studio installation. If this is the case then the user will observe an error message when connecting to the target. To resolve this issue either (a) run the debugger with administrator privileges or (b) change the permissions on this file (and any containing folders) to be writable.

4.1.20 Network access for MasterFrame.HostApplication.exe blocked by Microsoft* Windows* firewall

- The debugger utilizes a server process to access the target, this "Masterframe" process will attempt to access the network when initializing, and this access may trigger a warning from the Microsoft* Windows* firewall. This is expected and the user should allow the access for the debugger to function correctly.

4.1.21 MSDIA DLL not registered

- The debugger is dependent on a shared library provided by Microsoft* for access to debug information generated with the Microsoft* compiler. In some cases this library will not be correctly registered on the host system, which will lead to an error message when trying to load symbols for modules (e.g. EFI modules) that are compiled using the Microsoft* toolchain. The user can resolve this issue by manually registering the correct library using an administrator command prompt.
- From a command line window go to: C:\Program Files (x86)\Common Files\microsoft shared\VC
- Run: regsvr32 msdia90.dll
- if the msdia90.dll file is not found there, reinstall the Microsoft* Visual C++ 2008 Redistributable Package after downloading it from Microsoft's website

4.1.22 XDB Grey unresponsive window

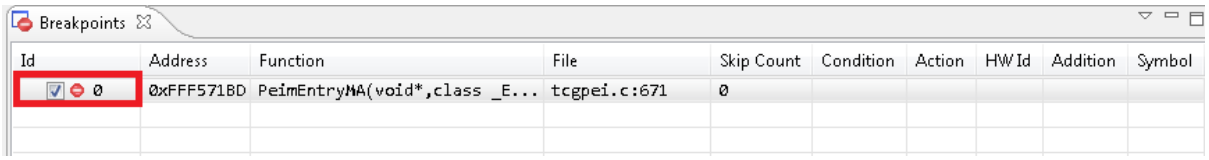
- On some hosts the System Debugger may come up with an empty gray unresponsive window. If this is observed then please verify that the Intel Remote Monitor opt-in dialog is not hidden behind the main window. If this dialog is present, clicking to dismiss it will allow the debugger to launch normally.

4.1.23 Debugging a module on the watchlist

- The debugger will stop the target when a module on the watchlist gets loaded. Target will stop at the debug agent code and load the symbols for the module on the watchlist. To debug the module you need to browse to the module source files and set a breakpoint somewhere in the code.
  1. Add your module into the watchlist.
  2. Wait for the module to get loaded and target to stop.
  3. Open the “Source Files” window. The window is visible at the left side of the screen.
  4. Browse to module source code and set a breakpoint.
  5. Run the target. Execution stops at the breakpoint and you can debug the module.

4.1.24 Using breakpoints

- After reset, the breakpoints will get disabled in target even though they look enabled in the GUI. After target stops at a watchlist module load, the breakpoints need to be disabled and re-enabled for them to work.
  1. Open the breakpoints window.
  2. Disable breakpoint by unchecking the enable box.
  3. Re-enable breakpoint by checking the enable box.
  4. Run the target.



4.1.25 Support for Intel Atom® processor bitfield editor register views

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

4.1.26 Failure of initial debugger re-connection can lead to debugger hang

- In some cases the debugger will hang when re-connecting to the target. This is typically observed when the initial connect attempt has failed (e.g. due to the CPUs or target being powered off). Workaround: always shut down the debugger and re-launch it when re-connecting to the target.

4.1.27 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.

4.1.28 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.

4.1.29 Debugger puts a Windows 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.

4.1.30 Memory Reads with Uninitialized Memory

- If any window other than the memory window is open on uninitialized memory (e.g. Assembly Window), memory read attempts may lead to the target entering an undefined state.

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.

4.2 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 Issues of Architectural Event Trace (AET)

- Restore of an existing workspace with AET traces in it might lead to an error message in the Message View. Workaround: close and reopen the Message View.
- AET Branch Trace Memory (BTM) enabling will not be disabled on Stop Capture, which will slow down the target system significantly. Workaround: reboot target machine after Stop Capture.
- AET configuration editor requires target connection for proper restore when launching Eclipse* with an existing workspace, which contains AET configurations. Workaround: create a new workspace.
- AET time sync overflows shown in the Message View when activating many AET sources. Workaround: enable only the sources needed for the user's debugging use case.

4.2.3 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.

4.2.4 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 PCH Power Management may not be configurable on Intel® 6th Generation Intel® Core Processors (code-named "Skylake") production systems

- PCH Power Management may not be configurable on Skylake production systems because of platform settings. During the Intel® Trace Hub configuration the following error message could appear in the Eclipse* console when trying to enable PCH Power Management on a production system:

```
10:48:55 [WARN ] Unable to halt the processor to configure PMC trace.
PMC trace may not be configured correctly
```

- In addition, a warning icon next to the PCH Power Management check-box in the trace configuration editor window is shown to indicate this access problem

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 Intel® Direct Connect Interface (Intel® DCI) communication instability

- Current platforms based on the Intel processor code-named “Skylake” and A0, A1 and B0 stepping of the chipset code-named “Sunrise Point” suffer from a hardware instability for Intel DCI communication, which impacts the Intel® System Debugger – System Trace
- To fix Intel DCI communication instabilities please execute the following steps:
    - Close the System Trace Tool.
    - Power-off the target
    - Kill the Intel® DFx Abstraction Layer (Intel® DAL) master frame process
    - (MasterFrame.HostApplication.exe) using the Windows* task manager.
    - Unplug the Intel SVT CCA from the host.
    - Plugin the Intel SVT CCA adapter to the host again.
    - Start the System Trace Tool and follow the connect procedure including powering on the target

4.2.8 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. C:\Users\<name>\workspace) or ensure to use a different workspace together with the System Trace feature.

4.2.9 Timeout Messages on Intel® Silicon View Technology (Intel® SVT) CCA Firmware Update

- If the Intel(R) DFx Abstraction Layer library detects old firmware on the Intel SVT CCA, the firmware will be updated automatically during the first connect to the target using the system trace tool. In some cases it can be that the connect procedure runs into a timeout and shows error messages on the Eclipse* console. In this case please do the following:
    1. Close Eclipse*

2. Unplug Intel SVT CCA
3. Kill the Intel DAL master frame process
4. Plug-in the Intel SVT CCA
5. Start Eclipse* and follow the connect procedure described in the system trace user guide

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

4.2.11 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.

4.2.12 Intel(R) Dfx Abstraction Layer crashes during live streaming

- If the target layer crashes during live streaming, System Trace Tool will stay in an undefined state. Please restart System Trace Tool in this case, connect again and continue with live streaming

4.2.13 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

4.2.14 Trace Viewer may become unresponsive after workspace upgrade

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

```
16:07:36 [ERROR] Cannot send message, target connection server is not
running.
16:07:36 [ERROR] Server is unresponsive
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.

4.2.15 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.

4.2.16 CPU Probe Mode may not be available on the platform.

- Possible reason: No CPUs were detected in the tap chain (merge port not implemented?)
- Possible solutions:

    - Check the merged port implementation.
    - Manually configure PCH power management by setting bit 27 of offset 3Ch in the PM
    - MMIO BAR.
- Possible reasons:
    - The Privacy MSR is not set in IA FW (BIOS)
- Possible solutions:
    - Set the Privacy MSR in the BIOS setup
    - Unlock the platform
    - Manually configure PCH power management by setting bit 27 of offset 3Ch in the PM MMIO BAR.

4.3 Intel® System Debugger – Intel® Debug Extensions for WinDbg*

4.3.1 Known Limitations

- The following limitations apply only to the Intel® Debug Extensions for WinDbg* for IA JTAG debugging feature.
    - A data breakpoint can be set on the first thread only.
    - If connection to target fails, process “dllhost” is not killed automatically. The process must be killed manually to be able to try connection again.
    - Breakpoint skip count is not supported.
    - Execution HW breakpoints is not supported.

4.3.2 Error message “Unable to read debugger data block header” in WinDBG*

- The error message “Unable to read debugger data block header” indicates that on the target the kernel debugging is not activated.
  Activate kernel debugging by executing the command “bcdedit /debug on” on the target in a command prompt.

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)
- Support Intel® Xeon® Processor D-15xx (Grangeville)

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