Intel® SoC Watch for Google Chrome* and Linux* Release Notes

(for use under NDA only)

4th November 2019

Version 2020.1

Intel Corporation

www.intel.com

Legal Information

Intel Confidential

Contents

Legal Information

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

Version History

These are the main releases of Intel SoC Watch for Google Chrome* and Linux* OS

Date	Revision	Description
July 2015	2.0	Initial release for 2.0 Product
October 2015	2.1	Update to 2.1 Product release
January 2016	2.1.1	Update to 2.1.1 Product release
June 2016	2.2	Update to 2.2 Product release
October 2016	2.3	Update to 2.3 Product release
April 2017	2.3.1	Update to 2.3.1 Product release
November 2017	2.4	Update to 2.4 Product release. First release that aligns command line parameters and output formats across all supported operating systems.
November 2017	2.4.1	Update to 2.4.1 Product release. Includes bug fixes for PCIe reporting of GBE devices.
February, 2018	2.5	Added feature pch-ip-lat-limit and added Intel® VTune™ Amplifier import support for most PCH metrics.
April, 2018	2.6	Enhancements include initial support for Intel platform code named Ice Lake, added s0i3-sstate-dbg and lpss-ltr metrics, and improved PCIe LPM and PCH reporting.
May, 2018	2.6.1	Adds feature xhci-lpm. Also, adds a warning if too many telemetry-based metrics requested and corrects minimum LTR reporting for platform LTR.
August, 2018	2.7	Added average frequency report, new options (program-delay), new metrics (cpu-pkgc-cfg, fpga-pwr, fpga-temp), new group names, support for Intel platforms code named Amber Lake, fixed issues in ddr-bw, automation summary, multiple pkg handling, PCH and XHCI reporting among others.
November, 2018	2.8	Added new tables to cpu-pkgc-dbg. Fixed errors in pch-slps0 and hw-cpu-cstate PC10 reporting (requires kernel version 4.4 or newer).
January, 2019	2.9	Added tcss-state, trace file report grouping, and informational messages. Fixes for PCH metrics, xhci-lpm, and more.
March, 2019	2.10	Added support for Intel platform code named Comet Lake, added feature pmc-ip-pg-res for Intel Atom® platforms, added histogram for CPU frequency, added TypeC subsystem metric tcss-cfg-status, added bandwidth metrics for FPGA, and bug fixes.
April, 2019	2.10.1	This release corrects an error in a JSON-formatted help file.
June, 2019	2.11	Added support for CentOS/RedHat 7. Added option (-z) to automatically put the system into suspend state during a collection.

Date	Revision	Description
		Added support for Intel platform code named Tiger Lake and limited support for Intel platform code named Ice Lake-Xeon. Added reporting of power limits (PL1 and PL2) to feature -f pkg-pwr on Intel platform code named Apollo Lake. Includes support for Intel platform code named Cascade Lake-Xeon. Improves handling of unrecognized CPUs, reporting S-state when hibernation occurs, and other bug fixes.
September, 2019	2019.12	Added support for Intel PCH code named Comet Lake V. Added feature s0ix-subs-res. Improved sata-lpm DevSlp Capabilities reporting, cpu-pkgc-dbg, and PCH metrics for various platforms. Modified pch-ip-active percentage calculation. Modified hw-cpu-pstate reporting.
October, 2019	2019.13	Added feature s0ix-subs-status for Intel platform code named Tiger Lake. Fixed issue in ddr-bw and hw-cpu-cstate for Intel platform code named Tiger Lake. Fixed issue in hw-cpu-pstate for Intel platform code named Ice Lake.
November, 2019	2020.1	Added support for Intel platform code named Elkhart Lake and Intel PCH code named Mule Creek Canyon. Added support for Intel platform code named Tiger Lake-H. Fixed issues in pcie-lpm, pcie-ltr among others.

Customer Support

For technical support, including answers to questions not addressed in this product, see the Intel System Studio forum (https://software.intel.com/en-us/forums/intel-system-studio).By default, support is available through Intel Premier Support (premiersupport.intel.com) and Online Service Center (http://www.intel.com/supporttickets). Visit this page to find information about using Intel Premier Support.

Introduction 1

Intel® SoC Watch is a data collector for power-related data that can help identify issues on a platform that prevent entry to power-saving states. Captured metrics include:

- System sleep states
- CPU and GPU sleep states
- Processor frequencies
- Temperature data
- Device sleep states
- IO controller link states and latency reporting
- Platform Controller Hub activity

You can correlate the collected data and visualize over time using Intel®VTune Amplifier.

This document provides system requirements, installation instructions, issues and limitations, and legal information.

To learn more about this product, see:

- New features listed in the New in This Release section below, or in the help.
- Reference documentation listed in the Related Documentation section below
- Installation instructions can be found in the Installation Notes section below.
- For a detailed quick start guide to running the tool, see the *Intel SoC Watch User's Guide* in your installed documentation.

2 New in This Release

Release v2021.1 includes these changes:

- Added support for Intel platform code named Elkhart Lake and Intel PCH code named Mule Creek Canyon. Metrics that could not be verified for correctness have been omitted from this release (cpu-pkgc-dbg, pch-ip-active, pch-ip-status).
- Added support for Intel platform code named Tiger Lake-H. The PCH metrics for its chipset are not supported at this time.

System Requirements 3

Supported Architectures

Intel SoC Watch supports these Intel microarchitecture or platform code names:

- Anniedale
- Cherryview (Cherry Trail)
- Denverton
- Apollo Lake
- Gemini Lake
- Elkhart Lake
- Broadwell
- Skylake
- Kaby Lake
- Cannon Lake
- Coffee Lake
- Whiskey Lake
- Amber Lake
- Comet Lake
- Ice Lake
- Lakefield
- Skylake-Xeon
- Cascade Lake-Xeon
- Ice Lake-Xeon
- Tiger Lake

Supported FPGA Architectures

- Intel® Xeon® processor with integrated FPGA
- Intel® Programmable Acceleration Card with Intel® Arria® 10 GX FPGA (Intel® PAC with Intel® Arria® 10 GX FPGA)

Dependencies

Intel SoC Watch depends on specific OS configurations and hardware capabilities. If these are not present on the target system, Intel SoC Watch may fail to work properly.

- Linux Kernel version needs to be 2.6.32 or later.
- GNU C Library version must be GLIBC_2.17 or later.
- KERNEL_CONFIG_TRACEPOINTS must be enabled.
- Kernel should be compiled with "CONFIG_MODULES" enabled.
- P States
  - Kernel config CONFIG_X86_SFI_CPUFREQ or CONFIG_X86_ACPI_CPUFREQ must be enabled (i.e. set to 'y' or 'm').
  - One of these pstate drivers must be utilized: sfi-cpufreq, acpi-cpufreq, or intel_pstate. To determine which driver is loaded, check the sysfs /sys/devices/system/cpu/cpu0/cpufreq/scaling_driver file.
  - If one of these pstate drivers is not loaded, the kernel needs to be reconfigured and recompiled.
- C States
  - Kernel config CONFIG_TIMER_STATS must be enabled.
  - Kernel config CONFIG_INTEL_IDLE must be enabled and the intel_idle kernel module has to support the core of the target platform.

- To determine if the intel_idle kernel module is loaded, check the sysfs /sys/devices/system/cpu/cpuidle/current_driver file. It must equal intel_idle. If it equals acpi_idle, only C0 and C1 will be used by the core.

Where to Find the Release 4

Go to the Intel® System Studio website (https://software.intel.com/en-us/intel-system-studio) to get either an Evaluation (30-day trial release) license or a commercial license, and download the package from the Intel Registration Center (http://registrationcenter.intel.com/).

Installation Notes

5

Intel SoC Watch for Chrome* OS and Linux* OS is available as part of Intel System Studio. Use the steps below to install Intel SoC Watch on a target Chrome or Linux system.

Intel SoC Watch Prerequisites for Yocto and Wind River* Linux

The Intel SoC Watch binary is a C++ program that requires libstdc++ to be present on the target system in order to function. Although most Linux distributions provide this by default, there are some minimal distributions of Yocto and Wind River Linux that may not.

To check if your OS image contains libstdc++, run this command:

```
find / -name "libstdc++*"
```

If the library is present, you should see an output similar to:

```
/usr/lib64/libstdc++.so.6
```

If you see no output, the required library is missing. To fix this, do one of these steps:

- Rebuild Yocto or Wind River Linux with an option to include libstdc++.so file in the image.
- Provide the library file directly to Intel SoC Watch.

Rebuild OS image to include libstdc++.so file

1. Open the projectDir/local.conf file in a text editor.
2. Add the library with the IMAGE_INSTALL_append option and save the file:

```
IMAGE_INSTALL_append = " libstdc++"
```

NOTE Make sure that you include the leading space as it will be concatenated with any libraries previously added in a list.

3. Rebuild the platform project by running the following command from the projectDir:

```
make
```

Once the build completes, the library will be part of the project image rootfs. Proceed with reflashing the OS image to the target system and subsequently installing Intel SoC Watch.

Provide libstdc++.so library file directly to Intel SoC Watch

If you are unable to rebuild the OS image, but are able to obtain a valid libstdc++.so file (you can copy it from another system or build it yourself):

1. Unpack the SoC Watch package.
2. Copy the file into the /libs subfolder where it will be picked up at runtime.

libgcc_s.so Library

Some minimal Yocto distributions may also lack another library: libgcc_s.so. In this case, follow either of the aforementioned solutions.

Build the OS and include the image:

Use this command when building the OS.

```
IMAGE_INSTALL_append = " libgcc_s"
```

Copy file into Intel SoC Watch package:

Copy the library libgcc_s.so file into the /libs folder of the Intel SoC Watch package.

Extracting the Intel SoC Watch Package

You must extract the Intel SoC Watch package to the system containing the kernel of the target device.

Use the `find . -name Module.symvers` command on the device containing the target system's kernel to determine the kernel build directory.

Build the Kernel Modules

If the Intel SoC Watch kernel modules (i.e. device drivers) are not present in the OS image of the target system, you will need to build and possibly sign them. Building and signing device drivers requires access to the kernel build directory for the OS image running on your target device. A kernel build directory is generated while building the OS image of the target system.

When building the kernel modules, do not open (or unzip) the Intel SoC Watch package (i.e. tar.gz file) on a Windows* based system and copy a Linux system. Unzip the package on the Linux build system using the unzip command to make sure the build scripts and make files are unmodified.

If a kernel is built with the CONFIG_MODULE_SIG kernel config enabled, any device driver loaded into that kernel must be signed with the same keys used to build the kernel. In general, drivers built for Linux targets do not need to be signed and the following description assumes the drivers do not need to be signed. But, if an end user tries to load an unsigned driver into a kernel that requires signed drivers, the `insmod` command will fail with the error Required key not available. If a signed driver is loaded into a kernel that does not require signed drivers, the load will succeed.

The build_drivers script is provided to simplify building all of the drivers. The script supports multiple switches including;

`-n` // do not build the socperf driver; used for Intel® Core™ processor based systems

`-l` // build the kernel for a Linux target

`-s <full path to sign-file>` // signs the drivers; the path is normally .../kernel/*/scripts/sign-file

`--hypervisor acrn` // builds the socwatchhv2_0.ko driver

Building Linux* Kernel Modules

Linux kernel modules may only be built after the Intel SoC Watch package and kernel headers are copied to and installed on the target. See the section Intel SoC Watch for Linux Installation below for instructions on how to build the kernel modules for a target device running Linux.

Building the Chrome Kernel Modules

1. Copy the `socwatch_chrome_linux_NDA_`*`<version>`*`.tar.gz` file to the chroot directory on your host system used to build the Chromium* image.
2. Extract the contents with the command:

```
tar xvzf socwatch_chrome_linux_NDA_<version>.tar.gz
```

   The `socwatch_ chrome_linux_NDA_`*`<version>`* directory is created. In the following description, the `<kernel-build-dir>` can be determined by finding the directory that contains the Module.symvers file. Use the `find . -name Module.symvers` command to determine the kernel build directory.
3. Use one of the following commands to build the appropriate files:

If the target system has an Intel Atom® processor, use the following command within the chroot environment to build the `socwatch2_10.ko` and `socperf3.ko` files.

```
sh ./build_drivers.sh -l -k <kernel-build-dir>
```

If the target system has an Intel® Core processor, use the following command within the chroot environment to build the `socwatch2_10.ko` file.

```
sh ./build_drivers.sh -n -l -k <kernel-build-dir>
```

4. Navigate to the `socwatch_chrome_linux_NDA_<version>` directory.

   Execute `sh ./socwatch_chrome_create_install_package` to create the `socwatch_chrome_CUSTOM.tar.gz` file.

Install Intel SoC Watch

Host: laptop, desktop, or server used to communicate with target device.

Target: device to be analyzed with Intel SoC Watch.

Intel SoC Watch for Linux* Installation

If Intel SoC Watch was previously installed on the target, delete the `socwatch_linux_*` directory before installing a new version. Then, perform the following steps on the target device.

1. Login to the target as root:

```
ssh root@<your_target_IP>
```

2. Extract the Intel SoC Watch package to the target Linux system. The `<extract_dir>/socwatch[_hypervisor]_linux_v<version>_x<architecture>` directory will be created.
3. Copy the Intel SoC Watch package to a working directory:

```
mkdir -p /home/socwatch
```

```
cp <extract_dir>/socwatch[_hypervisor]_linux_v<version>_x<architecture> /home/socwatch/.
```

4. Navigate to the Intel SoC Watch directory:

```
cd /home/socwatch/socwatch_linux_v<version>_x<architecture>
```

5. Use one of the following commands:

   a. If the target system is running the ACRN* hypervisor, use the following command to build the `socwatchhv2_0.ko` and `socwatch2_10.ko` files.

```
sh ./build_drivers.sh -l -k <kernel-build-dir> --hypervisor acrn -s <full-path-to-sign-file>
```

   b. If the target system has an Intel Atom® processor, use the following command to build the `socwatch2_10.ko` and `socperf3.ko` files.

```
sh ./build_drivers.sh -l -k <kernel-build-dir> -s <full-path-to-sign-file>
```

   c. If the target system has an Intel® Core™ processor, use the following command to build the `socwatch2_10.ko` file.

```
sh ./build_drivers.sh -l -k <kernel-build-dir> -n -s <full-path-to-sign-file>
```

Intel SoC Watch Chrome Installation

If the drivers are not installed or if the Intel SoC Watch executable requires a more recent version of the drivers, you will need to compile the drivers from source. See the Building the Chrome Kernel Modules section.

If Intel SoC Watch was previously installed on the target, delete the `socwatch_chrome_*` directory before installing a new version.

Summary: Three requirements are mandatory to install Intel SoC Watch On Chromium

1. Root access.
2. The system’s home partition must allow exec (allow the execution of any binary on this file system partition).
3. The kernel must allow kernel modules to be loaded – i.e. module_locking must be disabled.

The following steps are one way to meet these requirements.

1. Target: After your computer has booted to the Chromium OS login screen, press `[ Ctrl ]` `[ Alt ]` `[ F2 ]` to get a text-based login prompt. (`[ F2 ]` may appear as `[ → ]` on your Notebook keyboard.)
2. Target: login as **chronos**.
3. Target: `sudo -i`
4. Target: `chromeos-setdevpasswd` // set password
5. Target: `exit`
6. Target: `exit`
7. Target: login as chronos (with your new password)
8. Target: `sudo -i`
9. Target: `mount -o remount,rw rootfs /`
10. Target: `passwd root` // set password
11. Target: `mkdir /home/socwatch`
12. Target: `ifconfig` // determine the IP address assigned to your target
13. Target: `cd /`
14. Target: `find . -name module_locking`
15. Target: `sudo echo 0 > </path_to_module_locking_file found in previous step>` // if module_locking is set to 1, this step is required to disable module_locking
16. Target: `sudo mount -o remount,rw /home` // change read only bit to read write on /home partition
17. Target: `sudo mount -o remount,exec /home` // change noexec bit to exec on /home partition
18. Host: If the drivers are not preinstalled and are built using the instructions in the Building the Chrome Kernel Modules section, `scp socwatch_chrome_CUSTOM.tar.gz root@<target_IP>:/home/socwatch/` or `scp socwatch_chrome_linux_NDA_<version>.tar.gz root@<target_IP>:/home/socwatch/` // use root password set in step 10 and IP address determined in step 12
19. The `socwatch_chrome_*.tar.gz` file can also be copied to the target system's /home/socwatch directory using a USB drive. Step 20 can be executed directly on the target if easier.
20. Host: `ssh root@<your_target_IP>` // use root password set in step 10 and IP address determined in step 12
21. Host: Untar the file copied to the target in step 18. `tar -zxf <filename>.tar.gz`

Prerequisites for FPGA metric collection using Intel SoC Watch

Intel SoC Watch leverages Open Programmable Acceleration Engine (OPAE) drivers to collect data on supported FPGA platforms. OPAE drivers must be loaded on the system containing one of the supported FPGA architectures (see System Requirements) prior to running Intel SoC Watch collections. Source code for OPAE drivers may be downloaded from the OPAE GIT repository located at the following link: https://github.com/OPAE/opae-sdk/releases

The OPAE installation guide provides more information on the installation requirements for OPAE drivers. The guide may be downloaded from the following location:https://opae.github.io/latest/docs/install_guide/installation_guide.html

Follow these instructions to build the OPAE drivers for your system:

1. Download the kernel headers to your system.
2. Download and extract the file starting with `opae-intel-fpga-driver*.tar.gz` to your system.
3. Change to the extracted directory, locate the file named `Makefile`, and run the `make` command in that directory.

This process will generate several files with a `.ko` extension. These are the OPAE drivers that need to be loaded to enable Intel SoC Watch collections. Load these driver files using the following commands:

```
sudo insmod fpga-mgr-mod.ko
sudo insmod intel-fpga-pci.ko
sudo insmod intel-fpga-fme.ko
sudo insmod intel-fpga-afu.ko
```

Fixed Issues 6

Release v2020.1 has a fix for these issues.

- Fixed an issue that was preventing drivers from being built against ChromeOS sources.
- Feature -f pcie-ltr now reports "N/A" in its Capabilities summary report when the device's capability information is not available. Previously, the summary reported "No" when a device had entered D3 Cold and its capability data was not available.
- Fixed regression in feature -f pcie-lpm which caused incorrect reporting for residency in L1 substates for NVME devices. The regression occurred in v2.11, and manifested as always reporting 100% time in state L0s for NVME devices.
- Fixed an issue where SoC Watch did not handle both module and die topologies on Intel platform code named Apollo Lake running Linux kernel 5.3 .
- Fixed an issue where "-f lpss-ltr" was not working on some platforms.
- Fixed a regression introduced in v2019.13 where feature "-f s0ix-subs-res" was not available for Intel platform code named Lakefield.
- Fixed an issue affecting Intel server platform code named Ice Lake-Xeon where C-states shown for Package and Core did not match the supported states. Core C1 and Package C7 were added, Core C7 was removed.
- Fixed an issue affecting Intel platform code named Lakefield where feature "-f pch-ip-active" was using an incorrect base address for the underlying energy reporting counters. The firmware change that requires the base address to change has not been released yet.
- Fixed an issue affecting Intel platform code named Lakefield where feature "-f cpu-pkgc-dbg" was missing tables.
- Fixed issue resulting in some platforms being reported as Intel platform code name Amber Lake that should have been Intel platform code name Kaby Lake. The data reported was correct, only the code name was incorrect.

Known Issues 7

Bandwidth (on Intel Core Platforms)

- Memory bandwidth metric (-f ddr-bw) is not currently supported on Intel platforms code named Tiger Lake.
- GT DDR BW is incorrectly reported as 0 on Intel platforms code named Ice Lake on older steppings. The bandwidth counter for Ice Lake is currently attributing GT memory traffic to IA, thereby over-counting IA and leaving GT as 0 Mbytes/sec. The counter problem is fixed in C-step.
- The presence of EDRAM on a system may not be detected by Intel SoC Watch. This is known to occur when the accelerator card VCA2, which contains EDRAM, is present.
- Total DDR bandwidth does not include EDRAM. On systems using EDRAM, the `ddr-bw` feature report may have a discrepancy between the total data reads and writes and the total component requests. The Data Reads+Data Writes will be significantly higher than the total IA+GT+IO requests, because the EDRAM requests are not included.

Bandwidth and DRAM Self Refresh (on Intel Atom® Platforms)

- On Intel platforms code named Gemini Lake, memory bandwidth and memory self-refresh metrics not available. The following features are not supported: `ddr-bw, cpu-ddr-bw, cpu-ddr-mod0-bw, cpu-ddr-mod1-bw, disp-ddr-bw, isp-ddr-bw, gfx-ddr-bw, io-bw, all-approx-bw, dram-srr`. Use `-f dram-bw` as an alternative for DDR bandwidth. There is no alternative for memory self-refresh.
- Intel SoC Watch v2.10.0 and later versions require loading the socperf v3.0 driver to measure bandwidths or DRAM self-refresh on Intel platforms code named Cherry View, Broxton, and Apollo Lake. This version can automatically detect if the system is configured to support use of the hardware signals required to collect these metrics (`ddr-bw, gfx-ddr-bw, cpu-ddr-mod0-bw, cpu-ddr-mod1-bw, disp-ddr-bw, isp-ddr-bw, all-approx-bw, io-bw` and `dram-srr`). In older releases, these metrics were disabled completely on Apollo Lake platforms to avoid a system crash. If the system does not support these metrics, use `dram-bw` as an alternative for collecting bandwidth. Use the –v option to determine which version of the socperf driver is loaded. If a mismatch occurs (socperf v1.2.0 used with socwatch >=v2.6.1), Intel SoC Watch will report this error: *-1, SOCPERF ERROR configuring SOCPERF interface.*
- If Intel SoC Watch crashes while collecting a bandwidth feature (e.g. –f ddr-bw) or the DRAM self-refresh feature (i.e. –f dram-srr) AND a subsequent collection prints the error: `ERROR: ERROR configuring SOCPERF interface!` then both the socperf3.ko and socwatch2_10.ko kernel modules must be unloaded with the rmmod command and reloaded with the insmod command before Intel SoC Watch can be used to collect additional data.
- When measuring DRAM self refresh using the `-f dram-srr` feature on cost reduced systems (e.g. Intel platforms code named CherryTrail cost reduced), Intel SoC Watch may report 100% self refresh residency on Channel 1. These systems are single channel systems and therefore, the result should be 0%.
- On a very small number of systems, results from the all-approx-bw feature may be one half of the correct result. During testing, this issue was only experienced on the first collection after the system was booted. All subsequent collections correctly measured the systems bandwidth as expected.
- If socperf reads occur before the start of collection, a dmesg error message is generated: "socperf3: [ERROR] ERROR: RETURNING EARLY from Read_Data". This message is benign and can safely be ignored.

PCH Active

- The new table, PCH Active State Summary, reported by -f pch-ip-active is described as using collection duration time to calculate the Active residency percentage. To be more precise, it is using the collection duration based on the fist and last samples collected for this particular metric, which may be slightly different from collection duration reported at the top of the summary.
- PCH Active data collection rate may be too high which can prevent entry to lower power states. The recommended collection rate for `-f pch-ip-active` data is 5 second intervals, but currently a single interval is used for all metrics and 5 seconds is too long for metrics that are derived from sampling status data. A fix is in progress. You can work around this issue by using the `-n` option to increase the interval to 5 seconds and limiting the metrics collected to those that come from hardware accumulators since sampling frequency has no impact on accuracy of results in that case. The data source information is found in the Intel SoC Watch User's Guide "Options Quick Reference" section.
- Issues on Intel platform controller hub (PCH) code named Tiger Lake PCH-LP:
  - Feature -f pch-ip-active report for Physical Mode Lane function and rate assignment is invalid for Lanes 12-15. Only twelve lanes (0-11) are used on this platform, but 16 lanes are shown. The mapping information is not set for lanes 12-15. If the values happen to be 0, these lanes appear mapped to function PCIe/DMI and rate PCIe G1.
- Issues on Intel platform controller hub (PCH) code named Lakefield PCH-B:
  - Feature -f pch-slps0 is not available on Intel platforms code named Lakefield. The hardware support was not enabled on the platform, the workaround is to sum the S0ix.y substate residency values reported by -f s0ix-subs-res.
- Issues on Intel platform controller hub (PCH) code named Ice Lake PCH-LP:
  - Feature -f pch-ip-active report for Physical Mode Lane function and rate assignment is incorrect, showing all lanes assigned to function PCIe/DMI and rate PCIe G1. The hardware register containing that information is not being set resulting in all 0's, which map to those values.
  - Feature -f pch-ip-active reporting is incorrect for Ungated ISH SRAM. The status is not being set making it appear to be always on, never power-gated (value reported is always 20).
- Issues on Intel platform controller hub (PCH) code named Cannon Lake PCH-LP.
  - PCH PMC active residency may be incorrect when certain platform debug configurations are used that cause the PMC counter to never stop incrementing. The PCH SLP_S0 time (`-f pch-slps0`) + PCH PMC active time (-f pch-ip-active) will be greater than the collection time when this is occurring.
- Issues on Intel platforms using the Intel platform controller hub (PCH) code named Cannon Lake PCH-H:
  - PCH PMC active residency may be incorrect when certain platform debug configurations are used that cause the PMC counter to never stop incrementing. The PCH SLP_S0 time (`-f pch-slps0`) + PCH PMC active time (-f pch-ip-active) will be greater than the collection time when this is occurring.
  - PCH Slp-S0 residency reporting may be in error. If a certain throttling is enabled, the SLP-S0 residency reported by `pch-slps0` feature is ~60x higher than actual (or always 0 if using Intel SoC Watch v1.19.1). A workaround in firmware is being implemented.
  - PCH Active residency reporting may be all zero. The counters for PCH activity may be turned off by firmware. In this case, they will not increment resulting in all-zero reports for `-f pch-ip-active` metrics. The workaround requires use of Intel® Flash Image Tool (Intel® FIT) to enable the thermal power reporting feature. Using Intel FIT, select Power, then PCH Thermal Reporting, and set Thermal Power Reporting Enabled parameter value to Yes. A change is in progress to make the default setting enabled.
  - Feature -f pch-ip-active reporting of Physical Mode Lane function and rate may be incorrect and is under investigation. There are many Unknown values which may be due to incorrect setting of the register or incomplete mapping.

PCIe, XHCI, SATA, LPSS

- Root Port vs CLKREQ Mapping not available on some Intel platforms code named Coffee Lake. The `-f pcie-lpm` feature's Root Port vs CLKREQ Mapping information has been hidden by hardware on some platforms. In this case, all values read are invalid so the following message is reported: Root Port vs CLKREQ Mapping is not available on this platform (0xFFFFFFFF values read by all the counters). There is no workaround for this issue.
- CLKREQ Mapping does not recognize disabled port. The `-f pcie-lpm` feature's Root Port vs CLKREQ Mapping table does not recognize disabled ports resulting in CLKREQ # 0 being shown for disabled ports.
- It has been observed that SATA does not enter DevSleep when the in-box ACHI driver of the window is configured for HIPM only. If Intel® Rapid Storage Technology (Intel® RST) driver is used, which is configured to use HIPM+DIPM, entry to DevSleep does occur. If ACHI driver is configured for HIPM+DIPM, entry to DevSleep can be achieved.
- GbE device controller link state names are shown as PCIe link states in the PCIe LPM report for -f pcie-lpm, which includes GbE device controllers. The GbE link state K0 is listed as L0, and K1 as L1 in that report.
- XHCI LPM may report no device attached error when one is attached. Feature `-f xhci-lpm` will report a "no device attached" error message when the actual problem is that either the XHCI host controller or XHCI device driver does not provide the information needed to discover the device.

C-States / P-States

- The hardware CPU P-state data may be missing for some Cores when using feature `-f hw-cpu-pstate` on Intel platforms code named Skylake, Kaby Lake, Whiskey Lake, Comet Lake, and Amber Lake. The issue is caused by unexpected behavior of the hardware counters. The tool ignores these bad samples which results in the missing data.
- On Intel platforms code named Lakefield, the big core counter for Core C6 residency reports only 0 and requires microcode patch to be functional.
- Feature `cpu-pkgc-dbg` report for Transient/Uncommon Reasons can show an incorrect value for 'Max sampling interval'. It displays the duration since the last transient/uncommon reason was detected; however, data still gets sampled at the correct interval.
- On Intel Atom® platforms, if all cores in a module request C6FS but actual sleep time is short, the Auto-Demotion logic of the hardware resolves the module state to module C0. Consequently, you may find module C0 to be greater than the sum of core C0 and C1 on all the cores in a module. On the same lines (auto demotion at the package level), the package C0 may be greater than module C0 residencies of the two modules.
- During the transition time from core C1/C1e/C6 to core C0, a core may run in LFM which will be properly measured by Intel SoC Watch. Therefore, results that include a large number of C1/C1e/C6 residencies may show a lower PState than expected.
- The `gfx-cstate` metric is obtained by frequently polling GPU counters that provide the graphics c-state residencies. On some Intel Atom® platforms, we have noticed that for 1 out of every ~3000 samples the residency of one of the c-states (Render C0, C1, C6 or Media C0, C1, C6) as obtained from the GPU counters is greater than the sample duration by 5% or more. When this happens Intel SoC Watch discards that sample and throws the error, "ERROR: Residency counter for GPU C-state = xxxx TSC ticks, but actual sample duration = yyy TSC ticks. Difference is more than 5.000000 percent." The error by itself is non-fatal and Intel SoC Watch results give a good idea about the gfx-cstate residencies since the bad sample is collected only 1 out of ~3000 samples.
- In order to visualize graphics C-states that are reported as Render and Media, the table headers in the trace file (generated with option -r int), must be manually modified, adding *Render* and *Media* to the appropriate C0, C1, and C6 column headers.

S States & D States

- S0i3 State Residency may incorrectly report 0 for s0i3-total. The `-f s0i3-sstate` feature will report 0% for s0i3-total if the time in connected standby (s0i3) exceeds the collection duration.

- When the `-f sstate` switch is specified, Intel SoC Watch generates a summary table that describes both the system's ACPI S3 and S0ix behavior. The table may show the S0 column first or the S3 column first.
- On Intel platform code named CherryView based devices:
  - Even when the device's screen is off, the NC DState called Display DPIO is reported in the D0i0 state 100% of the time. This result may or may not be correct.
  - When collecting NC D0ix states with the `-f nc-dstate` switch, note that the Display Island B (HDMI) IP block will remain in D0i0 when the primary display is enabled even if an HDMI cable is removed.
  - When using the `sc-dstate` feature, the SEC IP block results are incorrect and should be ignored. Also, the UFS IP block results are incorrect because an internal fuse is disabled.
  - Users should not try to correlate S0ix or screen state (on | off) with NC GFX IP block states. Instead, confirm the NC Gfx IP block results by correlating with the gfx-pstate results.

Miscellaneous

- Feature -f dram-pwr is not supported by all versions of the server Intel platforms code named Skylake-Xeon, Cascade Lake-Xeon, and Denverton). The report contains all zero values in this case.
- Metrics report Unknown 0 when `-m` is not used and hibernation occurs. Metrics with a snapshot default collection mode, such as CPU C-state, will show the Unknown state with 0 time and the remaining states will not sum to the total collection duration if the system entered hibernation during the collection and the `-m` option was not specified. The snapshot metrics are only collected at the start and end of a collection by default, but finding hibernation time requires samples taken throughout the collection. Including `-m` will cause continuous sampling to occur for all metrics. When hibernation occurs, a message reporting time spent in hibernation appears at the beginning of the summary report. The Unknown state is then included for all appropriate metrics and the time in hibernation is included in that state. Refer to the *Intel SoC Watch User's Guide* "Options Quick Reference" section to learn which metrics have a snapshot collection mode by default.
- Intel SoC Watch reads PMIC and Skin Temperatures from the system's sysfs. Rarely, a sysfs read may not return before a subsequent sysfs read occurs. When this occurs, specific sample results may be missing in the timed trace CSV and raw text files.
- Permission issues with SELinux will cause Intel SoC Watch collection to fail. Some distributions enable SELinux by default. If you have the following file your system may have SELinux enabled:

```
/selinux/enforce
```

  If that file exists, you can disable by issuing:

```
'echo 0 > /selinux/enforce
```

- Syntax errors in the command line may not report a visible error message. If a collection did not run and you are not seeing any error message, add option `-d 2` to your command line to get more information.

Intel® VTune™ Amplifier Visualization

- Collections containing PCIe metrics (pcie-ltr, pcie-lpm) cannot be imported to Intel VTune Amplifier. A corrupted .pwr file message will result if an import is attempted. The issue is under investigation.
- Intel VTune Amplifier 2017 for Systems Update 1 or later is required for visualizing and analyzing Intel SoC Watch v2.10.0 and newer PWR files. We recommend using the latest version of Intel VTune Amplifier.
- If the bandwidth is 0 Mb throughout the collection for a particular bandwidth type, Intel VTune Amplifier will not show a timeline entry for it. The timeline is shown only if there is at least one non-zero value.
- In some cases, the summary CSV results produced by Intel SoC Watch can vary from the summary results shown by Intel VTune Amplifier even though they represent the same collection. For example, the summary CSV file may report a specific cpu-pstate residency of 50.78% and Intel VTune Amplifier may report the same cpu-pstate residency as 50.8%.
- Intel VTune Amplifier currently does not support bandwidth ranges used for ReadPartial and WritePartial. In order to keep the visualization consistent with Intel SoC Watch v1.x, Intel VTune Amplifier uses the upper bound of the range to visualize the bandwidth.

- The minimum and average calculations displayed in the grid for Sampled Value metrics don't take 0 values into consideration in older versions of Intel VTune Amplifier. For example, Sampled Graphics P-States minimum values may show a value higher than 0 Mhz even when some samples have 0 Mhz values. This in turn affects the average value calculation.
- Some metrics are not supported for visualization in Intel VTune Amplifier. When `-r vtune` is specified, unsupported metrics will be excluded from the .pwr file. When this occurs, a message is printed to the console for each metric that was collected but not included in the .pwr file. The unsupported metric is `cpu-pkgc-dbg`. If there were no supported metrics in the collection, no .pwr file is created.

Related Documentation

8

The below documents are available with this release.

- Intel® SoC Watch for Google Android* OS, Google Chrome* OS, and Linux* OS User's Guide.
- Energy Analysis help (https://software.intel.com/en-us/energy-analysis-user-guide)