Intel® SoC Watch for Google Android* OS Release Notes

27 March 2019

Version 2.10

Intel Corporation

www.intel.com

Legal Information

Contents

Legal Information

Version History

These are the main releases of Intel SoC Watch for Google Android* OS:

Date	Revision	Description
July 2015	2.0	Initial release for 2.0 Product
October 2015	2.1	Update to 2.1 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.
February, 2018	2.5	(External) Includes support for Intel platform code named Gemini Lake and other fixes.
May, 2018	2.6.1	Enhancements include new hw-cpu-hwp metric, --log option, and improved support in gfx metrics.
August, 2018	2.7	Added average frequency report, new options (program-delay, disable-alts), new metric (pkg-pwr), new group names, fixed issues in ddr-bw, automation summary, and multiple pkg handling.
November, 2018	2.8	Fixed errors in hw-cpu-cstate PC10 reporting (requires kernel version 4.4 or newer).
January, 2019	2.9	Added trace file report grouping and informational messages.
March, 2019	2.10	Added histogram for CPU frequency and bug fixes.

Customer Support

For technical support, including answers to questions not addressed in this product, visit the technical support forum, FAQs, and other support information at: Intel System Studio forum (https://software.intel.com/en-us/forums/intel-system-studio) discuss issues with your peers, ask questions of the development team, etc.

To submit an issue, go to Intel® Premier Support: (https://employeeportal.intel.com/irj/portal/IntelPremierSupportUser)

Please remember to register your product at https://registrationcenter.intel.com/ by providing your email address. Registration entitles you to free technical support, product updates and upgrades for the duration of the support term. It also helps Intel recognize you as a valued customer in the support forum.

1 Introduction

Intel® SoC Watch is the data collector for power-related data that helps identify issues on a platform that are preventing entry to power-saving states. The metrics it captures include system sleep states, CPU and GPU sleep states, processor frequencies, temperature data, device sleep states among others. The collected data can be correlated and visualized over time using the 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 v2.10 includes the following changes compared to v2.9:

- Enhanced -f hw-cpu-pstate feature reporting to include *Core P-state/Frequency Histogram* summary making it easier to see residencies in turbo and throttled frequency states.
- This release updates socperf from v2 to v3. It is no longer compatible with the older socperf2 driver.
- This release (v2.10) is the last containing support for 32-bit Linux OS and Android OS.

3 System Requirements

Supported Architectures

Intel SoC Watch for Google Android* OS supports the following Intel microarchitecture or platform code names:

- Apollo Lake (Broxton-P)
- Gemini Lake
- Denverton
- Cherryview
- Anniedale
- Skylake
- Kaby Lake
- Coffee Lake
- Broadwell
- Haswell

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
- GNU C Library must by version GLIBC_2.17 or later
- KERNEL_CONFIG_TRACEPOINTS must be enabled
- Kernel must have been 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 the following 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 target platform's core.
  - 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.

4 Where to Find the Release

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/).

5 Installation Notes

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

Extracting the Intel SoC Watch package

Users will need to extract the Intel SoC Watch package to the system containing the target device's kernel (this may be the host system if the target device is running an Android kernel, or the target system if the device is running a Linux kernel).

By default the package can be found in the following locations:

- On Linux systems: `/opt/intel/system_studio_<version>/energy_profiler_and_socwatch/socwatch_for_target`
- On Windows* systems: `C:\Program Files (x86)\IntelSWTools\energy_profiler_and_socwatch\socwatch_for_target`

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 target system's OS image, 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 target system's OS image.

When building the kernel modules, the Intel SoC Watch package (i.e. ZIP file) should not be opened (i.e. unzipped) on a Windows* based system and then copied to a Linux system. The package should be unzipped 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

Building Android* Kernel Modules

1. Copy the Intel SoC Watch package to the host system used to build the target system's Android kernel.
2. Extract the contents. The *<extract_dir>*`/system_studio_target/socwatch_android_v`*<version>* directory will be created.
3. Use one of the following commands to build the appropriate files:

a. If the target system is based on an Intel® Core™ processor, use the following command to build the `socwatch2_10.ko` file.

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

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

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

NOTE
If the build fails because the `asm/intel_mid_pcihelpers.h` file is missing, remove the following line from the `soc_perf_driver/src/Makefile`

```
EXTRA_CFLAGS += -DCPI_HELPERS_API
```

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 Android* Installation

Make sure the host is connected to the target via adb before running the install script.

1. After extracting the Intel SoC Watch package on your host system, run the `socwatch_android_install.sh` script on a Linux host or from a Cygwin window on a Windows host. Run `socwatch_android_install.bat` from a Windows host.

```
socwatch_android_install.sh or socwatch_android_install.bat
```

2. The script installs the Intel SoC Watch executables to the `/data/socwatch` directory on the target by default. Use the `-d` option to select a different install directory and the `-s` option to define a specific Android* device if multiple devices are connected to the host via `adb`.
3. Using the `adb` command, start a shell with root privileges.

```
adb root
```

```
adb shell
```

4. Navigate to the directory containing the drivers on the target system. If the drivers are preinstalled, they are located in the `/lib/modules` or `/system/lib/modules` directory. If the drivers are built on the host system, they should be copied to a directory on the target system.
5. Load the drivers. Note that the `socperf` driver must be loaded before the `socwatch` driver on a system with an Intel Atom processor.

```
insmod socperf3.ko
```

```
insmod socwatch2_10.ko
```

6. Confirm the drivers are loaded.

```
> lsmod
```

NOTE

- Make sure to use the latest version of the `socperf` driver.
- Previous versions of the `socwatch2_x.ko` driver (e.g. `socwatch2_0.ko`) will work, but new collector support and/or bug fixes may be missing in the older drivers. If an older `socwatch2_x.ko` driver is used, some metrics may not be collected.

Key Files

The following table describes the key files.

File	Description
`build_drivers.sh`	The build script used to build all of the device drivers utilized by Intel SoC Watch.
`socperf3.ko`	The socperf kernel module used to measure bandwidth and DRAM self refresh on systems with an Intel Atom processor.
`socwatch2_x.ko`	The Intel SoC Watch kernel module used to collect both hardware and kernel data at runtime.
`setup_socwatch_env.sh`	The script used to setup the Intel SoC Watch runtime environment.
`socwatch`	The Intel SoC Watch executable built as a native application. Use this file to collect data and generate additional results from a raw SW2 file.
`SOCWatchConfig.txt`	The Intel SoC Watch configuration file. The configuration file is read by Intel SoC Watch immediately before each collection. It contains hardware addresses utilized by the device driver during the collection.
`EULA.txt`	End User License Agreement file.
`third-party-programs.txt`	List of third party programs included in the package.
`plugins/libSWCore.so`	A library providing Intel SoC Watch functionality.

Remove the Intel SoC Watch Drivers

After using Intel SoC Watch, remove the drivers using the `rmmod` command (e.g. `rmmod socwatch2_10`). The `socwatch` driver must be unloaded before the `socperf` driver is unloaded.

6 Fixed Issues

Release v2.10 has a fix for the below issues.

- The logical core count displayed in Intel Vtune™ Amplifier summary page for an imported SoC Watch collection is now correct.
- On Intel platforms code named Apollo Lake, a check has been added to determine if the North Peak channel is disabled which prevents collecting bandwidth data per IP. If it is not disabled, the individual bandwidth metrics can be used.

7 Known Issues

Intel® VTune™ Amplifier Visualization

- Intel VTune Amplifier 2017 for Systems Update 1 or later is required for visualizing and analyzing Intel SoC Watch v2.10.0 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.
- Sometimes the summary CSV results produced by Intel SoC Watch do not match exactly 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.

Bandwidth and DRAM Self Refresh

- Intel SoC Watch v2.10.0 requires loading the socperf v3.0 driver to measure bandwidths or DRAM self-refresh on Intel platforms code named Cherry View, Broxton, and Apollo Lake. Use the –v switch 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 *-1, SOCPERF ERROR configuring SOCPERF interface....*
- 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 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 BayTrail cost reduced or 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%.
- Only one bandwidth (`ddr-bw, cpu-ddr-mod0-bw, cpu-ddr-mod1-bw, io-bw, disp-ddr-bw, gfx-ddr-bw, isp-ddr-bw, all-approx-bw`) or DRAM self refresh (`dram-srr, dram-srr-ch0, dram-srr-ch1`) can be measured during a collection due to hardware limitations. For example, the following Intel SoC Watch command line will fail. `./socwatch -f ddr-bw -f cpu-ddr-mod0-bw ...`

Miscellaneous

- In order to graph graphics cstates using the filename_trace.csv file generated with the -r int switch, the table headers should be manually modified. Render and Media should be added to the C0, C1, and C6 cells as appropriate in order to properly graph the results.

- The `gfx-cstate` metric is obtained by frequently polling GPU counters that provide the graphics c-state residencies. On a few devices, 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. This issue is being investigated further.
- 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.
- 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.
- Metrics such as CPU C-state, that report state residency that comes from hardware accumulators 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. 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. In order to find the hibernation time, data must be sampled throughout the collection. Data that comes from hardware accumulators and noted as Snapshot collection mode (in the Intel SoC Watch User's Guide "Options Quick Reference" section) are only collected at the start and end of collection unless the max-detail option (`-m`) is specified.
- Total DDR bandwidth does not include EDRAM. On systems using EDRAM, the `ddr-bw` feature report may have a discrepancy between the total data read 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. There is no software access to a counter for the traffic between EDRAM and DDR at this time.
- 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
```

C-States / P-States

- If all cores in a module request C6FS but actual sleep time is short, hardware's Auto-Demotion logic 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.
- On Intel platforms code named Broxton and Apollo Lake, the `cpu-cstate` metric results do not contain module C-state information.

S States & D States

- 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 on Intel platforms code named CherryTrail or Braswell based systems, the SEC IP block results are incorrect and should be ignored. This issue is under investigation. Also, the UFS IP block results are incorrect because an internal fuse is disabled.

8 Related Documentation

The below documents are available with this release.

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