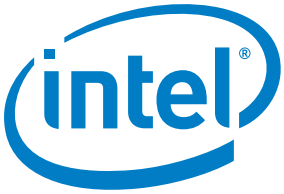

Intel® SoC Watch for Google Chrome* and Linux* OS 2.6.1

Release Notes –NDA Version Only

03 May 2018

Intel Confidential

Version History

These are the main releases of Intel® SoC Watch for Google Chrome* and Linux* OS 2.X

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 fix 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 new lpss-ltr and s0i3-sstate-dbg metrics, and improved PCIe LPM and PCH reporting.
May, 2018	2.6.1	Adds feature xhci-lpm. Also adds warning if too many telemetry-based metrics requested, corrects minimum LTR reporting for platform LTR.

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.

Contents:

1 Introduction

Intel® SoC Watch is the data collector for power-related data to debug blockers to low power sleep states. The metrics it captures include system sleep states, CPU and GPU sleep states, processor frequencies, temperature data, device sleep states, IO controller link states and latency reporting, and Processor Controller Hub (PCH) activity among others. The collected data can be correlated and visualized 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 v2.6.1 includes the following significant changes compared to v2.6:

- New features supported:
  - `xhci-lpm`: XHCI host controller link power management state residency. A table reporting capability settings will be included in a future release.

- Command line changes:
  - The `-f gfx-cstate` feature name has become a group name. The individual metric name is `-f hw-gfx-cstate` and reports C-state residencies for the graphics subsystem. This aligns feature names across all operating systems.

- The `--version` option output format has changed.

3 Fixed Issues

- Reduced tool overhead when collecting PCH IP active residency. Collecting PCH active (-f pch-ip-active) data too often can prevent the system from entering low power states, therefore if -m is specified, this data is now collected at a minimum interval of 5 seconds rather than the interval used for other metrics (default 100ms). The -n option no longer controls the minimum interval for -f pch-ip-active, use --option pch-int instead. Note that PCH active comes from residency counters so there is no loss in accuracy by decreasing the frequency at which that data is collected.
- Collecting -f pcie-lpm on a platform with a GbE device could result in significant tool overhead that reduced Pkg C10 residency. The problem has been resolved by reducing collection of the GbE controller link states from every 100ms to every 1 second. This is currently a fixed interval for GbE link state collection, not changed by using the -n option.
- Fixed collection issue when options -t and -p are specified. Previously specifying both the –t and –p options would result in a message similar to the following: "*FATAL: couldn't retrieve CPU topology at collection end*". This message is no longer reported after the fix.
- Fixed an issue where cpu-gpu-concurrency data would occasionally sum to >100%. This occurs when hardware over counts. SoC Watch now checks the timestamp delta and adjusts for over counting.
- A warning message is now displayed when you have specified metrics that cannot be collected at the same time due to hardware limitations. For example, features comp-max-temp and edram-state cannot be collected at the same time. Also, only one of the following metrics can be collected at a time: pmc-ip-d0i3-occ, pmc-ip-d0i3-res, pmc-ip-d3-occ, pmc-ip-d3-res, and pmc-ip-pg-res, s0i3-sstate, and s0i3-sstate-dbg. The warning message will indicate which metric is not being collected (e.g., *Warning: Exceeded hardware telemetry counter limitation. Will not collect metric "XXXXX"*).
- Trailing hyphens from report titles in the trace output have been removed (e.g., Graphics Active State).
- Large collection durations specified with -t option are now handled correctly. Previously, a time value > 2^{16} would be truncated.

4 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 v2.6.1 PWR files. We recommend using the latest version of Intel VTune Amplifier whenever possible.
- If the bandwidth is 0 Mb throughout the collection for a particular metric, 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 Intel SoC Watch v2.6.1 `ddr-bw` feature measures both bandwidth and bandwidth requests on Intel platforms code named Haswell, Broadwell, and Skylake. When this data is imported and visualized with Intel VTune Amplifier, only actual bandwidth is shown.
- 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 metrics are: `pch-platform-ltr, cpu-pkgc-dbg`. If there were no supported metrics in the collection, no .pwr file is created.

Bandwidth and DRAM Self Refresh

- Intel SoC Watch v2.6.1 requires loading the socperf v2.0 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 following error

  ERROR: ERROR configuring SOCPERF interface!

then both the socperf2_0.ko and socwatch2_6.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 graphics-cstate metric is obtained by frequently polling GPU counters that provide the gfx-cstate 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.
- 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.
- If socperf reads occur before the start of collection, a dmesg error message is generated: "socperf2_0: [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.
- 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's *Options Quick Reference* section.

- 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.
- Feature cpu-pkgc-dbg report for Transient/Uncommon Reasons is unreliable**.** All tables reported by the -f cpu-pkgc-dbg feature are correct, except for the last table Transient/Uncommon reasons which does not get collected at the same rate as the others. The issue is being investigated.
- 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.

S States & D States

- 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 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.

5 Related Documentation

The below documents are available with this release.

- Intel® SoC Watch for Google Android* OS, Google Chrome* OS, and Linux* OS User's Guide.

6 System Requirements

Supported Architectures

Intel SoC Watch for Google Chrome* and Linux* OS works on systems based on the following platforms:

- Apollo Lake (Broxon-P)
- Gemini Lake
- Skylake
- Kaby Lake
- Denverton
- Coffee Lake
- Cannonlake
- Broadwell
- Cherryview
- Anniedale
- Haswell
- Skylake-Xeon
- Ice Lake

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.

7 Installation Notes

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.

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). 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 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_6.ko and socperf2_0.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_6.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. For more information, see Extracting the Intel SoC Watch package. The

`<extract_dir>/system_studio_target/socwatch_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>/system_studio_target/socwatch_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 has an Intel Atom® processor, use the following command to build the socwatch2_6.ko and socperf2_0.ko files.

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

   b. If the target system has an Intel Core processor, use the following command to build the socwatch2_6.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`

8 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.

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 www.intel.com/design/literature.htm.

Intel, the Intel logo, Intel SoC Watch, and Intel VTune Amplifier are trademarks of Intel Corporation in the U.S. and/or other countries.

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

Microsoft, Windows, and the Windows logo are trademarks, or registered trademarks of Microsoft Corporation in the United States and/or other countries.