Intel® SoC Watch User Guide for Windows* OS

July 2025

Version 2025.5

Intel Corporation

www.intel.com

Legal Information

Contents

Legal Information

Intel technologies may require enabled hardware, software or service activation.

No product or component can be absolutely secure.

Your costs and results may vary.

Performance varies by use, configuration and other factors. Learn more at www.Intel.com/PerformanceIndex.

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

The products described may contain design defects or errors known as errata which may cause the product to deviate from published specifications. Current characterized errata are available on request.

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.

Version History

These are the main releases of Intel® SoC Watch:

Date	Revision	Description
June, 2019	2.11	Improves handling of unrecognized CPUs, reporting S-state when hibernation occurs, and other bug fixes.
September, 2019	2019.12	Added support for Intel platform code named Ice Lake. Modified hw-cpu-pstate reporting.
October, 2019	2019.13	Fixed issue in hw-cpu-pstate for Intel platform code named Ice Lake.
November, 2019	2020.1	Added support for Intel platform code named Comet Lake.
February, 2020	2020.2	Added collection of tool usage analytics. Added new features pch-slps0, pch-slps0-dbg. Improved error messages and help output. Enhanced driver security.
June, 2020	2020.3	Bug fix release.
July, 2020	2020.3.1	Bug fixes
September, 2020	2020.3.2	Bug fixes including correcting hibernation detection.
October, 2020	2020.4	Added support for Intel platform code named Tiger Lake. Added topology label in reports for some metrics. Re-named feature cpu-gpu-concurrency to cpu-igpu-concurrency. Removed support for older platforms.
November, 2020	2020.5	Changed hw-cpu-pstate to report frequencies per thread rather than per core. Added term *integrated* to hw-igfx-cstate and hw-igfx-pstate report titles.
April, 2021	2021.1	Added system name, OS name, and Intel PMT GUID to output reports. Included throt-rsn and other sampled count results in Automation_Summary. Re-ordered feature reports in summary output. Modified WakeupAnalysis report and added percentage table.
July, 2021	2021.2	Enhanced WakeupAnalysis report with idle time per process. Improved ia-throt-rsn support when Windows* OS secure modes are enabled. Bug fix release.
September, 2021	2021.3	General enhancements and third-party library updates.
October, 2021	2021.3.1	Improved support for platforms with many sockets and/or discrete graphics cards. Modified option --update-usage-consent.
November, 2021	2021.4	Added support for Intel platforms code named Rocket Lake and Tiger Lake - H.

Date	Revision	Description
		Added support for Intel platforms code named Ice Lake -X and Cooper Lake -X. Bug fixes.
December, 2021	2021.4.1	Improved tool stability.
February, 2022	2022.1	Added support for Intel platform code named Alder Lake and Intel® Arc Discrete Graphics Card . Bug fixes.
April, 2022	2022.2	Bug fixes.
June, 2022	2022.3	Bug fixes.
August, 2022	2022.4	Bug fixes.
September, 2022	2022.5	Bug Fixes.
November, 2022	2022.6	Report complete topology path on hybrid systems. Bug fixes for data correctness.
January, 2023	2023.0	Deprecation of Google Analytics support. Bug fixes for data correctness.
February, 2023	2023.1	Bug fixes for data correctness.
April, 2023	2023.2	Added support for Intel platform code named Raptor Lake. Added support for Intel server platform code named Sapphire Rapids -Xeon. Bug fixes including BSOD on systems with integrated graphics disabled when running igfx features. .
June, 2023	2023.3	Bug fixes.
August, 2023	2023.4	Bug fixes
October, 2023	2023.5	Change in scope for certain group features including -f sys. Bug fixes
November, 2023	2023.6	Added support for Intel server platform code named Emerald Rapids-Xeon. Bug Fixes
December, 2023	2023.7	Enhanced wakeup analysis histogram report Bug fixes.
January, 2024	2024.0	Enabled additional features support on Intel platform code named Meteor Lake. Bug Fixes.
February, 2024	2024.1	Bug fixes.
April, 2024	2024.2	Deprecation of support for older atom based features. Bug Fixes
May, 2024	2024.3	Enabled support for Intel discrete graphics code named Ponte Vecchio.

Date	Revision	Description
		Bug Fixes
July, 2024	2024.4	Bug fixes including data correctness issue in the feature sata-lpm.
September, 2024	2024.6	Added support to Intel platforms code named Elkhart Lake, Arrow Lake and Lunar Lake. Bug Fixes
October, 2024	2024.7	Additional features support on Intel Platform code named Lunar Lake Added support for option --result-slice-range. Bug Fixes
November, 2024	2024.8	Bug Fixes
January, 2025	2025.0	Bug fixes including an issue with the help output.
February, 2025	2025.1	Enabled support for Intel® Arc™ B-series graphics. Enabled support for the feature memss-pstate on Intel platform code named Lunar Lake. Bug fixes
April, 2025	2025.2	Added support for Intel server platforms code named Granite Rapids and Sierra Forest.
April, 2025	2025.3	Added support for latest Pcode telemetry GUID on Intel support for Intel® Arc™ B-series graphics.
May, 2025	2025.4	Added support for latest Pcode telemetry GUID on Intel support for Intel® Arc™ B-series graphics.
July, 2025	2025.5	Enabled additional features support for Intel platform code named Panther Lake PCD P. Enabledsupport for Intel platforms code named Panther Lake PCD H and Wildcat Lake. Bug fixes.

About Intel® SoC Watch 1

Intel® SoC Watch is a command line tool for monitoring and debugging system behaviors related to power consumption on Intel® architecture-based platforms. It reports active and low power states for the system/CPU/GPU/devices, processor frequencies and throttling reasons, wakeups, and other metrics that provide insight into the system's energy efficiency. The tool includes utility functions that include delaying the start of collection and launching an application prior to starting collection.

Data is collected from both hardware and OS sources. When using the default mode of collection, the tool collects data at normally occurring OS context-switch points so that the tool itself is not perturbing the system sleep states. Tool overhead when collecting during idle scenarios can be < 1%, however active workloads with a high-rate of context switching will increase the overhead. A minimum collection interval is used to control the rate of collection.

Intel SoC Watch writes a summary report file (.csv) at the end of collection on the system under analysis (target system), allowing immediate access to results. Additional result files can be specified including: an import file (.pwr) for Intel® VTune™ Profiler that can be used for visualization of correlated timelines for all the collected metrics with powerful zoom and filtering functions, and a time trace file (.csv) that can be viewed as a timelines in tools like Microsoft* Excel*.

Related Information

See the Intel® SoC Watch Release Notes for information on new features as well as known issues.

For online help, including information about importing results into Intel® VTune™ Profiler, see the Energy Analysis User Guide (https://software.intel.com/en-us/energy-analysis-user-guide).

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

Installation

2

See the *Intel® SoC Watch Release Notes* for supported platforms and installation instructions.

Getting Started with Intel® SoC Watch 3

1. Launch a command prompt with administrative privileges. Intel SoC Watch requires administrative access to load a device driver.
2. Run the `socwatch` command with one or more required options to collect energy data.
3. Review the reports generated by Intel SoC Watch to analyze energy consumption.
4. [Optional] Import the results into Intel VTune Profiler to review a visual representation of the data collected.

Listed below are a few basic commands that will have you looking at Intel SoC Watch reports quickly. Use the Options Quick Reference section to learn about all the commands, their options, and abbreviations.

To collect and report CPU C-state and P-state residency data and system sleep state residency for the platform, replace `<duration>` with the number of seconds you want to collect data and use the command below. Intel SoC Watch will write a summary report to the file `SOCWatchOutput.csv`.

```
socwatch -t <duration> -f cpu -f sstate
```

If you want to name the output files something other than the default `SOCWatchOutput`, replace `<string>` in the command below with the name you want. After collection, look for the `<string>.csv` file to find the summary report.

Tip
Include a folder name in the string to group the files under a folder. The folder will be created if it doesn’t exist.

```
socwatch -t <duration> -f cpu -f sstate -o <string>
```

To collect a broad set of system metrics for a high level look at platform power behavior, replace `<duration>` with the number of seconds you want to collect data in the following command. Intel SoC Watch will write a summary report to `SOCWatchOutput.csv`.

```
socwatch -t <duration> -f sys
```

To get detailed reports available for any metric being collected (e.g., idle and wakeup analysis and timelines), include the `--max-detail` (`-m`) option during collection and specify additional report types using the `--result` option. In the example below, all metrics in group name `sys` are collected for 60 seconds, including over time data and any detailed data for metrics that offer it. A trace file is generated (`-r int`) in addition to the summary report. All the results are in files named `run1` under the folder `sysDetail`.

```
socwatch -t 60 -f sys -m -r int -o sysDetail/run1
```

Note that trace (timeline) reports for long collections can take many minutes to generate. In this case, you may wish to generate only summary reports at the end of collection and later request the trace file be generated as shown below using the `--input` (`-i`) option. In this example, the sys group of metrics is collected for 60 minutes with maximum detail, but only summary report is generated initially. The second command causes the results to be re-processed and timeline reports generated. All of the new reports have the same base name and location unless a `-o` is included to give the re-processed results a new name.

Tip

Use the -o option in conjunction with -i to change the base name.

```
socwatch -t 3600 -f sys -m -o myResult
socwatch -i myResult -r int
```

To generate an export file containing all supported metrics that can be opened for viewing in Intel VTune Profiler, include the -r vtune option alongside any other collection parameters. The --max-detail option is required to generate a complete set of trace data for all metrics. The following command will collect platform data for 30 seconds and produce a file called run1.pwr that can be imported to Intel VTune Profiler.

```
socwatch -t 30 -f sys -r vtune -m -o run1
```

Entering Connected Standby (Modern Standby)

If running on a system that supports Connected Standby and has WDTF installed, use the --auto-connected-standby (-z) option to automatically put the system in Connected Standby and then exit standby when the collection time completes. You can manually exit standby early and collection will stop at that time and generate the reports. You can also enter Connected Standby manually after starting a collection.

NOTE

In order to use the --auto-connected-standby (-z) option, the Windows* OS Driver Test Framework (WDTF) must be installed. See the *Installation Notes* section in the Release Notes for instructions.

Your platform must be enabled for *S0 Low Power Idle* state for this option to successfully put the system into Modern Standby after the WDTF is installed. The following command should list a sleep state such as *Standby (S0 Low Power Idle) Network Connected*, as available on this system if Modern Standby is possible: powercfg /a

Example: The following command collects CPU C-state residency for 60 seconds after putting the system into Connected Standby. It will delay collection and the transition to standby by the --startdelay (-s) time of 20 seconds. (There is no option to delay the start of collection after entering standby.)

```
socwatch -s 20 -t 60 -f cpu-cstate -z
```

4 Options Quick Reference

Invoke Intel SoC Watch with administrator privilege, using the following syntax:

```
socwatch <general options><post-processing options><collection options>
```

- Order of options does not matter unless specifically noted.
- Help is displayed if no option is specified.
- All features are not available on all systems, so the help text is dynamic, meaning it displays only the collection options that are supported by the system on which it is run. The metrics available differ because of changes in the system's hardware architecture support. This User's Guide contains a list of all metrics across all systems.
- You can specify feature names that are not available or not enabled on a particular system. When the tool starts, it will display console messages regarding features that cannot be collected, but collection will proceed if at least one feature is valid on that system.

Intel SoC Watch terminates data collection for one of three reasons (whichever occurs first):

1. the `--time` option was specified and the timer elapsed,
2. the `--program` option was used and the specified program exited,
3. a Ctrl-C interrupt was entered in the command window.

The location and name of the results files is displayed at the end of a collection. The summary report will be there with that name and a `.csv` extension. Raw data files and additional files based on post-processing options specified on the command line are located there as well, all with the same base name (default name is `SoCWatchOutput`).

NOTE
Result files are replaced if the same name is used for multiple collections.

General Options

The following options display information about the tool or system on which it is run.

Abbreviation	Option Name	Description
	`--enable-alts`	Enable alternative key for terminating collection [ALT-S].
	`--export-help`	Write help output to JSON formatted file.
`-h`	`--help`	Display tool usage information and exit. The help shown is specific to the system on which it is run. Only metrics supported by the system architecture will be listed.
`-l`	`--log` *`<filename>`*	Redirect all console output, including errors, to specified file.
	`--print-fms`	Display CPU ID as Family.Model.Stepping and exit.
`-v`	`--version`	Display tool version information and exit.

Post-processing Options

The following options affect how results are reported and where they are stored.

Abbreviation	Option Name	Description
`-i`	`--input` *`<filename>`*	Specify the collection name (with full path) of an existing collection to generate reports.
`-o`	`--output` *`<filename>`*	Specify the output collection name (default "SoCWatchOutput"). Specifying `console` as the filename will cause the summary results to also write to stdout. If the filename already exists, the previous results will be replaced.
`-r`	`--result` *`<result_type>`*	Specify the type of result to generate. This option can be repeated to get multiple types of reports. Following are the result types that can be specified: • `sum` Write summary reports to .csv file. [default] • `int` Write over-time data to _trace.csv file. • `vtune` Generate .pwr file for import to Intel VTune Profiler. • `json` Generate .swjson file. • `auto` Write summary results as a single line to file Automation_Summary.csv in current directory. Appends results, does not overwrite. If column headers for the new result changed, new headers will be inserted. Use to generate sets of data in a single file for comparison.
	`--result-slice-range` *`<rangeStartTime in msec><rangeEndTime in msec>`*	Specify a time range in milliseconds bounding a subset of the result file to post-process. Time range specified should be relative to collection start time and only one result slice may be processed at a time. Use with the -i option to specify input file to post-process.

Collection Options

These options affect what is collected and how it is collected.

Abbreviation	Option Name	Description
`-f`	`--feature` *`<name>`*	Specify which metric to collect, choose from the group names or individual names listed in the tables below. This option can be repeated to collect multiple metrics in a single run. Most features can be collected simultaneously, exceptions noted in the table of feature names.
`-m`	`--max-detail`	Collect all data available for each feature specified. This will cause snapshot metrics to be sampled. Use of this option can increase tool overhead, so best used only

Abbreviation	Option Name	Description
		when timeline of the data is needed or when collecting across system entry to hibernation. Without this option, the tool collects data at the minimum required by the data source for best accuracy. Data may be traced, sampled, or snapshot. • Traced data is obtained at state transition points resulting in accurate summary and timeline results. • Sampled data is read at OS context switch points (or at timed intervals if polling option is used). This is less accurate as changes that take place between samples will not be measured. Metrics that come from hardware status/ state data must be sampled. • Snapshot data can be read at the beginning and end of the collection and the difference gives an accurate result with lowest overhead, but no timeline. Only metrics that come from hardware accumulators can be snapshot. The algorithm used to determine the collection method for each data type is as follows: If `-m` is specified: if the data can be traced, trace it; else sample it. If `-m` is not specified: if the data can be snapshot, snapshot it; else if the data can be traced, trace it; else sample it.
`-n`	`--interval` *`<milliseconds>`*	Specify the time in milliseconds that should pass before reading next hardware data sample (default 100 ms). For default collection mode, this is the minimum time between sampling at context switch points. When `--polling` option is used, this is actual time between samples. The minimum polling interval is 1ms. However, using low polling intervals will result in higher overhead and may fail to measure some metrics (e.g. bandwidths) with intervals shorter than the default.
	`--no-post-processing`	Do not generate the summary file or other result files at the end of collection. Use -i option to process the intermediate results files and generate summary or other result file types at a later time.

Abbreviation	Option Name	Description
	`--polling`	Make data collection occur at regular intervals rather than at context switch points. Use the `--interval` option to set the interval period (default: 100ms). Use of this option significantly increases perturbation of sleep states because it employs a timer which will interrupt sleep states, increase wakeup counts, and change timer resolution.
`-p`	`--program` *<application> <parameters>*	Specify the name of an executable to be started automatically prior to collection. The name can be followed by zero or more arguments that will be passed to the program. **NOTE** This option must occur at the end of the command line, everything following the executable name will be given to it as arguments. **NOTE** The executable can only be launched as a desktop application (not as Windows Modern UI application).
	`--program-delay` *<seconds>*	Specify number of seconds to wait before starting the program specified by `-p`. Has no effect if `-p` not used.
`-s`	`--startdelay` *<seconds>*	Specify number of seconds to wait before starting collection of data. If used with `-p` and `--program-delay`, this delay is applied after the program starts. If used with `--auto-connected-standby`, this delays entry to standby.
`-t`	`--time` *<seconds>*	Specify collection duration in seconds. Collection will stop when this time has elapsed unless Ctrl-C is entered or an executable specified with `--program` option exits prior to the specified duration.
`-z`	`--auto-connected-standby`	Automatically enter Connected Standby for the duration of the collection. Will automatically exit Connected Standby when the -t specified time expires. If system is woken from Connected Standby prior to the end of the duration, the collection will stop as well. If `--start-delay` is specified, it occurs prior to entering Connect Standby.

Abbreviation	Option Name	Description
		NOTE Requires Windows* Driver Test Framework, WDTF to be installed on the system under test.

Feature Names (Individual)

The available feature names for the `--feature` option and their collection methods are listed below. You can specify multiple feature names individually or using group names described in the Feature Group Names section.

Note that every feature listed is not available on every platform supported by Intel SoC Watch. The `--help` option is dynamic, only showing features available for the platform on which it is run. Use it to determine which features are supported. You can specify unsupported features on the command line and the tool will simply display a message for those that cannot be collected, but continue with collection if there is at least one that is supported.

Collection methods are indicative of a metric’s level of accuracy and overhead. Traced collection provides high accuracy along with precise transition points between states. Sampled collection is least accurate since transitions can occur which are never noted. Sampled data needs to be read at intervals throughout the collection period which increases tool overhead. Increasing the sampling rate (reading at closer intervals) will improve accuracy but increase overhead. Snapshot collection means the data comes from an accumulator so it can be collected only at the start and end of the collection period and give perfect accuracy. This gives accuracy and the lowest overhead. If the `--max-detail` (`–m`) option is given, the Snapshot metrics will instead be read at the same intervals as the Sampled metrics throughout the collection, so that you can generate a trace file to see how it changed overtime.

The list of supported features are:

Name	Collection Methods	Description
`acpi-dstate`	Trace	Device D-state residencies, from OS event trace. If a device has multiple components, the component F-state residency report is included. The list of devices included in the report are filtered to only include selected ACPI devices. To get the complete device list, disable the filter using option `--option no-device-filter`.
`acpi-sstate`	Trace	ACPI Sx system state residency (S1-S4), from OS event trace.
`cdie-ccp-llc-bw`	Sampled	Converged Core Perimeter (per Core/Module) to LLC Read and Write bandwidth for Compute Die. [from Intel (R) PMT]
`cdie-cstate-dbg`	Snapshot	Reasons blocking entry to lower-power Die C-states and wakeup from Die C-states for Compute Die. [from Intel (R) PMT]
`cdie-cstate-res`	Snapshot	Residency for Die C-states (DCx.y) for Compute Die. [from Intel (R) PMT]
`core-temp`	Sampled	IA core temperature statistics, from hardware status data.

Name	Collection Methods	Description
`cpu-igpu-concurrency`	Snapshot	Concurrent active time of CPU and integrated GPU, from hardware accumulators.
`cpu-pkgc-dbg`	Snapshot	Counts for reasons blocking entry to Package C-states and wakeups from Package C-states, from hardware accumulators. **NOTE** This data is retrieved from Intel (R) PMT on Intel Platforms starting from Meteor Lake.
`cpu-pkg-cstate-res`	Snapshot	Residency for Package C-states (PCx). [from Intel (R) PMT]
`ddr-bw`	Sampled	Total bandwidth per memory channel, from hardware accumulators. The hardware accumulator data is always collected over time due to frequent overflow, so snapshot is not available.
`dgfx-media-cstate`	Sampled	Discrete Graphics Media C-state (RCx) residency. [from Intel (R) PMT]
`dgfx-pcie-dstate`	Sampled	Discrete Graphics PCIE USP controller D-state (Dx) residency. [from Intel (R) PMT]
`dgfx-pkg-cstate`	Snapshot	Discrete graphics Package C-state residency (PCx). [from Intel (R) PMT]
`dgfx-pwr`	Sampled	Discrete graphics package energy usage. [from Intel (R) PMT]
`dram-pwr`	Sampled	Total DRAM power consumption from hardware accumulators. This data can be collected on Intel server platforms.
`hw-cpu-hwp`	Sampled	HWP capabilities, from hardware status data.
`hw-cpu-pstate`	Sampled	CPU P-state operating frequency residencies, from hardware status data.
`hw-igfx-cstate`	Snapshot*	Integrated GPU C-state residency (RC6), from hardware accumulators. *Always sampled due to short overflow time period.
`hw-igfx-pstate`	Sampled	Integrated GPU P-states approximated residency, from hardware status data.
`ia-throt-rsn`	Sampled	Reasons for throttling the CPU frequency, from hardware status data.
`igfx-throt-rsn`	Sampled	Reasons for throttling the integrated GPU frequency, from hardware status data.

Name	Collection Methods	Description
`io-bw`	Sampled	Total IO bandwidth Reads and Writes. [from Intel (R) PMT]
`memss-pstate`	Sampled	Memory subsystem (QCLK) P-states approximated residency. [from Intel (R) PMT]
`npu-bw`	Sampled	NPU memory bandwidth. [from Intel (R) PMT]
`npu-dstate-cnt`	Sampled	Entrance count for NPU D-states (D0/D0ix). [from Intel (R) PMT]
`npu-dstate-res`	Sampled	Residency for NPU D-states (D0/D0ix). [from Intel (R) PMT]
`npu-pwr`	Snapshot	NPU power statistics. [from Intel (R) PMT]
`pch-slps0`	Snapshot	PCH SLP_S0 residency, from hardware accumulator.
`pch-slps0-dbg`	Sampled	Blocking reasons for SLP_S0, from hardware status data.
`pkg-pwr`	Snapshot	Total SoC/Package power consumption, from hardware accumulator.
`ring-throt-rsn`	Sampled	Reasons for throttling the ring clock frequency, from hardware status data.
`timer-resolution`	Trace	Timer resolution change events from OS trace events.

Feature Group Names

The following features are groupings of the previously described features. These group names can be used to simplify command lines to collect multiple features concurrently. For example, `-f cpu` can replace the `-f cpu-cstate -f cpu-pstate` in a command line.

If a group includes a feature that is not enabled on the target platform, that feature will be ignored and collection continue, as long as there is one feature that can be collected.

All features are not supported on all platforms, a group will only include the supported features. Use the `--help` option on the target platform to see the list of group names and specific features included each group.

The list of supported feature groups are:

Name	Description
`chipset-all`	All PCH related metrics (all PCH active groups)
`cpu`	cpu-hw + cpu-os
`cpu-hw`	Most CPU metrics obtained from hardware data sources
`cpu-os`	All CPU metrics obtained from OS event traces
`cpu-pkg-cstate`	Includes cpu-pkg-cstate-cnt
`cpu-pstate`	CPU P-state metrics
`device`	Device state residency metrics

Name	Description
gfx	All graphics metrics from hardware and OS. gfx-hw + gfx-os
gfx-hw	Most GPU metrics obtained from hardware data sources.
gfx-os	All GPU metrics obtained from OS event traces
power	Power/energy metrics
sys	Spectrum of metrics commonly used to get general information about platform power behavior. Provides information about physical components (CPU, GPU) and not include firmware component information
temp	Temperature metrics
throt	Frequency throttling reason metric.

Viewing Intel SoC Watch Results with Intel® VTune™ Profiler 5

You can analyze Intel SoC Watch data graphically using the Intel® VTune™ Profiler GUI. Intel® VTune™ Profiler provides a dynamic timeline view for interacting with Intel SoC Watch data and provides powerful filtering of data for in-depth analysis of a platform's power management behavior.

For detailed instructions, refer to the Energy Analysis section of the Intel® VTune™ Profiler Help.