## Power Profiler

CodeXL's Power Profiler is a powerful tool to help analyze the energy efficiency of systems based on AMD APUs and majority of the recent dGPU(discrete GPU).

Features of the Power Profiler include:

* Report the following data:
  + Estimated average power consumed by APU and supported dGPU subcomponents.
  + Average frequency of the CPU cores and the internal GPU and supported dGPU.
  + Thermal trend of the CPU compute-units and the internal GPU.
  + Thermal trend of supported dGPU.
  + CPU cores P-States.
* A command-line tool to for data collection and dump to text/binary format.
* The CodeXL graphic client provides these Power Profiling capabilities:
  + Real-time monitoring
  + Timeline view
  + Summary view
  + Offline review of session data
* Following hardware’s are supported
  + AMD APUs: Carrizo, Kaveri, Mullins, Temash, Stoney, Bristol
  + AMD dGPUs: Graphics IP 7 GPUs, Radeon and FirePro models.

**Installing the Power Profiler Linux Driver**

[**Wider Linux power profiling support (DKMS)**](#_Wider_Linux_power)

**Power Profiler’s Performance Counters**

**Power Profiler Command Line Interface**

**Power Profiler Project Settings**

**Power Profiler Counters Selection Dialog**

**Power Profiler Session Tree**

**Power Profiler Timeline View**

**Power Profiler Summary View**

**Power Profiler Session Control**

**Remote Power Profiling**

**[AMDTPowerProfileAPI Library](#_AMDTPowerProfileAPI_Library)**

**Limitations**

Legal Disclaimer

The reports generated from this Software may include thermal and power estimates which may contain errors. The report values may deviate from the actual values and/or the published specifications. Therefore, these reports should not be relied upon for any commercial or production systems and the user is responsible for obtaining accurate thermal and power measurements.

### Installing the Power Profiler Linux Driver

On Linux systems, the CodeXL Debian and RPM packages perform the driver installation automatically. However, if you’ve downloaded the CodeXL tar archive, you have to install the Power Profiler’s Linux driver manually. This includes a simple step of running **AMDTPwrProfDriver.sh** script with root credentials.

Example:

$ tar –xf CodeXL\_Linux\_x86\_64\_2.0.XXXX.tar.gz

$ cd CodeXL\_Linux\_x86\_64\_2.0.XXXX

$ sudo ./AMDTPwrProfDriver.sh install

Installer will create a source tree for power profiler driver under /usr/src/ amdtPwrProf -<version number>. All the source files required for module compilation is located in this directory are under MIT license.

To uninstall the driver run the following command:

$ $ cd <codexl-install-dir>

$ sudo ./AMDTPwrProfDriver.sh uninstall

### Wider Linux power profiling support (DKMS)

On Linux machine Power Profiler driver can also be installed with Dynamic Kernel Module Support (DKMS) framework support. DKMS framework automatically upgrades the power profiler driver module whenever there is a change in the existing kernel. This saves user from manually upgrading the power profiler driver module.

The DKMS package needs to be installed on target machines before running the installation steps mentioned in the above section. AMDTPwrProfDriver.sh installer script will automatically takes care of DKMS related configuration if DKMS package is installed in the target machine.

Example (for Ubuntu system):

$ sudo apt-get install dkms

$ tar –xf CodeXL\_Linux\_x86\_64\_2.0.XXXX.tar.gz

$ cd CodeXL\_Linux\_x86\_64\_2.0.XXXX

$ sudo ./AMDTPwrProfDriver.sh install

If the user upgrades the kernel version frequently it is recommended to use DKMS for installation.

### Power Profiler’s Performance Counters

|  |  |  |  |
| --- | --- | --- | --- |
| **Category** | **Name** | **Description** | **Notes** |
| **Power** | Total APU Power | Average APU Power for the sampling period, reported in Watts. This is an estimated consumption value which is calculated based on APU activity levels. |  |
| CPU Compute Unit 0 Power  CPU Compute Unit 1 Power | Average CPU Compute Unit Power for the sampling period, reported in Watts. This is an estimated consumption value which is calculated based on APU activity levels. |  |
| iGPU Power | Average Integrated-GPU Power for the sampling period, reported in Watts. This is an estimated consumption value which is calculated based on APU activity levels. |  |
| PCIe-Controller Power | Average PCIe-Controller Power for the sampling period, reported in Watts. This is an estimated consumption value which is calculated based on APU activity levels. This value does not include the power consumed by PCIe devices connected to the PCIe bus. |  |
| Memory-Controller Power | Average DDR Memory-Controller Power for the sampling period, reported in Watts. This is an estimated consumption value which is calculated based on APU activity levels. This value does not include the power consumed by the memory DIMMs. |  |
| Display-Controller Power | Average Display-Controller Power for the sampling period, reported in Watts. This value refers to the APU's internal display controller which may be used in notebook and embedded configurations. This is an estimated consumption value which is calculated based on APU activity levels. This value does not include the power consumed by the display. |  |
| Cumulative APU Power | The accumulated energy consumed by the APU throughout the profile session. Reported in Joules. | Available only in the command line tool |
| Cumulative Compute Unit 0 Power  Cumulative Compute Unit 1 Power | The accumulated energy consumed by the CPU Compute Unit throughout the profile session. Reported in Joules. | Available only in the command line tool |
| Cumulative iGPU Power | The accumulated energy consumed by the APU’s Internal GPU throughout the profile session. Reported in Joules. | Available only in the command line tool |
| dGPU power | Average Discrete-GPU Power for the sampling period, reported in Watts. This is an estimated consumption value which is calculated based on dGPU activity levels. | The dGPU family name is prefixed with this counter name. |
| **Frequency** | CPU Core 0 Average Frequency  CPU Core 1 Average Frequency  CPU Core 2 Average Frequency  CPU Core 3 Average Frequency | Average CPU Core Frequency for the sampling period, reported in MHz. This is the Core Effective Frequency (CEF). The core can go into various P-States within the sampling period, each with its own frequency. The CEF is the average of the core frequencies over the sampling period. |  |
| iGPU Average Frequency | Average Integrated-GPU Frequency for the sampling period, reported in MHz. |  |
| dGPU Average Frequency | Average Discrete-GPU Frequency for the sampling period, reported in MHz. | The dGPU family name is prefixed with this counter name. |
| CPU Core 0 Frequency Histogram  CPU Core 1 Frequency Histogram  CPU Core 2 Frequency Histogram  CPU Core 3 Frequency Histogram | Histogram of CPU Core Effective Frequency (average frequency for the sampling period). | Available only in the command line tool |
| iGPU Frequency Histogram | Histogram of Internal-GPU Effective Frequency (average frequency for the sampling period). | Available only in the command line tool |
| **Temperature** | CPU Compute-Unit 0 Measured Temperature  CPU Compute-Unit 1 Measured Temperature | Measured CPU Compute Unit Average Temperature, reported in Celsius. The reported value is normalized and scaled, relative to the specific processor's maximum operating temperature. This value can be used to indicate rise and decline of temperature. |  |
| iGPU Measured Temperature | Measured Integrated-GPU Average Temperature, reported in Celsius. The reported value is normalized and scaled, relative to the specific processor's maximum operating temperature. This value can be used to indicate rise and decline of temperature. |  |
| dGPU Measured Temperature | Measured Discrete-GPU Average Temperature, reported in Celsius. The reported value is normalized and scaled, relative to the specific processor's maximum operating temperature. This value can be used to indicate rise and decline of temperature. | The dGPU family name is prefixed with this counter name. |
| **CPU Core State** | CPU Core 0 P-State  CPU Core 1 P-State  CPU Core 2 P-State  CPU Core 3 P-State | CPU Core P-State at the time when sampling was performed. |  |
| **Other counters** | Core 0 Process ID  Core 1 Process ID  Core 2 Process ID  Core 3 Process ID | Process Id of the process that was executed by the CPU core at the point in time when the core was sampled. | Available only in the command line tool |

### Power Profiler Command Line Interface

CodeXL Power Profiler provides a command line interface utility for users who prefer to use command interpreters like **cmd.exe** on Windows and **bash** on Linux. This CLI utility can be used to collect and analyze the profile data. It can also be used from a batch file or a test script.

**Usage**:

On Windows:

CodeXLPowerProfiler.exe <options>

On Linux:

CodeXLPowerProfiler <options>

The following options are supported:

|  |  |
| --- | --- |
| -P <profile options> | Specify what types of data will be collected during the profile session. The following data types are supported.  **power** - collect all the available power counters  **temperature** - collect all the available temperature counters  **frequency** - collect all the available frequency counters  **cu\_power** - collect cpu compute-unit power counters  **cu\_temperature** - collect cpu compute-unit temperature counters  **gpu\_power** - collect gpu power counters  **gpu\_temperature** - collect gpu temperature counters  **core** - collect core specific attributes. This includes core frequency, p-state and process id executing on the core.  **all** - collect all the supported counters |
| -l | List all the counters supported by the local hardware, and the hardware devices that the profiler recognizes.  The counter IDs can be used with the ‘-e’ option.  The hardware device IDs can be used with the ‘-D’ option. |
| -e <counter,...> | Specify the comma separated list of counter names to be collected.  Use option ‘-l’ to get the supported counter names.  Note: use any one of the options -P or -e. |
| -D <counter,...> | Specify the comma separated list of device ids. All the counters of these devices will be profiled and collected.  Use option (-l) to get the supported devices. |
| -T <sampling interval> | Sampling interval in milli-seconds. The minimum value is 20ms. |
| -d <duration> | Profile duration in seconds. |
| -o <path> | Specify the output file path. The default path will be %Temp%\Codexl-Power\_<timestamp> on Windows and /tmp/ Codexl-Power\_<timestamp> on Linux. |
| -F <csv|txt> | Define the output file format:   * csv - Comma Separated Value text file * txt - plain text file   The default file format is CSV file. |
| -C <core mask> | Specify core affinity mask for the application to be launched. Default affinity mask is all the available CPU cores.  0x1 = Core 0  0x2 = Core 1  0x4 = Core 2  0x8 = Core 3 |
| -b | Terminate the launched application after the specified profile duration. |
| -w | Specify the working directory. Default will be the path of the launched application. |
| -h | Displays this help information. |
| -v | Print version string. |
| -z <db file output dir> | Export results to a \*.cxldb file which can be imported to CodeXL GUI application. |
| -M process | Process profiling is based on the IPC load. Collects power consumption of all running processes during their profile run. These power values may differ with the actual power consumption. These power values can be used to get a notion of power trend and relative power consumption among running processes.  Note: Currently PMC counters are supported only on windows operating system. Supported AMD platforms are Kaveri, Mullins, Temash and Carrizo. Please refer limitation section for further details. |
| -M module | Module profiling is based on the IPC load. Collects power consumption of all running modules during their profile run. These power values may differ with the actual power consumption. These power values can be used to get a notion of power trend and relative power consumption among running modules.  Note: Currently PMC counters are supported only on windows operating system. Supported AMD platforms are Kaveri, Mullins, Temash and Carrizo. Please refer limitation section for further details. |

Examples

* Collect power values for CPU Compute Units and GPU for the duration of 10 seconds, with sampling interval of 100 milliseconds:

CodeXLPowerProfiler.exe –P cpu\_power -P gpu\_power -o c:\Temp\powerprof-out.txt –T 100 –d 10

* Collect all the supported counter values for the duration of 10 seconds with sampling interval of 100 milliseconds:

CodeXLPowerProfiler.exe –P all -o c:\Temp\powerprof-out.txt –T 100 –d 10

* Collect all the supported counter values for the duration of 300 seconds with sampling interval of 100 milliseconds, and output the data to a binary file that can be imported into the graphic client:

CodeXLPowerProfiler.exe –P all –C 0x3 -z c:\Temp\PowerOutput –T 100 –d 300

* Collect all running process with their energy consumption shares as well as all counters during a particular profile run. Where profile duration is set to 300 seconds and profile sampling period is set to 10 milliseconds.

CodeXLPowerProfiler.exe –M process –e all c:\Temp\PowerOutput –T 10 –d 300

* Display help:

CodeXLPowerProfiler.exe -h

* Display version string:

CodeXLPowerProfiler.exe -v

Sample Text output files

Sample 1: Collecting non-cumulative counters

The following command collects all non-cumulative power counters for 10 seconds, sampling them every 500 milliseconds and dumping the results to a text file:

CodeXLPowerProfiler.exe -P power -d 10 -T 500 -o C:\temp\pwr\_out.txt –F txt

When run on a Kaveri APU, the content of the result text file is as below:

CODEXL POWER PROFILE REPORT

PROFILE DETAILS

CPU Details: Family(0x15) Model(48)

CPU Core Mask: 0xf

Sampling Interval: 500 milli-seconds

Profile Start Time: Dec-09-2014\_18-10-08

Profile Duration: 10 seconds

PROFILED COUNTERS

COUNTER ID NAME CATEGORY UNIT DESCRIPTION

0. pcie-ctrl-power Power Watt Average PCIe-Controller Power for the sampling period, …

1. mem-ctrl-power Power Watt Average DDR Memory-Controller Power for the sampling period,…

2. total-apu-power Power Watt Average APU Power for the sampling period, reported in Watts…

4. display-ctrl-power Power Watt Average Display-Controller Power for the sampling period,…

5. cpu-cu0-power Power Watt Average CPU Compute Unit Power for the sampling period, …

16. cpu-cu1-power Power Watt Average CPU Compute Unit Power for the sampling period,…

27. igpu-power Power Watt Average Integrated-GPU Power for the sampling period, …

PROFILE RECORDS

RecordId Timestamp pcie-ctrl-power mem-ctrl-power total-apu-power display-ctrl-powe cpu-cu0-power cpu-cu1-power igpu-power

0 18:10:9:002 0.56 4.88 56.28 0.00 15.49 17.36 17.98

1 18:10:9:501 0.56 4.79 39.92 0.00 13.29 13.19 8.08

2 18:10:10:003 0.56 4.83 38.00 0.00 12.86 11.58 8.17

3 18:10:10:503 0.56 4.80 38.05 0.00 12.70 12.02 7.96

4 18:10:11:003 0.56 4.79 38.80 0.00 12.84 12.53 8.07

5 18:10:11:503 0.56 4.79 38.95 0.00 13.05 12.63 7.92

6 18:10:12:003 0.56 4.82 38.16 0.00 12.15 12.57 8.06

7 18:10:12:503 0.56 4.81 37.93 0.00 12.23 12.42 7.91

8 18:10:13:003 0.56 4.80 37.46 0.00 11.97 12.06 8.05

9 18:10:13:503 0.56 4.80 37.64 0.00 12.32 11.97 7.99

10 18:10:14:003 0.56 4.83 38.91 0.00 12.49 12.99 8.04

11 18:10:14:501 0.56 4.79 38.56 0.00 12.54 12.71 7.96

12 18:10:15:003 0.56 4.81 38.89 0.00 12.38 13.08 8.06

13 18:10:15:501 0.56 4.80 39.38 0.00 12.71 13.30 8.00

14 18:10:16:003 0.56 4.80 39.37 0.00 13.03 12.91 8.07

15 18:10:16:503 0.56 4.80 39.25 0.00 12.99 12.91 7.99

16 18:10:17:003 0.56 4.80 39.72 0.00 12.50 13.80 8.05

17 18:10:17:503 0.56 4.80 38.97 0.00 12.93 12.67 8.01

18 18:10:18:003 0.56 4.81 38.95 0.00 12.97 12.56 8.05

19 18:10:18:503 0.56 4.80 39.21 0.00 12.75 13.11 7.99

20 18:10:19:003 0.56 4.80 38.66 0.00 12.69 12.56 8.05

Sample 2: Collecting cumulative counters

The following command collects the cumulative power counters for 10 seconds and dumping the results to a text file:

CodeXLPowerProfiler.exe -d 10 -o c:\temp\cumulative.txt -e 3,6,17,28 –F txt

When run on a Kaveri APU, the content of the result text file is as below:

CODEXL POWER PROFILE REPORT

PROFILE DETAILS

CPU Details: Family(0x15) Model(48)

CPU Core Mask: 0xf

Sampling Interval: 100 milli-seconds

Profile Start Time: Dec-11-2014\_15-57-44

Profile Duration: 10 seconds

PROFILED COUNTERS

COUNTER ID NAME CATEGORY UNIT DESCRIPTION

3. total-apu-power-cuml Power Joule Cumulative APU Power, reported in Joules.

6. cpu-cu0-power-cuml Power Joule Cumulative CPU Compute Unit Power, reported in Joules.

17. cpu-cu1-power-cuml Power Joule Cumulative CPU Compute Unit Power, reported in Joules.

28. igpu-power-cuml Power Joule Cumulative Integrated-GPU Power, reported in Joules.

CUMULATIVE COUNTERS

COUNTER CUMULATIVE VALUE

total-apu-power-cuml 228.11

cpu-cu0-power-cuml 48.85

cpu-cu1-power-cuml 50.30

igpu-power-cuml 78.65

Sample 3: Collecting histogram counters

The following command collects the frequency histogram counters for 10 seconds and dumping the results to a text file:

CodeXLPowerProfiler.exe -d 10 -o c:\temp\histogram.txt -e 10,14,21,25,31 –F txt

When run on a Kaveri APU, the content of the result text file is as below:

CODEXL POWER PROFILE REPORT

PROFILE DETAILS

CPU Details: Family(0x15) Model(48)

CPU Core Mask: 0xf

Sampling Interval: 100 milli-seconds

Profile Start Time: Dec-11-2014\_16-00-35

Profile Duration: 10 seconds

PROFILED COUNTERS

COUNTER ID NAME CATEGORY UNIT DESCRIPTION

10. cpu-core0-frequency-hist Frequency MHz Histogram of CPU Core Effective Frequency.

14. cpu-core1-frequency-hist Frequency MHz Histogram of CPU Core Effective Frequency.

21. cpu-core2-frequency-hist Frequency MHz Histogram of CPU Core Effective Frequency.

25. cpu-core3-frequency-hist Frequency MHz Histogram of CPU Core Effective Frequency.

31. igpu-frequency-hist Frequency MHz Histogram of Integrated-GPU Frequency.

HISTOGRAMS OF COUNTERS

COUNTER cpu-core0-frequency-hist

HISTOGRAM

low high count

0 200 0

200 400 0

400 600 0

600 800 16

800 1000 16

1000 1200 5

1200 1400 22

1400 1600 29

1600 1800 11

1800 2000 1

2000 2200 0

2200 2400 0

2400 2600 0

2600 2800 0

2800 3000 0

3000 3200 0

3200 3400 0

3400 3600 0

3600 3800 0

3800 4000 0

COUNTER cpu-core1-frequency-hist

HISTOGRAM

low high count

0 200 0

200 400 0

400 600 0

600 800 6

800 1000 79

1000 1200 13

1200 1400 1

1400 1600 1

1600 1800 0

1800 2000 0

2000 2200 0

2200 2400 0

2400 2600 0

2600 2800 0

2800 3000 0

3000 3200 0

3200 3400 0

3400 3600 0

3600 3800 0

3800 4000 0

COUNTER cpu-core2-frequency-hist

HISTOGRAM

low high count

0 200 0

200 400 0

400 600 0

600 800 0

800 1000 0

1000 1200 0

1200 1400 0

1400 1600 2

1600 1800 98

1800 2000 0

2000 2200 0

2200 2400 0

2400 2600 0

2600 2800 0

2800 3000 0

3000 3200 0

3200 3400 0

3400 3600 0

3600 3800 0

3800 4000 0

COUNTER cpu-core3-frequency-hist

HISTOGRAM

low high count

0 200 0

200 400 0

400 600 0

600 800 0

800 1000 28

1000 1200 45

1200 1400 25

1400 1600 1

1600 1800 1

1800 2000 0

2000 2200 0

2200 2400 0

2400 2600 0

2600 2800 0

2800 3000 0

3000 3200 0

3200 3400 0

3400 3600 0

3600 3800 0

3800 4000 0

COUNTER igpu-frequency-hist

HISTOGRAM

low high count

0 100 0

100 200 0

200 300 0

300 400 100

400 500 0

500 600 0

600 700 0

700 800 0

800 900 0

900 1000 0

Sample 3: Collecting process profiling data

The following command collects the process profiling data for 100 seconds and dumping the results to a text file:

CodeXLPowerProfiler.exe -d 10 –T 10 -o c:\temp\histogram.txt –M process –F txt

When run on a Carrizo APU, the content of the result text file is as below:

CODEXL POWER PROFILE REPORT

PROFILE DETAILS

CPU Details: Family(0x15) Model(0x60)

CPU Core Mask: 0xf

Sampling Interval: 10 milli-seconds

Profile Start Time: Mar-30-2016\_00-20-44

Profile Duration: 100 seconds

PROCESS PROFILING DATA

SNo PID Samples IPC Power(Joules) Power(%) Name Path

0 4956 11 3.34 0.04 0.03 CodeXLPowerProfiler.exe \Program Files (x86)\CodeXL

1 0 383503 31755 109.58 95.28 [System Process] Unable to read path

2 648 1 0.38 0.01 0.01 lsass.exe C:\Windows\System32

3 900 54 13.72 0.06 0.05 dwm.exe C:\Windows\System32

4 4 224 96.76 0.24 0.21 System Unable to read path

5 2140 294 232.54 0.82 0.71 explorer.exe C:\Windows

6 760 4 1.04 0.00 0.00 svchost.exe C:\Windows\System32

7 1648 16 5.71 0.02 0.01 MsMpEng.exe Unable to read path

8 548 15 2.21 0.01 0.01 csrss.exe Unable to read path

9 4396 1066 710.18 3.86 3.36 iexplore.exe C:\Program Files (x86)\Internet Explorer

10 680 178 170.15 0.29 0.25 svchost.exe C:\Windows\System32

11 5296 1 0.44 0.00 0.00 conhost.exe C:\Windows\System32

12 2500 2 0.29 0.00 0.00 svchost.exe C:\Windows\System32

13 1320 6 2.57 0.01 0.01 svchost.exe C:\Windows\System32

14 1160 2 0.80 0.00 0.00 SearchIndexer.exe C:\Windows\System32

15 456 35 15.70 0.04 0.03 svchost.exe C:\Windows\System32

16 1108 11 7.65 0.02 0.01 svchost.exe C:\Windows\System32

17 640 3 0.84 0.00 0.00 services.exe Unable to read path

18 936 2 1.00 0.00 0.00 svchost.exe C:\Windows\System32

19 1448 3 1.22 0.00 0.00 taskhostex.exe C:\Windows\System32

20 720 2 0.46 0.00 0.00 svchost.exe C:\Windows\System32

21 1612 2 0.63 0.00 0.00 WUDFHost.exe C:\Windows\System32

22 1296 4 1.42 0.00 0.00 spoolsv.exe C:\Windows\System32

23 2736 1 0.25 0.00 0.00 sppsvc.exe Unable to read path

Profile Sesssion Power Consumption: 115.00

Total PID record collected 385440

Sample 4: Collecting module profiling data

The following command collects the module profiling data for 10 seconds and dumping the results to a text file:

CodeXLPowerProfiler.exe -M module -T 10 -d 2 -O c:\temp\module\_profile.txt -F txt

When run on a Carrizo APU, the content of the result text file is as below:

CODEXL POWER PROFILE REPORT

PROFILE DETAILS

CPU Details: Family (0x15) Model (0x60)

CPU Core Mask: 0xf

Sampling Interval: 10 milli-seconds

Profile Start Time: Jul-28-2016\_02-29-02

Profile Duration: 10 seconds

MODULE PROFILING DATA

SNo PID Process Name Samples Kernel Power Power(%)Load Addr size Module Name ModulePath

0 0 System Idle 6214 1 4.4 49.31 0xf8032ca0e000 7913472 ntoskrnl.exe C:\Windows\system32\

1 1920 Explorer.EXE 263 1 0.96 10.8 0xf8032ca0e000 7913472 ntoskrnl.exe C:\Windows\system32\

2 1920 Explorer.EXE 97 0 0.57 6.35 0x7ffa520a0000 22192128 SHELL32.dll C:\Windows\system32\

3 1920 Explorer.EXE 116 0 0.51 5.74 0x7ffa53a10000 1757184 ntdll.dll C:\Windows\SYSTEM32\

4 364 svchost.exe 76 0 0.33 3.73 0x0 0 unknown name unknown path

5 1920 Explorer.EXE 81 1 0.23 2.6 0xf960000fa000 5783552 win32k.sys C:\Windows\System32\

6 828 dwm.exe 56 0 0.2 2.29 0x0 0 unknown name unknown path

7 1920 Explorer.EXE 37 0 0.17 1.91 0x7ffa4ece0000 1839104 DUI70.dll C:\Windows\SYSTEM32\

8 1920 Explorer.EXE 33 0 0.12 1.3 0x7ffa513d0000 2166784 combase.dll C:\Windows\SYSTEM32\

9 1920 Explorer.EXE 22 0 0.11 1.2 0x7ffa515f0000 696320 msvcrt.dll C:\Windows\system32\

10 1920 Explorer.EXE 24 0 0.1 1.14 0x7ffa519f0000 1372160 GDI32.dll C:\Windows\system32\

11 1920 Explorer.EXE 19 0 0.1 1.09 0x7ffa50ed0000 1134592 KERNELBASE.dll C:\Windows\system32\

12 1920 Explorer.EXE 18 0 0.1 1.07 0x7ffa51de0000 790528 OLEAUT32.dll C:\Windows\system32\

13 1920 Explorer.EXE 14 0 0.08 0.86 0x7ffa4d290000 1568768 PROPSYS.dll C:\Windows\SYSTEM32\

14 1920 Explorer.EXE 17 1 0.07 0.8 0xf801ba869000 2064384 Ntfs.sys C:\Windows\System32\Drivers\

15 1920 Explorer.EXE 16 0 0.07 0.77 0x7ffa35cf0000 4505600 UIRibbon.dll C:\Windows\system32\

16 1864 CodeXLPowerPr 18 1 0.05 0.52 0xf8032ca0e000 7913472 ntoskrnl.exe C:\Windows\system32\

17 1920 Explorer.EXE 14 0 0.04 0.49 0x7ffa51b40000 1310720 RPCRT4.dll C:\Windows\system32\

18 4 System 20 1 0.04 0.44 0xf8032ca0e000 7913472 ntoskrnl.exe C:\Windows\system32\

19 1864 CodeXLPowerPr 8 0 0.03 0.38 0x617f0000 425984 CXLCpuProfilingC:\Program Files (x86)\CodeXL\

20 1920 Explorer.EXE 7 1 0.03 0.37 0xf801ba700000 376832 fltmgr.sys C:\Windows\system32\drivers\

21 404 svchost.exe 20 0 0.03 0.37 0x0 0 unknown name unknown path

22 688 svchost.exe 16 0 0.03 0.37 0x0 0 unknown name unknown path

23 1920 Explorer.EXE 7 0 0.03 0.36 0x7ffa4c440000 3137536 explorerframe.dC:\Windows\system32\

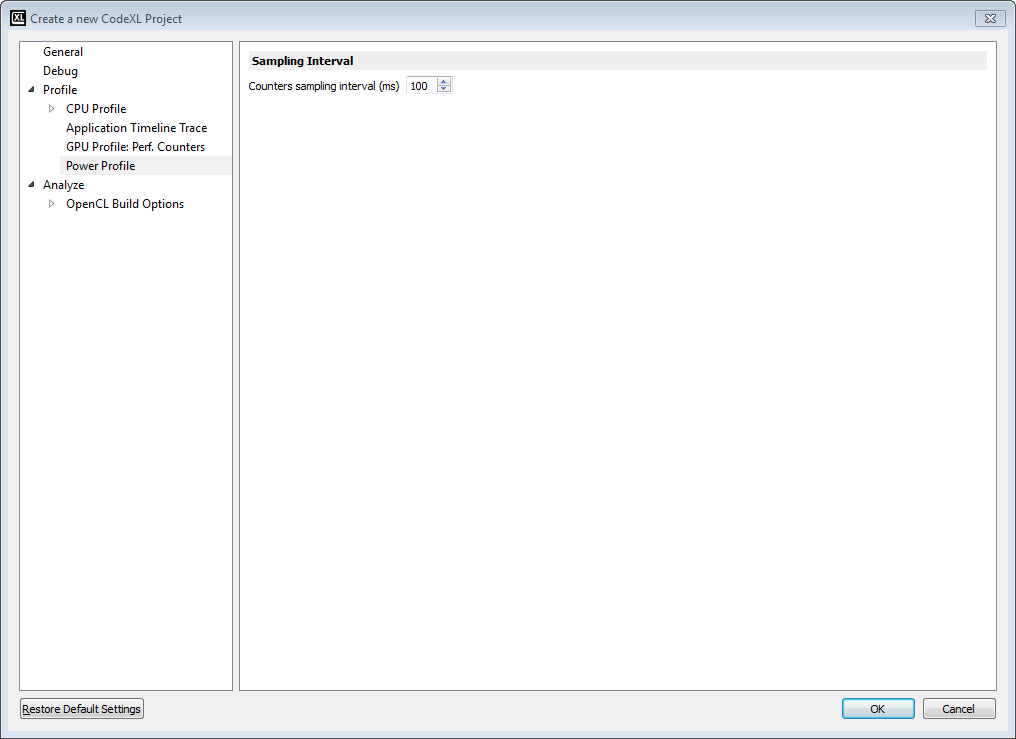
24 828 dwm.exe 20 1 0.03 0.35 0xf960000fa000 5783552 win32k.sys C:\Windows\System32\

### Power Profiler Project Settings

After switching to Profile mode, the Power Profiler Project Settings are accessible by any of these methods:

* In the Standalone application
  + Pressing CTRL + P
  + From the Profile menu, selecting “Profile Settings…”
* In Visual Studio
  + From the CodeXL menu, selecting “CodeXL Project Settings…”

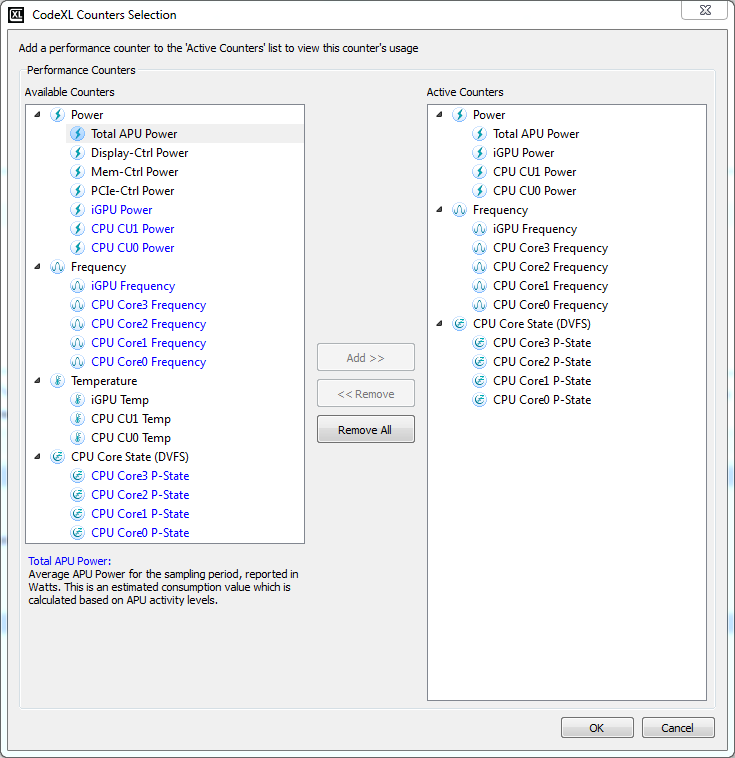
The CodeXL Project Settings dialog appears. Navigate to the Power Profile sub-node under the Profile node in the options tree on the left.



The “Counters sampling interval (ms)” field sets the amount of time between every two consecutive sample operations of the profiler. The units used in this field are milliseconds. Setting an interval smaller than 100 milliseconds is discouraged as the overhead of the frequent sampling may skew the results and the design of the counters mechanism is not optimal for higher sample rates.

### Power Profiler Counters Selection Dialog

The counter selection dialog allows you to choose which performance counters will be monitored in the next power profiling session. The available counters are divided into 4 categories: Power, Frequency, Temperature, and CPU Core State. At the right side of the dialog window you will find the list of Active Counters. These are the counters which will be monitored in the next power profiling session. To get the description of a specific counter, click on it in the Available Counters pane at the left side of the dialog window.

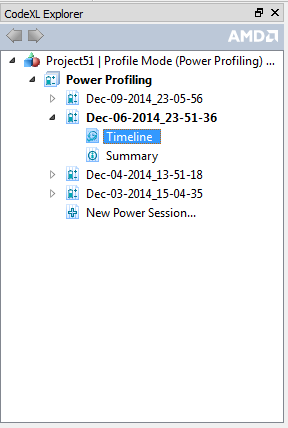


Note: The Total APU Power counter is enabled by default, and cannot be deactivated. Except for the Total APU Power counter, the activation of all other counters is optional.

### Power Profiler Session Tree

Navigating and displaying power profiling sessions is performed using the CodeXL Explorer tree, same as for other types of CodeXL profiling sessions. Each power profiling session is represented by a node in the tree, containing two sub-nodes for the Timeline and Summary views. Double clicking a node opens its respective view in the MDI space.

A session can be imported into the CodeXL project by right-clicking on the Power Profiling node and selecting ‘Import Session’. The dialog that opens allows you to navigate to the location of the session to import, and displays a filtered list of files that correspond to the \*.clxdb naming format.

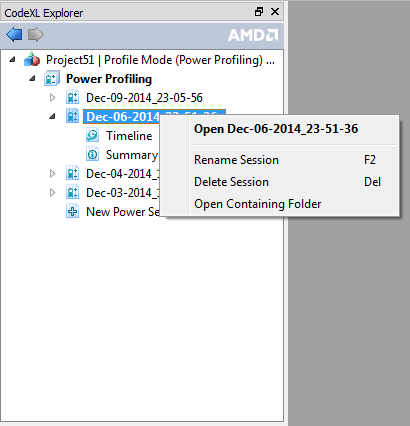


By clicking the “New Power Session…” node a new session is created, but sampling of data does not yet begin. The Timeline and Summary view tab will appear, and you can select which counters will be collected during the profile session by either

* Double-clicking the last line in the timeline charts legend table “Double-click to add or remove counters…”
* Selecting the “Select Power Profiling Counters…” command from the menu. This command is located in the standalone CodeXL app’s Profile menu, and in the Visual Studio CodeXL menu.

To begin the sampling of data in the new session, click the “Start Profiling” toolbar button, or select “Start Profiling” from the Profile menu (standalone CodeXL) or CodeXL menu in Visual Studio.

Right-click on a session node in the explorer tree will display the session context menu which is identical to the context menu for other profile session types.



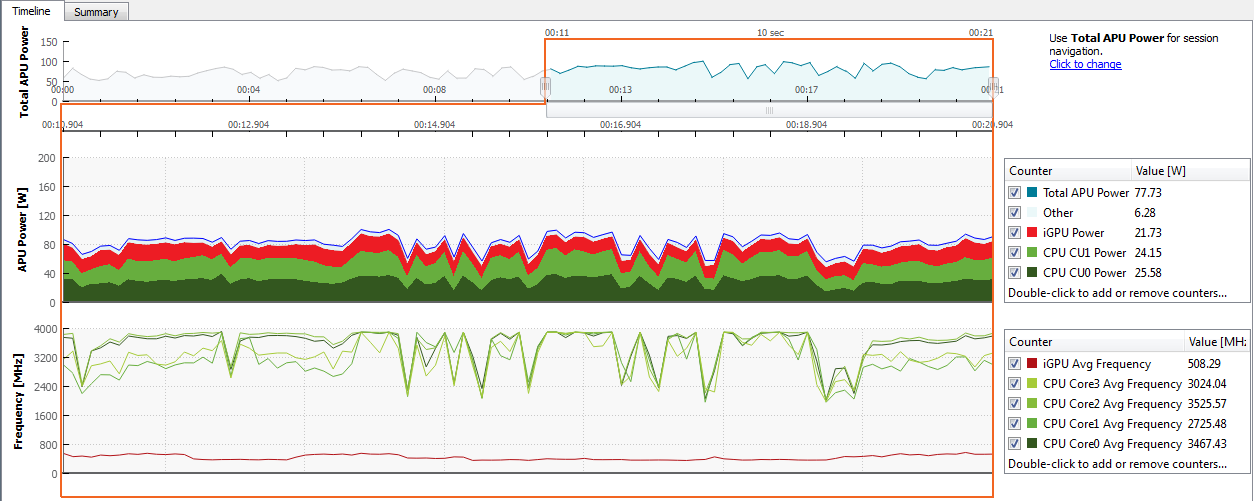
### Power Profiler Timeline View

The Power Profiler Timeline View displays the measured values of the activated counters throughout the session. The horizontal axis of all charts represents the time that has passed since the beginning of the session. During profiling sessions, all of the charts in the Timeline View are being updated in real-time with the measured values which are streaming in. The uppermost ribbon (titled “Total APU Power”) displays the overall power consumption of the APU throughout the session.

The top chart has an adjustable range slider that controls the display of all the other timeline charts. By performing such actions as dragging the slider sideways, extending or retracting it, you set the scope of attention and the focus of the timeline charts. Each of the charts below displays only the data that was collected in the time range corresponding to the slider’s position and length. That is, the data in all timeline charts, except for the Total APU Power chart itself, is dictated by the time range which is selected by the Total APU Power chart’s range slider.

Below the Total APU Power ribbon, you will find additional ribbons containing more graphs, according to the set of activated counters: A Power chart which displays the power consumed by specific APU components (such as CPU cores or integrated GPU), a Frequency chart which displays the frequency of the selected components, a Temperature chart which displays the thermal trend of the selected components, and a CPU State chart which displays the CPU core states. The APU Power graph is always displayed, since the Total APU Power counter is activated by default and cannot be deactivated. The other charts (frequency, temperature and CPU core state) are optional, and will only be displayed if the relevant counters were activated.

To the right side of each chart you will find a legend that displays the measured values at a specific point in time. To change the point in time for which the values are displayed, reposition the mouse cursor horizontally on one of the graphs. The list of counters in the legends is customizable, and specific counters can be removed/added between profile sessions.



### Power Profiler Summary View

The Power Profiler Summary View displays an analysis of the values measured throughout the session. Similarly to the Timeline View, this view is updated in real-time when power profiling sessions are running. At the upper-left side of the summary view, you can see the session duration which is the amount of time that the profiling session was in progress. The uppermost chart (titled “Power”) shows the Total APU Power/Energy consumption. This chart has two modes: Cumulative and Average. The Cumulative mode displays the cumulative energy consumed by the APU components, measured in Joules. The Average mode displays the average power consumption of the APU components, measured in Watts. For each of these two modes, the Total APU Power Consumption value at the top of the summary view represents the value consumed by the whole APU.

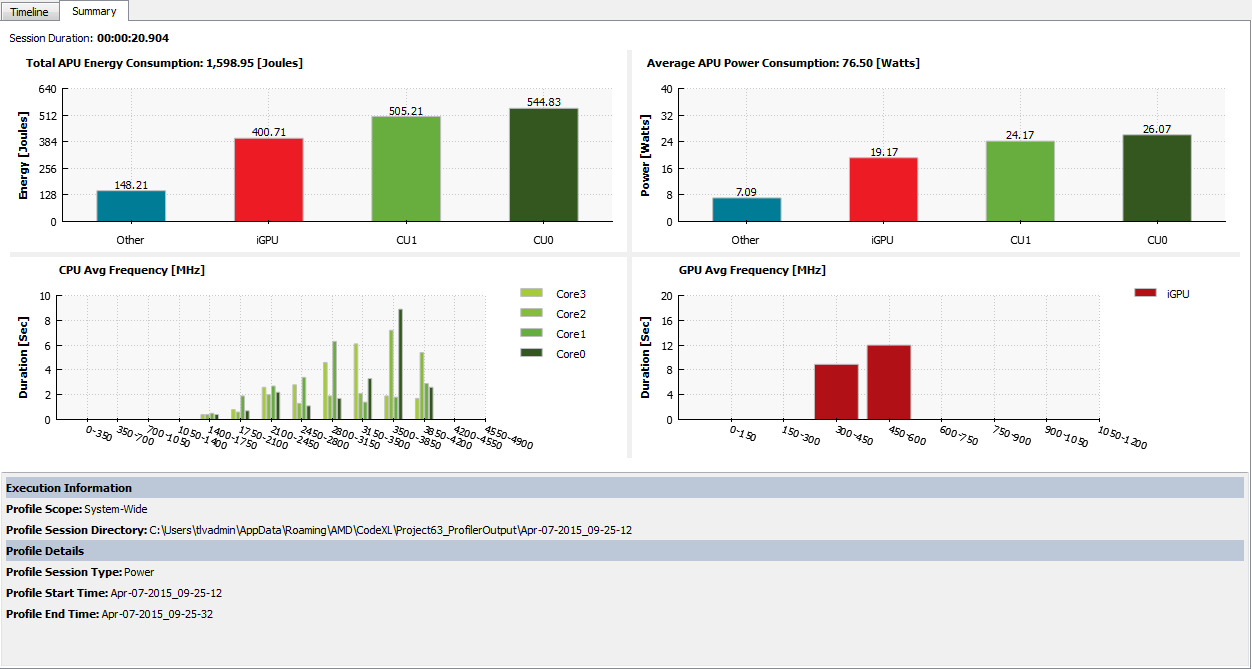
If CPU or GPU frequency counters were activated for the session, you will find additional histogram graphs below the Power graph in the Summary View:

CPU Frequency Graph

This stacked histogram displays for each CPU core how much time it spent at each frequencies range.

GPU Frequency Graph

This histogram displays how much time the GPU spent at each frequencies range.



At the bottom of the Summary View you can find useful information about the current session:

* **Target Path:** The target application’s path. In system-wide sessions, this field will be left empty.
* **Working directory:** the target application’s working directory. In system-wide sessions, this field will be left empty.
* **Profile Session Directory:** the directory where CodeXL stored the session file.
* **Command Line Arguments:** the target application’s command line arguments (if any). In system-wide sessions, this field will be left empty.
* **Environment Variables:** the target application’s environment variables (if any). In system-wide sessions, this field will be left empty.
* **Profile Scope:** the scope of the current session (either system-wide, or system-wide with focus on an application).
* **Profile Session Type:** the type of this profiling session. This field will always be filled with the “Power” value for power profiling sessions.
* **Profile Start Time:** the time when the current session began.
* **Profile End Time:** the time when the current session ended.

### Power Profiler Session Control

The control of power profiling sessions is very simple:

* To start a power profiling session, open the Counters Selection Dialog (in the CodeXL menu, click on Profile->Select Power Profiling Counters…) and select the counters which will be monitored during the session. Click OK, then click the ‘Start’ button on CodeXL’s session control toolbar.
* To stop a power profiling session, click the ‘Stop’ button on CodeXL’s session control toolbar.

### Remote Power Profiling

For a detailed description of how to run a power profiling session on a remote machine, consult the Remote GPU Profiling, Power Profiling and Debugging section.

### AMDTPowerProfileAPI Library

The CodeXL Power Profile API library is useful to analyze the energy efficiency of systems based on AMD CPUs, APUs and dGPUs (Discrete GPU). These APIs provide interface to read the power, thermal and frequency characteristics of APU/dGPU and their subcomponents.

These APIs are targeted for software developers who want to write their own application to sample the power counters based on their specific use case.

For detail information of these APIs, refer **AMDTPowerProfileAPI.chm** (on Windows platforms) or **AMDTPowerProfileAPI.pdf** (on Linux based platforms) file in the **SDK/AMDTPowerProfile/doc** subdirectory under the CodeXL installation directory.

AMDTPowerProfileAPI shared library has dependencies on AMDTBaseTools and AMDTOSWrappers shared libraries, so the corresponding .DLL (on Windows system) and .SO (on Linux system) should be added.

To build and execute a test application (test.cpp) following steps should be performed on Linux machine.

1. Assuming test.cpp is located at in /home/<user-dir>/samples

$ cd /home/<user-dir>/samples

1. Set LD\_LIBRARY\_PATH

$ export LD\_LIBRARY\_PATH=<codexl-install-dir>

1. Compile application code

$ g++ test.cpp –I<codexl-install-dir>/SDK/AMDTPowerProfile/inc –L<codexl-install-dir>/SDK/AMDTPowerProfile/bin/x86\_64 –lAMDTPowerProfileAPI -L<codexl-install-dir> -lAMDTOSWrappers -lAMDTBaseTools -o test

1. Execute

$ ./test

**Note:** When using the static AMDTPwrProfile library for the Power Profiler API on Linux, the user must build his application with “-Wl,--whole-archive -lpthread -Wl,--no-whole-archive”. Otherwise not all the symbols from pthread library will be linked, since most of them are WEAK symbols. Failing to use these flags will lead to a crash. (1040)

### Limitations

* Multiple instance of CodeXL Power Profiler cannot be run simultaneously. If the CodeXL graphic client is conducting a power profile session then no other instance of the graphic client can perform a power profile and the command line tool cannot be run either, until the original profile session ends.
* ICELAND discrete GPU(Topaz-XT, Topaz PRO, Topaz XTL, Topaz LE) series is not support in 2.0 release.
* Please make sure you have latest catalyst driver installed before running power profiler. Newer version of discrete GPU may go to sleep (low power) state frequently if there is no activity in that GPU. In that case, power profiler may emits a warning AMDT\_WARN\_SMU\_DISABLED. Counters may not be accessible during this state. It is advisable to bring discrete GPU to active state by running some openCL or openGL application, then run power profiling on that GPU.
* Process profiling is supported only with command line tool. If PMC (Performance Monitoring Counters) counters are not accessible and unable to calculate the IPC load, then compute unit power is distribute equally to each core. In that case power distribution to each process may not be accurate.
* Module profiling is supported only with command line tool. If PMC (Performance Monitoring Counters) counters are not accessible and unable to calculate the IPC load, then compute unit power is distribute equally to each core. In that case power distribution to each process may not be accurate. Modules which are started after the process run are not considered in this release.