FrameView

Integrated Frame Benchmarking & Power Tool

USER GUIDE

INTRODUCTION **4**
What's New in FrameView 1.2 4
FrameView Interface & Settings 5
Installing FrameView 7
Running FrameView 8
FrameView Overlay 10
Overlay Metrics 11
Overlay Mode Tags 12
FRAMEVIEW FILES **13**
FrameView Summary File 13
FrameView Log File 15
LAPTOP BATTERY LIFE **18**
Testing Battery Life 18
Setup 18
Capturing Battery Data 21
Plotting Battery Life 21
FRAMEVIEW ANALYSIS TEMPLATE **25**
Using the Template 25
Getting Started 25
Capture Properties and Label Templates 25
Run Summary 27
Capture Health 28
Run Summary Bar Plots 30
Individual Capture Plots 30
Categorical Comparison Plots 31
Charting Performance Data 32
Charting Percentile Data 32
Charting NVIDIA Power Data 33
Charting AMD Power Data 34
Power Capture Analysis Tool (PCAT) 35

HOW FRAMEVIEW WORKS **36**

Frame Rendering Pipeline 36

TROUBLESHOOTING **37**

FrameView 37

Excel Analysis Template 38

INTRODUCTION

FrameView is a software tool designed to capture and measure performance and power utilization of PC-based graphics and CPU hardware. It's especially useful for measuring frame rates and GPU power usage when running stressful "real world" gaming scenarios. FrameView captures performance and power data with minimal overhead so as not to impact frame rates or gameplay and includes an overlay that shows performance and power metrics as a game is being played. It also allows benchmark runs to be captured and charted in detailed reports.

It's no surprise that battery life is the most important factor to determining the practicability of gaming and running other power hungry applications on a laptop. Regardless of whether you plan to unplug or not, you will eventually find yourself in a situation without a power outlet, and it's important to know how your device will perform in these situations.

Introduced with the latest version 1.2, **FrameView is the first tool to correlate frame time performance and latency with battery life metrics on laptops.** FrameView gives you the power to test and compare battery life on various performing laptops to see which is best suited for unplugged gamers. Click **here** to see our recommended system settings for gaming on battery.

➢ **Download FrameView 1.2 at nvidia.com**

What's New in FrameView 1.2

- **Laptop Battery Drain Metrics**
  - Battery Percentage
  - Battery Drain Rate(W)
  - Current Battery Capacity(Wh)
  - Total Battery Capacity(Wh)
- **GPU Memory Clock Frequency**
- **DPI Aware Overlay**

FrameView captures game performance metrics including average and percentile frame-per-second (FPS) data for both

API SUPPORT
DirectX APIs (versions 9-12), OpenGL, Vulkan

SINGLE-GPU CONFIGURATIONS
NVIDIA GeForce, AMD, Intel

MULTI-GPU CONFIGURATIONS
NVIDIA SLI, AMD Crossfire, MSHybrid- and Optimus-based platforms

DISPLAY SUPPORT
G-SYNC, Non-G-SYNC, Adaptive SYNC (including FreeSync)

SCREEN MODES
Full Screen, Windowed, UWP apps

OS SUPPORT
Windows 10 and above

OVERLAY
Average rendered FPS, 90/95/99th frame percentile, dropped frames, render present latency, perf/watt for NVIDIA GPUs, CPU & GPU utilization/core speed/temp, display mode tags, and DPI aware overlay

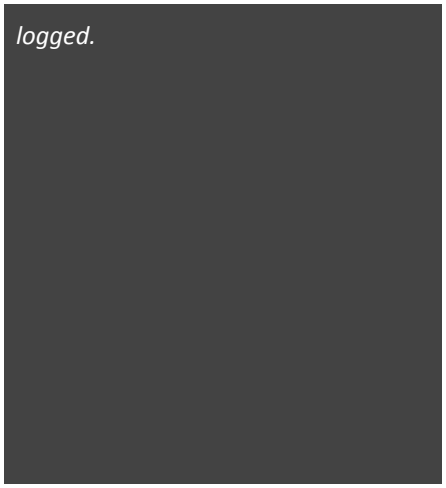
NOTE: *FrameView overlay is DPI aware: the overlay stats will scale based on the DPI of the display to allow users to view overlay stats more easily on high resolution displays. Please note that FrameView may need to be restarted when monitor settings are changed in order for the proper DPI scaling to take effect.*

LOGGING
Average rendered FPS, 90/95/99th frame percentiles, HW/SW info, API tested, full GPU & CPU frequency/power/utilization stats, perf/watt data, laptop battery percentage, laptop battery drain rate, and much more!

NOTE: *DX9/DX10 games do not have overlay support although data capture is supported and properly*

single- and multi-GPU configurations. Percentile FPS data is valuable for illustrating the severity and frequency of stutters that can interrupt gameplay. FrameView has been optimized particularly for detailed frame time, present, and display scheduling metrics for measuring stutter.

FrameView also captures real-time power measurements for both total board power (including graphics memory) and GPU chip-only power through application programming interfaces (APIs), which is publicly-available software that communicates with the hardware and returns data.

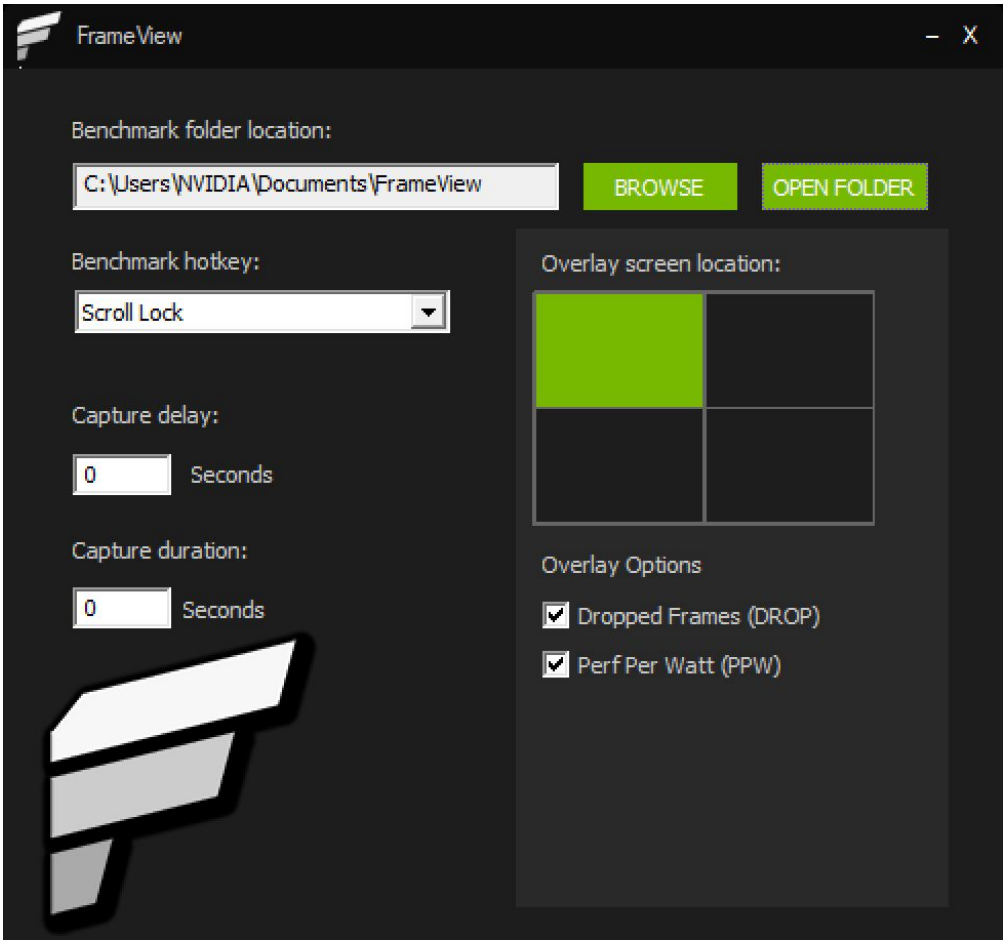


FrameView Interface & Settings

This section outlines the functionality of the settings provided in the FrameView interface.

FrameView captures game performance metrics including average and percentile frame-per-second (FPS) data for both single- and multi-GPU configurations. Percentile FPS data is valuable for illustrating the severity and frequency of stutters that can interrupt gameplay. FrameView has been optimized particularly for detailed frame time, present, and display scheduling metrics for measuring stutter.

FrameView also captures real-time power measurements for both total board power (including graphics memory) and GPU chip-only power through application programming interfaces (APIs), which is publicly-available software that communicates with the hardware and returns data.



Benchmark folder location

This is where the benchmark logs will be saved. Use the Browse button to choose a location and the Open Folder button to access saved results in Windows File Explorer.

Benchmark hotkey

This is the button assigned to start and stop the benchmarking process. At this time, FrameView only supports **Scroll Lock** and **F10** as the benchmarking hotkeys.

Capture delay

This will delay the capture of a game by the seconds specified in the window. The default is 0 seconds.

Capture duration

This will set a capture time limit for the benchmark. The default is 0 seconds, which means the benchmark capture logging must be manually started and stopped with the hotkey. When the time limit is set to a number greater than 0, the benchmark logging must still be manually started, but it will be automatically stopped after the specified capture duration.

Overlay screen location

FPS, percentiles, and power information will be displayed by default in the upper-left corner of your monitor when running a game. To change the overlay location, click a different quadrant in the FrameView interface, represented by green blocks. More information about the overlay can be found in the **FrameView Overlay** section.

> **NOTE**: The overlay is automatically disabled during benchmarking to ensure more accurate results. The overlay will return once the benchmark hotkey is pressed a second time.

Overlay Options

Selecting these will show real-time power reporting. Please refer to the **Charting NVIDIA Power Data** section for more information on how power data is measured and reported.

Dropped Frames (DROP)

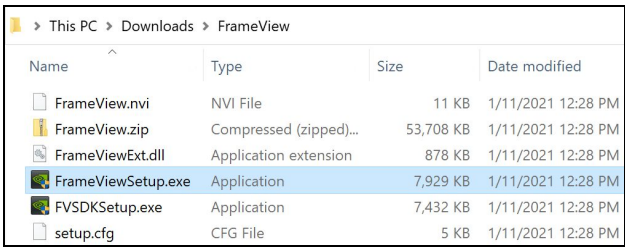
When enabled, FrameView will measure and report whether the presented frame was dropped (1) or displayed (0). This value is reported in the FrameView Log under the ***Dropped*** header.

Perf Per Watt (PPW)

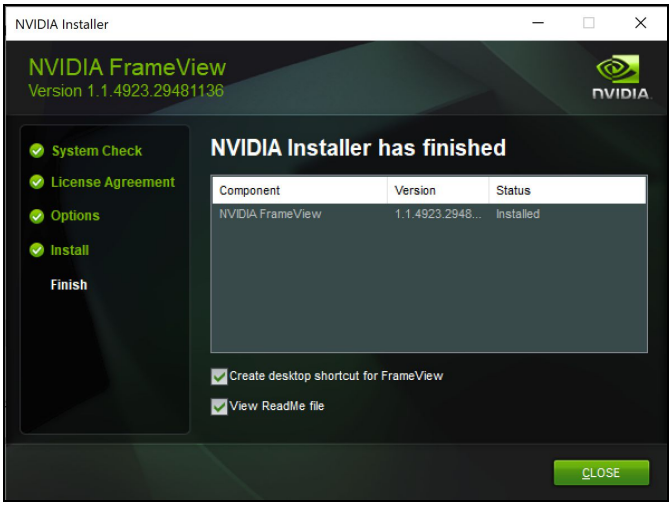
When enabled, FrameView will measure and report performance-per-watt (PPW) data for GPUs. This value is reported in the FrameView Log under the ***Perf/W Total(F/J) (API)*** and ***Perf/W GPUOnly(F/J) (API)*** headers for NVIDIA GPUs. To achieve PPW on AMD GPUs, PCAT must be utilized. AMD API power, shown as ***AMDPwr(W) (API)*** appears to report a value in-between chip-only and full board power.

Installing FrameView

FrameView version 1.2 includes a system installer. The FrameView install files include the following:



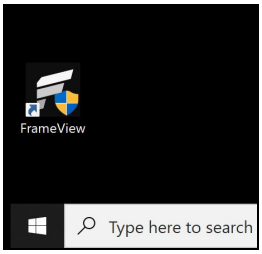
Run **FrameViewSetup.exe** to initiate the



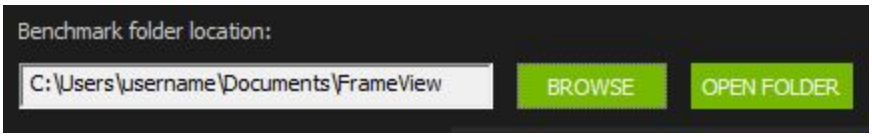
installation process.

Running FrameView

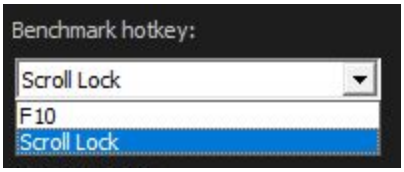
1. **Launch FrameView** using the desktop shortcut.



2. **Click Browse** to determine the **Benchmark folder location** where results will be stored.



3. At this time, FrameView supports **Scroll Lock** and **F10** as the benchmarking hotkeys.



4. **Launch a game.**
   The **FrameView overlay** should show up in the designated area chosen in the FrameView interface.

5. **Press** the benchmarking hotkey (default is **Scroll Lock**) to begin benchmarking.
   The overlay will disappear during data collection to reduce overhead in the captured data.

6. **Press** the benchmarking hotkey again to stop data collection.
   The overlay will reappear in the designated area.

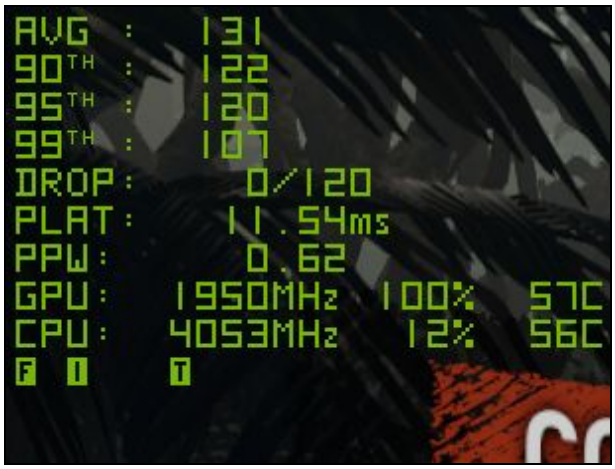
7. **Exit the game** and return to FrameView.
   Click the **Open Folder** button to view benchmark results.

8. **FrameView Results** will be saved as **.CSV files** with an application and timestamp name. Consider renaming the files or creating a directory to reflect the GPU, game, and settings tested.

FrameView Overlay

The overlay will appear in games with proper API support when FrameView is running in the background. If the overlay does not appear, make sure to check if FrameView is running. Adjust the overlay screen location in the FrameView settings to move the overlay to a different corner of the screen.

When benchmarking is enabled through the hotkey, the overlay will disappear. Removing the overlay reduces overhead to ensure a more accurate capture of the game data. The overlay will reappear when the hotkey is pressed again and capture is stopped or capture duration expires.



FrameView Overlay when using NVIDIA GeForce RTX 3070

FrameView Overlay when using AMD Radeon RX 6800 XT

NOTE: PPW is labeled N/A on AMD. They appear to report a value in-between chip-only and full board power. *However, PPW will be shown for AMD if PCAT is utilized.*

The overlay displays real-time data for average FPS, percentile FPS (90/95/99), dropped frames, render present latency, perf-per-watt, CPU/GPU utilization, core speed, and temperature. Please refer to the **FRAMEVIEW FILES** section for more information.

NOTE: Overlay information will not be shown in DX9/10 games. However, data capture is supported and the information will be properly logged.

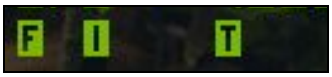
Overlay Metrics

LABEL	METRICS	DESCRIPTION
AVG	**Rendered Avg FPS**	Uses Rendered FPS scheduling metrics to show the overall average FPS.
90th	**Rendered 90th %**	Uses rendered FPS scheduling metrics to show 90th percentile data 10 frames out of 100 are slower than this frame rate. 90% of the frames will achieve at least this frame rate.
95th	**Rendered 95th %**	Uses rendered FPS scheduling metrics to show 95th percentile data 5 frames out of 100 are slower than this frame rate. 95% of the frames will achieve at least this frame rate.
99th	**Rendered 99th %**	Uses Rendered FPS scheduling metrics to show 99th percentile data 1 frame out of 100 is slower than this frame rate. 99% of the frames will achieve at least this frame rate.
DROP	**Dropped Frames**	Whether the present was dropped (1) or displayed (0).
PLAT	**Render Present Latency**	The time when the present call entered the queue to the time the present call was executed on the GPU.
PPW	**Perf Per Watt**	Performance per Watt measured as frames/joule (F/J) and described in more detail below. Considering MsBetweenPresents for performance and NV-Total-USBCPwr(W) (API). ***NOTE:*** *PCAT value is used if connected and running in the background. Otherwise, NVAPI power is used for NVIDIA. PPW is only shown for AMD boards if PCAT is used.*
GPU	**GPU Frequency**	GPU Frequency (MHz)
	GPU % Utilization	GPU utilization (percentage)
	GPU Temperature	GPU Temperature (Celcius)
CPU	**CPU Frequency**	CPU Frequency (MHz)
	CPU Utilization	CPU utilization (percentage)

	CPU Temperature	CPU Temperature (Celcius)

Overlay Mode Tags

The overlay also includes three mode tags. These letters are used to provide information about game settings that can impact overlay data reporting and data captures.




F (Full Screen)

Running the game at full screen will ensure that accurate performance results are measured at the resolution specified in the game.

I (Independent flip)

iFlip (also called Independent flip), is the mode where the app is simulating as if it was running in Full Screen Exclusive mode.

T (Tearing)

When Vertical Sync is disabled, the full performance of the game can be measured (due to higher frame rates beyond the refresh rate of the monitor). However, a major artifact of disabling Vertical Sync is tearing. This is the optimal method for testing game performance.

W (Windowed mode)

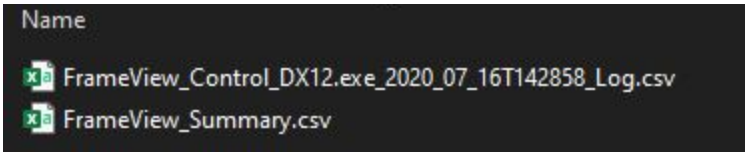
If the game runs in windowed mode, pressing the Alt+Enter keys on your keyboard while the game is running can often force the game into fullscreen mode. Check the game settings if that doesn’t work.

V (Vsync ON)

Vertical Sync is enabled, which forces the frame rate of the game to synchronize with the refresh rate of the monitor. However, this limits the frames that can be displayed, and will not show the full performance potential of the hardware being measured.

FRAMEVIEW FILES

Two files are created once the benchmarking is completed using the hotkeys. Both files are comma-separated value (CSV) files using UTF-8 encoding that can be imported into Microsoft Excel, OpenOffice Calc, or Google Sheets. These files are saved in the benchmark folder location that is specified in the FrameView user interface. They are named using the application process name and include date and time stamps.



FrameView Summary File

The smaller file is called the Summary and it contains the high-level data from all captured runs using the FrameView hotkey.

When opened, the file will look like this:

	A	B	C	D	E	F	G	H	I	J	K	L	M	N
1	TimeStamp	Application	Log Name	GPU0	GPU1	CPU	Resolution	Runtime	Avg FPS	Min FPS	Max FPS	90th %	95th %	99th %
2	2020-12-3(	WatchDogs	FrameView	NVIDIA Gef	NA	AMD Ryzer	2560x1440	D3D12	56.016	12.207	108.31	42.687	31.795	20.513
3	2020-12-3(	SOTTR.exe	FrameView	NVIDIA Gef	NA	AMD Ryzer	2560x1440	Other	83.022	11.518	315.487	67.349	65.277	55.925
4	2020-12-3(	csgo.exe	FrameView	NVIDIA Gef	NA	AMD Ryzer	2560x1440	D3D9	554.379	19.044	1109.139	324.444	268.025	218.083

You can duplicate this file and make your own custom table:

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P
1	Application	GPU0	CPU	Resolution	Runtim	Avg FPS	RenderPres	GPU0Clk	GPU0 Util	GPU0 Temp	GPU NV Pow	CPUClk	CPU Util	CPU Temp	CPU Package Pov	
2	WatchDogsl	NVIDIA	AMD Ryz	2560x1440	D3D12	56.016	24.187	2011.2	98.729	60.442	190	4438	17.975	57.741	88.65	
3	SOTTR.exe	NVIDIA	AMD Ryz	2560x1440	Other	83.022	16.386	1954.5	99.147	69.45	212	4251	18.305	59.379	85.38	
4	csgo.exe	NVIDIA	AMD Ryz	2560x1440	D3D9	554.38	0.495	2006.5	63.455	60.086	171	4488	9.04	70.726	82.98	
5	csgo.exe	NVIDIA	AMD Ryz	2560x1440	D3D9	589.48	0.475	1997.9	63.991	64.935	176	4492	9.306	71.285	83.29	
6	ACOdyssey.e	NVIDIA	AMD Ryz	2560x1440	D3D11	76.227	31.429	1995	97.546	68.868	196	4463	27.778	73.95	107.4	
7	DOOMEtern	NVIDIA	AMD Ryz	3840x2160	D3D11	112.87	14.72	1824.4	98.989	71.987	218	4141	16.074	65.955	87.36	
8	StrangeBrig	NVIDIA	AMD Ryz	3840x2160	Other	110.35	35.305	1832	100	71.515	218	3859	3.912	52.734	58.48	
9	StrangeBrig	NVIDIA	AMD Ryz	3840x2160	Other	109.93	35.427	1830.6	100	71.272	219	3859	4.105	52.769	58.49	
10	StrangeBrig	NVIDIA	AMD Ryz	3840x2160	Other	110.03	35.401	1830.5	100	71.281	218	3851	4.147	53.363	58.3	
11	WatchDogsl	AMD Ra	AMD Ryz	2560x1440	D3D12	37.231	44.808	2396.6	98.964	71.104	NA	4424	13.786	55.962	73.94	

This table explains each header and the data contained in it:

SUMMARY HEADERS	DESCRIPTION
TimeStamp	Date/time at end of logging
Application	Executable name that was captured
Log Name	Name associated with the captured Log file

GPU#	Retail GPU make/model name for first GPU.
CPU	Retail CPU make/model name.
Resolution	Output resolution of the game/application captured.
Runtime	The runtime used to present (e.g., D3D9, DXGI, VK, OGL).
Avg FPS	Uses rendered FPS scheduling metrics to show the overall average FPS.
Min FPS	Uses rendered FPS scheduling metrics to show the minimum (single lowest) FPS.
Max FPS	Uses rendered FPS scheduling metrics to show the maximum (single highest) FPS.
90th %	Uses rendered FPS scheduling metrics to show 90th percentile data. 10 frames out of 100 are slower than this frame rate. 90% of the frames will achieve at least this frame rate.
95th %	Uses rendered FPS scheduling metrics to show 95th percentile data. 5 frames out of 100 are slower than this frame rate. 95% of the frames will achieve at least this frame rate.
99th %	Uses rendered FPS scheduling metrics to show 99th percentile data. 1 frame out of 100 is slower than this frame rate. 99% of the frames will achieve at least this frame rate.
Time (ms)	The amount of time that comprises the capture.
Render Present Latency	The time when the present call entered the queue to the time the present call was executed on the GPU.
GPU#Clk(MHz)	GPU frequency (MHz) [# denotes GPU number]
GPU# Util %	GPU utilization (percentage) [# denotes GPU number]
GPU# Temp (C)	GPU temperature (Celcius) [# denotes GPU number]
Perf/Watt (F/J) (PCAT)	Performance per Watt considering MsBetweenPresents for performance and board power as measured by PCAT.
PCAT Power (Watts)	GPU board power as measured by PCAT.
GPU NV Power (Watts) (API)	NVIDIA GPU power as reported by NVAPI (same as PCAT power).
CPU# Util %	CPU utilization (percentage)
CPU# Temp (C)	CPU temperature (Celcius)
CPU# Freq (MHz)	CPU frequency (MHz)
CPU Package Power (Watts)	Total CPU power.
OS	Operating System Info - taken from WMI (Windows Management Instrumentation)
GPU Base Driver	Base Driver Version - taken from WMI (Windows Management Instrumentation)

GPU Driver Package	Driver Package - taken from WMI (Windows Management Instrumentation)
System RAM	Size and type of System RAM - taken from WMI (Windows Management Instrumentation)
Motherboard	Motherboard make/model - taken from WMI (Windows Management Instrumentation)

FrameView Log File

The larger file is called the Log, and it contains all of the data from a captured run using the FrameView hotkey, including laptop battery life metrics. Frame metadata like the resolution, runtime, flip model, and more are captured for each frame while power, temperature, frequency, and utilization are collected for the GPU(s) and the CPU at regular intervals.

When opened, the file will look like this:

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
1	Application	GPU	CPU	Resolution	Runtime	AllowsTear	ProcessID	SwapChain	SyncInterva	PresentFlag	PresentMo	Dropped	TimeInSecc	MsBetweer	MsBetweer
2	WatchDogs	NVIDIA Gef	AMD Ryzer	2560x1440	D3D12	1	12056	0x000001C	0	0	Hardware (	0	830.0762	14.661	14.671
3	WatchDogs	NVIDIA Gef	AMD Ryzer	2560x1440	D3D12	1	12056	0x000001C	0	0	Hardware (	0	830.089	12.823	14.138
4	WatchDogs	NVIDIA Gef	AMD Ryzer	2560x1440	D3D12	1	12056	0x000001C	0	0	Hardware (	0	830.1039	14.899	14.474
5	WatchDogs	NVIDIA Gef	AMD Ryzer	2560x1440	D3D12	1	12056	0x000001C	0	0	Hardware (	0	830.1179	14.051	14.791
6	WatchDogs	NVIDIA Gef	AMD Ryzer	2560x1440	D3D12	1	12056	0x000001C	0	0	Hardware (	0	830.1333	15.309	14.557

This table explains each header, if the data is collected per frame or sampled on a regular interval, and how to interpret the column as a whole:

LOG HEADER	COLLECTION	DESCRIPTION
Application	Per Frame	The name of the process that called Present (if known)
GPU	Per Frame	Retail GPU make/model name
CPU	Per Frame	Retail CPU make/model name
Resolution	Per Frame	Output resolution of the game/application captured
Runtime	Per Frame	The runtime used to present (e.g., D3D9, DXGI, VK, OGL)
AllowsTearing	Per Frame	Whether tearing possible (1) or not (0)
ProcessID	Per Frame	The process ID of the process that called Present
SwapChainAddress	Per Frame	The address of the swap chain that was presented into
SyncInterval	Per Frame	Sync interval used in the Present call
PresentFlags	Per Frame	Flags used in the Present call
PresentMode	Per Frame	Flip model used for this Present
Dropped	Per Frame	Whether the frame was dropped (1) or displayed (0); if dropped, MsUntilDisplayed will be 0

TimeInSeconds	Per Frame	The time of the Present call, measured from when FrameView recording started in seconds
MsBetweenPresents	Per Frame	The time between this Present call and the previous one, in milliseconds
MsBetweenDisplayChange	Per Frame	The time between when the previous frame was displayed and this frame was, in milliseconds
MsInPresentAPI	Per Frame	The time spent inside the Present call, in milliseconds
MsRenderPresentLatency	Per Frame	The time when the present call entered the queue to the time the present call was executed on the GPU.
MsUntilDisplayed	Per Frame	The time between the Present call (TimeInSeconds) and when the frame was displayed, in milliseconds
Render Queue Depth	Per Frame	Maximum pre-rendered frames
GPU#Clk(MHz)	Sampled	GPU0 frequency (MHz) [# denotes GPU number]
GPU#Util(%)	Sampled	GPU0 utilization (percentage) [# denotes GPU number]
GPU#Temp(C)	Sampled	GPU0 temperature (Celcius) [# denotes GPU number]
GPU#MemClk(MHz)	Sampled	GPU0 memory clock (MHz) [# denotes GPU number]
PCAT Power Total(W)	Sampled	GPU board power as measured by PCAT
Perf/W Total(F/J) (PCAT)	Sampled	Performance per Watt considering MsBetweenPresents for performance and board power as measured by PCAT
Perf/W Total(F/J) (API)	Sampled	Performance per Watt considering MsBetweenPresents for performance and board power
Perf/W GPUOnly(F/J) (API)	Sampled	Performance per Watt considering MsBetweenPresents for performance and GPU/Chip/ASIC power
Perf/W Total-USBC(F/J) (API)	Sampled	Performance per Watt considering MsBetweenPresents for performance and board power excluding USB-C
GPUOnlyPwr(W) (API)	Sampled	GPU/Chip/ASIC power, post-regulator
NV-Total-USBCPwr(W) (API)	Sampled	Board power excluding USB-C
NV Pwr(W) (API)	Sampled	Board power
AMDPwr(W) (API)	Sampled	Board power **NOTE**: The AMD API used by FrameView appears to report a value in-between chip power and board power for AMD graphics cards. Therefore it's currently not possible to use FrameView to directly compare AMD GPU power to NVIDIA GPU power.
CPUClk(MHz)	Sampled	Average CPU frequency (MHz)
CPUUtil(%)	Sampled	CPU utilization (percentage)
CPU Package Temp(C)	Sampled	Overall CPU temperature (Celsius)
CPU Package Power(W)	Sampled	Total CPU power

CPU TDP (W)	Sampled	CPU thermal design power
CPUCoreUtil%[##]	Sampled	CPU utilization (percentage) [## denotes CPU core number]
Current Battery Capacity(Wh)	Sampled	Remaining battery life in watt-hours.
Total Battery Capacity(Wh)	Sampled	Maximum battery capacity in watt-hours.
Battery Percentage	Sampled	Remaining battery life as a percentage.
Battery Drain Rate(W)	Sampled	Current battery drain rate in watts.

LAPTOP BATTERY LIFE

It's no surprise that battery life is the most important factor to determining the practicability of gaming and running other power hungry applications on a laptop. Regardless of whether you plan to unplug or not, you will eventually find yourself in a situation without a power outlet, and it's important to know how your device will perform in these situations.

Introduced with the latest version 1.2, **FrameView is the first tool to correlate frame time performance and latency with battery life metrics on laptops.** FrameView gives you the power to test and compare battery life on various performing laptops to see which is best suited for unplugged gamers.

FrameView reports the following values in the FrameView log file:

Current Battery Capacity(Wh)	Remaining battery life in watt-hours.
Total Battery Capacity(Wh)	Maximum battery capacity in watt-hours.
Battery Percentage	Remaining battery life as a percentage.
Battery Drain Rate(W)	Current battery drain rate in watts.

Testing Battery Life

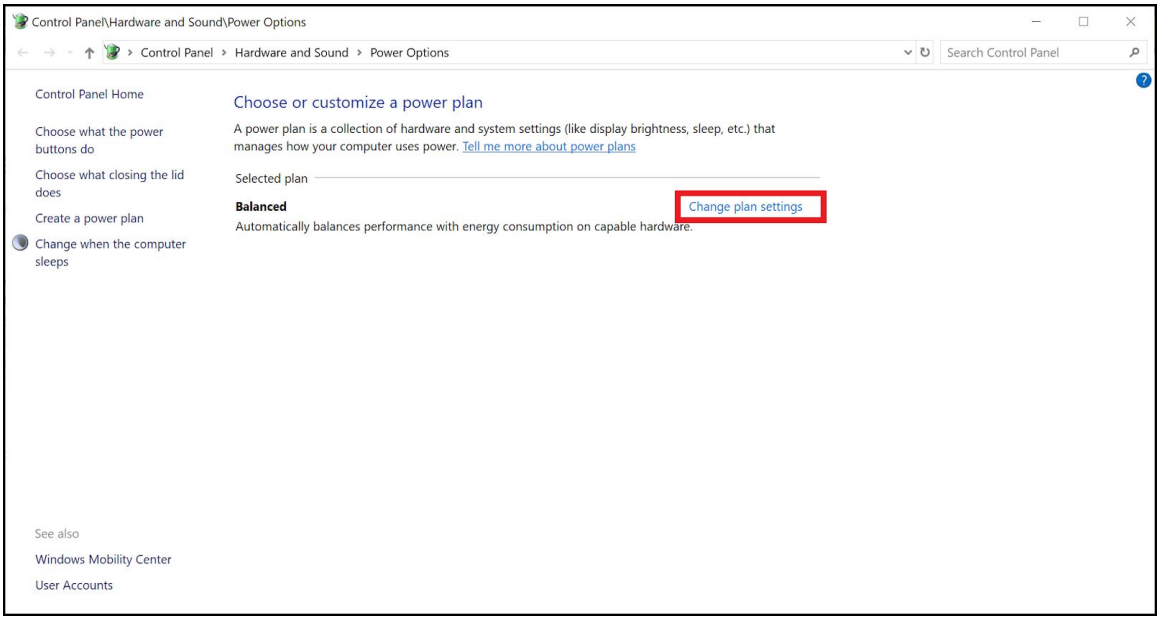
Setup

Close all non-essential applications in the background and note the remaining applications in use. It's recommended to choose the default/balanced Windows power plan that ships with the laptop.

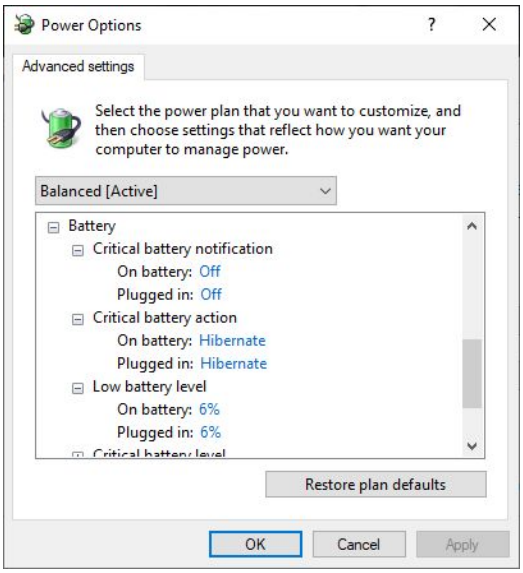
You can verify the current Windows power plan by navigating to Control Panel → Hardware and Sound → Power Options. Click **Change Plan Settings** to modify other settings. See our recommended list of changes below.

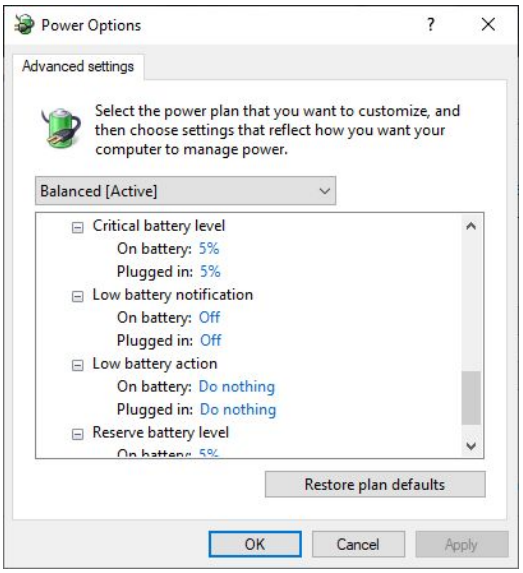
Recommended Power Settings for Gaming on Battery

1. **Windows Power Plan** (Control Panel → Hardware and Sound → Power Options)
    a. Windows Power Plan: **Balanced**

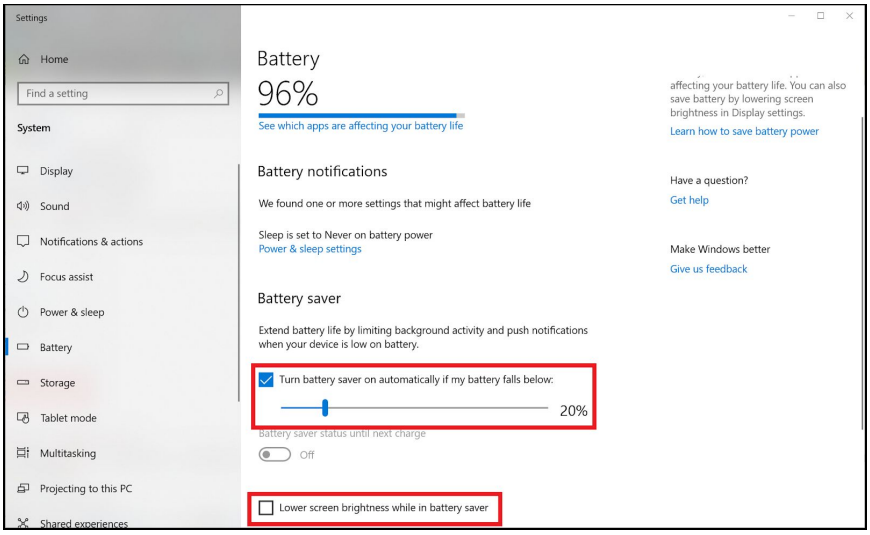


2. **Advanced Power Settings** (Control Panel → Hardware and Sound → Power Options → Change Plan Settings)
    a. Critical Battery Action: **Hibernate**
    b. Low Battery Level: **6%**
    c. Critical Battery Level: **5%**
    d. Low Battery Notification: **Off**
    e. Low Battery Action: **Do Nothing**
    f. Reserve Battery Level: **5%**
    g. Adaptive Brightness: **Off**
    h. Wi-Fi: **On**

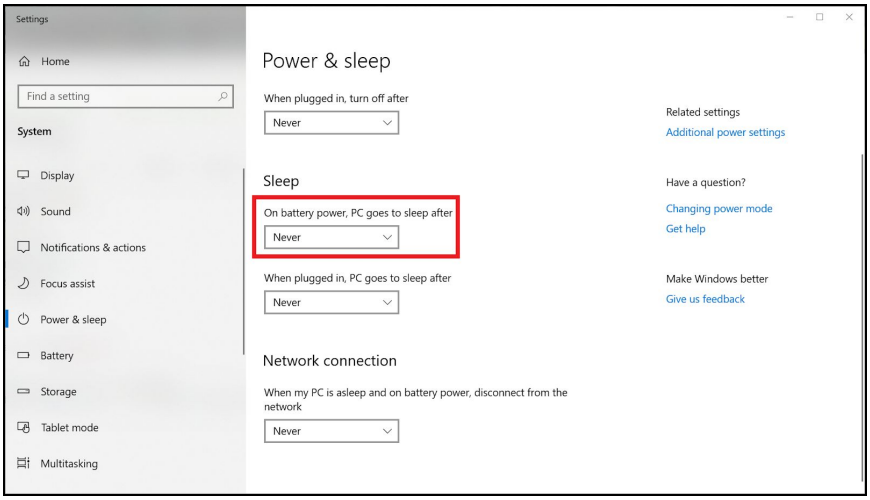




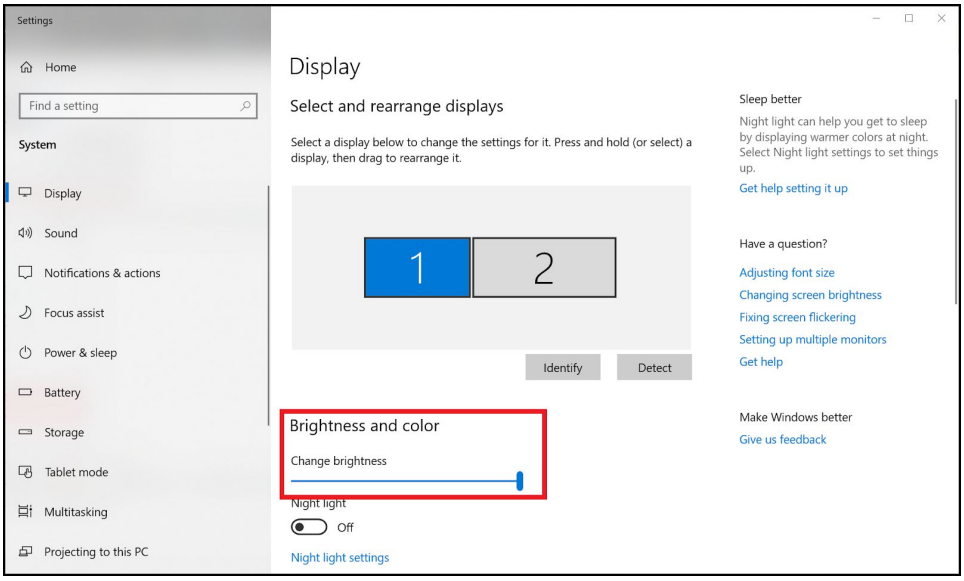
3. **Battery Settings** (Settings → System → Battery)
    a. Turn battery saver on automatically if my battery falls below: **20%**
    b. Lower screen brightness while in battery saver: **Unchecked**



4. **Power & Sleep Settings** (Settings → System → Power & Sleep)
    a. On battery power, PC goes to sleep after: **Never**



5. **Display Settings** (Settings → System → Display)
    a. Open a Notepad window and make sure it is centered in the middle of the screen. Measuring with a light meter in the center of the screen over the blank notepad page, adjust brightness using the slider getting as close to 150 nits as possible.



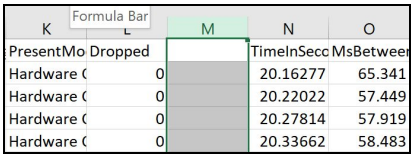
6. Also keep in mind of other system components like Bluetooth or other devices installed/attached to the laptop that could be draining battery life.
7. It's recommended to Install NVIDIA GeForce Experience™ and enable NVIDIA Battery Boost™. Battery Boost works hand in hand to extend battery life whether you are working or playing games, extending battery life by up to 2x.

Capturing Battery Data

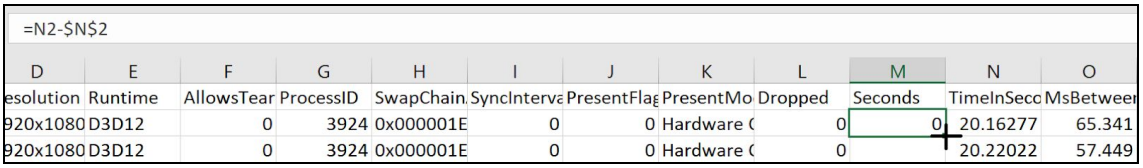
- Open FrameView using the desktop shortcut.
- Decide which game to test and keep note of the app's graphics and video settings.
- Launch the game and find a static scene you wish to test. Unplug the power to the laptop and then press **SCROLL LOCK** to start collecting system and battery life performance while running a game. Do not interact with the system until the battery runs out. Stop the capture manually by pressing **SCROLL LOCK** or wait until the system shuts off (log will be saved). If you completely drain the battery, reconnect the power and boot the system.

Plotting Battery Life

1. Open the FrameView log file (not the summary file) and "Save as…" a new **.xlsx** extension.
    a. Default location of benchmark files: C:\Users\NVIDIA\Documents\FrameView
2. Open the new **.xlsx** file and **create a new column** called Time next to TimeInSeconds.



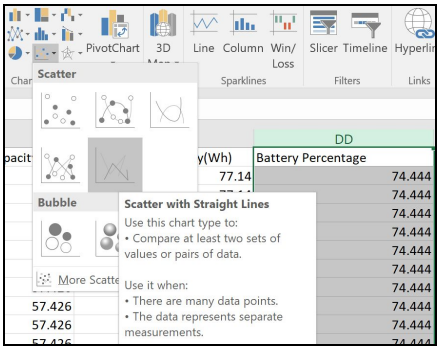
3. In the second row of the new column (M/Seconds), type the formula **=N2-N2** and press ENTER. **Double click the bottom right corner of the 0** to propagate values for all rows. This formula continuously subtracts other TimeInSeconds values from the first TimeInSeconds value in order to get true time for plotting your chart.



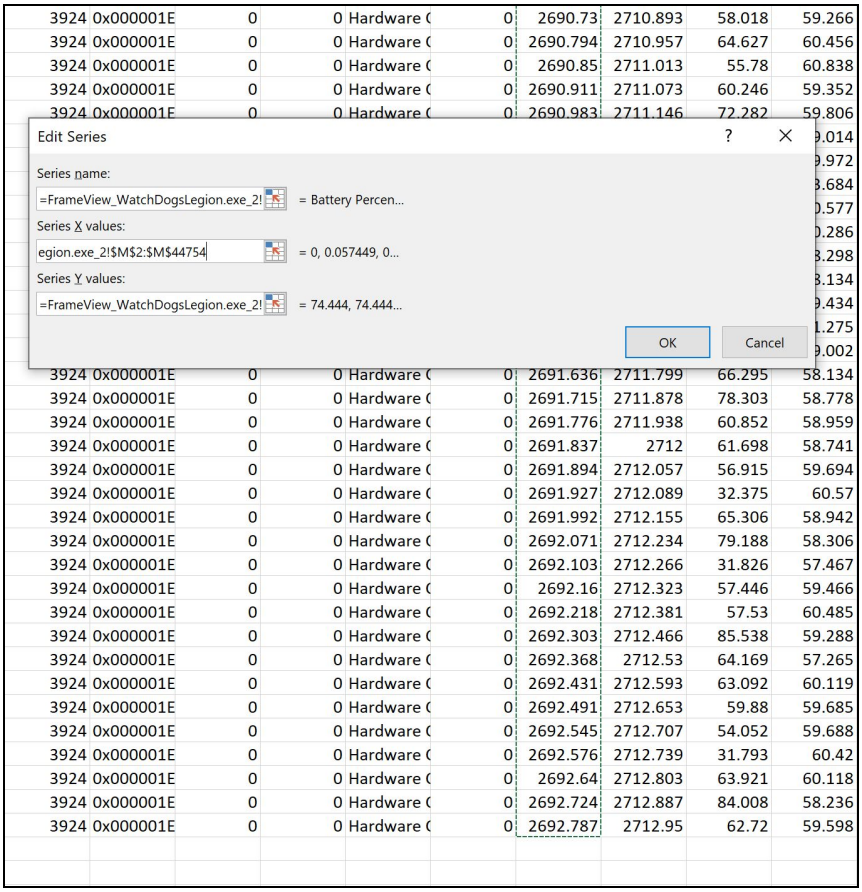
4. Now horizontally scroll to the end of the log to find battery life metrics.

Current Battery Capacity(Wh)	Total Battery Capacity(Wh)	Battery Percentage	Battery Drain Rate(W)
57.426	77.14	74.444	-69.038
57.426	77.14	74.444	-69.038
57.426	77.14	74.444	-69.038
57.426	77.14	74.444	-69.038
57.426	77.14	74.444	-69.038
57.426	77.14	74.444	-69.038
57.426	77.14	74.444	-69.038
57.426	77.14	74.444	-69.038
57.426	77.14	74.444	-69.038
57.426	77.14	74.444	-69.038
57.426	77.14	74.444	-69.038
57.426	77.14	74.444	-69.038
57.426	77.14	74.444	-69.038

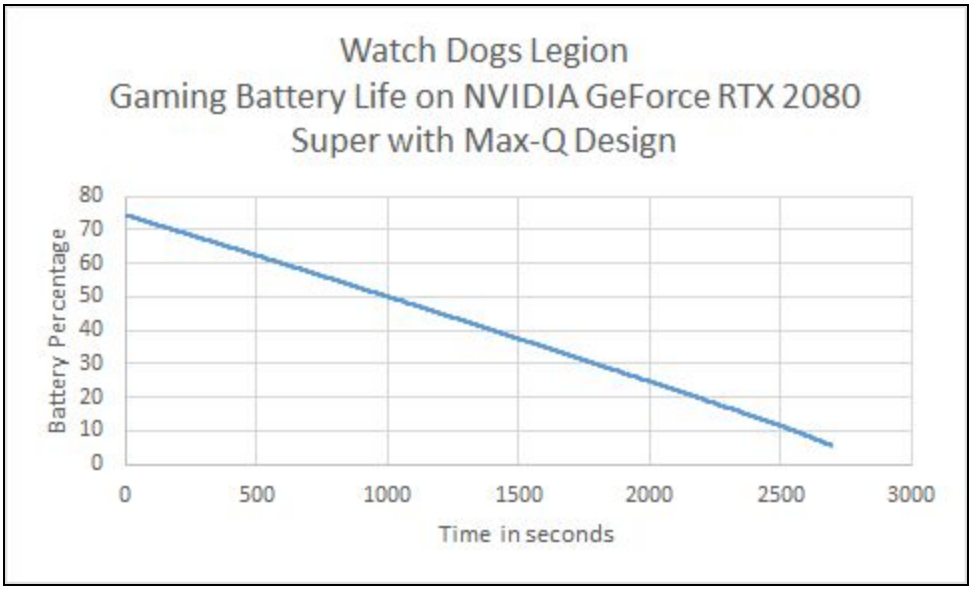
5. Highlight the entire Battery Percentage column and add a **Scatter with Straight Lines**.



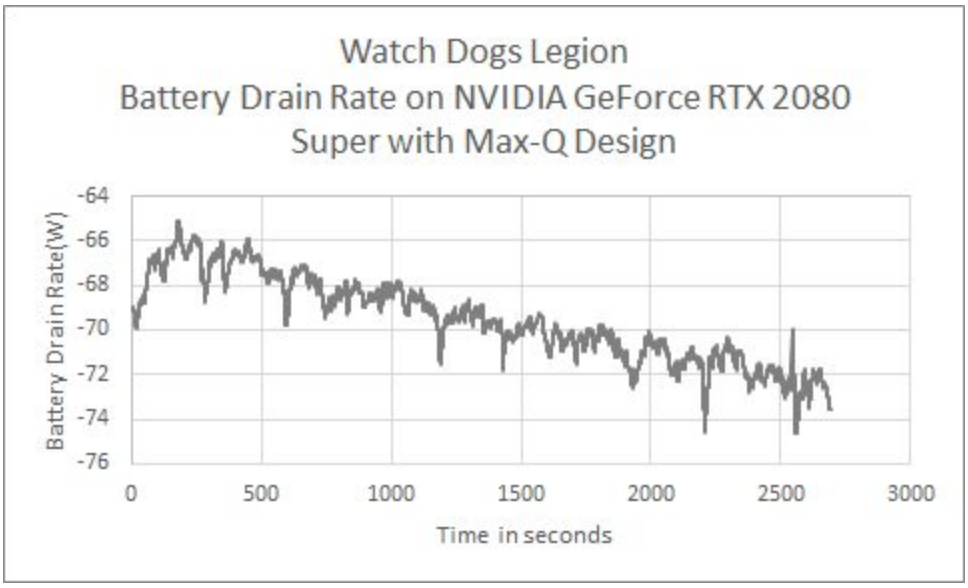
6. Right click the graph, choose Select Data, click Edit, click within the **Series X Values** box and navigate to the newly created Time column using the scrolling bar. Click the first 0 in the newly created Time column then **hold SHIFT + CTRL** at the same time and press **DOWN ARROW** once to select all values for that column. Press ENTER and click OK.



3924	0x000001E	0	0	Hardware (	0	2690.73	2710.893	58.018	59.266
3924	0x000001E	0	0	Hardware (	0	2690.794	2710.957	64.627	60.456
3924	0x000001E	0	0	Hardware (	0	2690.85	2711.013	55.78	60.838
3924	0x000001E	0	0	Hardware (	0	2690.911	2711.073	60.246	59.352
3924	0x000001E	0	0	Hardware (	0	2690.983	2711.146	72.282	59.806
3924	0x000001E	0	0	Hardware (	0	2691.636	2711.799	66.295	58.134
3924	0x000001E	0	0	Hardware (	0	2691.715	2711.878	78.303	58.778
3924	0x000001E	0	0	Hardware (	0	2691.776	2711.938	60.852	58.959
3924	0x000001E	0	0	Hardware (	0	2691.837	2712	61.698	58.741
3924	0x000001E	0	0	Hardware (	0	2691.894	2712.057	56.915	59.694
3924	0x000001E	0	0	Hardware (	0	2691.927	2712.089	32.375	60.57
3924	0x000001E	0	0	Hardware (	0	2691.992	2712.155	65.306	58.942
3924	0x000001E	0	0	Hardware (	0	2692.071	2712.234	79.188	58.306
3924	0x000001E	0	0	Hardware (	0	2692.103	2712.266	31.826	57.467
3924	0x000001E	0	0	Hardware (	0	2692.16	2712.323	57.446	59.466
3924	0x000001E	0	0	Hardware (	0	2692.218	2712.381	57.53	60.485
3924	0x000001E	0	0	Hardware (	0	2692.303	2712.466	85.538	59.288
3924	0x000001E	0	0	Hardware (	0	2692.368	2712.53	64.169	57.265
3924	0x000001E	0	0	Hardware (	0	2692.431	2712.593	63.092	60.119
3924	0x000001E	0	0	Hardware (	0	2692.491	2712.653	59.88	59.685
3924	0x000001E	0	0	Hardware (	0	2692.545	2712.707	54.052	59.688
3924	0x000001E	0	0	Hardware (	0	2692.576	2712.739	31.793	60.42
3924	0x000001E	0	0	Hardware (	0	2692.64	2712.803	63.921	60.118
3924	0x000001E	0	0	Hardware (	0	2692.724	2712.887	84.008	58.236
3924	0x000001E	0	0	Hardware (	0	2692.787	2712.95	62.72	59.598



8. Optionally, you can also plot the battery drain rate.



FRAMEVIEW ANALYSIS TEMPLATE

A Microsoft Excel-based analysis template is provided, offering a fast, simple, and transparent path between capturing your gameplay and reviewing the results. Upload a FrameView log file and you'll see many parts of a basic data processing workflow are automated, including:

1. Describing the log's various capture attributes like the executable name, runtime, resolution, and graphics card.
2. Calculating a summary table with metrics like average, minimum, maximum, and percentile values for each loaded file.
3. Performing a health evaluation to determine if the capture contains any qualities that might affect the validity, accuracy, or comparability of the capture to other files.
4. Creating bar graphs using summary metrics and plot various metrics over time for individual captures as well as categorical comparisons between loaded files.

Using the Template

Getting Started

The analysis template uses macros, or custom VBA code, to load and unload capture files. Depending on how your version of Microsoft Excel is configured, you may need to enable macros by clicking "Enable Content" on the yellow ribbon that appears or navigating to your version's Trust Center and managing permissions. Visit this Microsoft support article for information on enabling macros or to learn about Trust Center settings.

NOTE: You can review these macros before granting permissions by pressing Alt+F11 to open the VBA editor. The "UDFs" module contains all of the code used by this template.

Capture Properties and Label Templates

Here you can load or unload FrameView log files, read or define capture properties, specify the power measurement method, and customize chart titles and capture labels.

Clear All Data	Load Capture #1	Load Capture #2	Load Capture #3
	Clear #1	Clear #2	Clear #3
Capture Properties			
Application	SOTTR.exe	SOTTR.exe	SOTTR.exe
Runtime	D3D12	D3D12	D3D12
Resolution	3840x2160	3840x2160	3840x2160
Graphics Card	NVIDIA GeForce RTX 2080 Ti	***NVIDIA GeForce RTX 2060 SUPER***	***AMD Radeon RX 5700 XT***
Extra Variable			
Power Scheme	(●) GPU Chip Power ( ) Total Graphics Power ( ) Power Capture Analysis Tool (PCAT) **NOTE:** The AMD API currently reports a value that appears to be in-between chip power and board power for AMD graphics cards. It is not currently possible to accurately compare AMD GPU power consumption to NVIDIA GPU power consumption using FrameView.		
Label Templates			
Chart Title Tags	`[Application] ([Runtime]) at [Resolution] w/ [GPU]`		
Capture Tags	`[Variable]`		

Buttons

To analyze a FrameView capture file, press a “Load Capture” button to open a file explorer window and select a FrameView log file, then press “OK” to load the file into the template. Note that the template is designed to read FrameView log files specifically, not the FrameView summary file, and loading a non-FrameView CSV file is not likely to generate useful results. There is some limited compatibility with tools built atop Intel’s PresentMon framework but this template is best used with files generated by FrameView 1.2.

To unload a specific file, click the “Clear” button beneath the respective “Load Capture” button. You also have the option of overwriting a loaded file with another. To clear all loaded files, click the “Clear All Data” button.

Chart Properties

When a FrameView log file is loaded, the cells will reflect the executable name, runtime, output resolution, and retail GPU name found within the capture file, which can save time and reduce errors. If these aren’t written in the way you’d prefer, these cells can be safely overwritten with custom information; the formulas will be reinstated the next time a file is loaded. Also, as shown above, these cells will turn a reddish color in response to differences between capture files like mismatched application names, runtimes, resolutions, or GPUs.

There is also a row for prescribing an extra variable for these captures, such as RTX, DLSS, different image quality settings, or other attributes that aren’t described in the log file and may influence the capture in some manner.

Lastly, you can choose to plot GPU chip power, total graphics power, or data captured by the Power Capture Analysis Tool (PCAT). While FrameView accurately reports both chip and board power for NVIDIA graphics cards, the AMD API used by FrameView appears to report a value in-between chip power and board power for AMD graphics cards. Therefore it's currently not possible to use FrameView to directly compare AMD GPU power to NVIDIA GPU power. Note that AMD total board power can be measured by third-party combination hardware/software interposer testing methodologies including NVIDIA PCAT and others. PCAT works in conjunction with FrameView as described above.

Label Templates

These two cells allow you to customize how information in the Capture Properties section is displayed on charts using a simple tag system. The available tags are: [Application], [Runtime], [Resolution], [GPU], and [Variable]. Tags must be spelled correctly and enclosed by square brackets. All tags are optional and can be repeated, excluded, or rearranged as desired and can be used in either or both templates. This information and default templates are available in the cell comments (red triangle in the corner).

Below are a few examples of chart title template for a set of captures from Shadow of the Tomb Raider (SOTTR.exe) using the DirectX 12 (D3D12) runtime at varying resolutions and running on an NVIDIA GeForce RTX 2060 SUPER:

TEMPLATE	CHART TITLE
[Application] ([Runtime]) at [Resolution] w/ [GPU] |	SOTTR.exe (D3D12) at Different Resolutions w/ NVIDIA GeForce RTX 2060 SUPER |
[GPU] | [Runtime] | [Application] | [Resolution] |	NVIDIA GeForce RTX 2060 SUPER |D3D12| SOTTR.exe | Different Resolutions |
[Runtime] Config: [Application] @ [Resolution] on [GPU] |	D3D12 Config: SOTTR.exe @ Different Resolutions on NVIDIA GeForce RTX 2060 SUPER |

Run Summary

Once files have been loaded, this table will be populated with figures describing various metrics about the capture, including the length in seconds, frame rate, render present latency, power draw, utilization, and more for the CPU and primary graphics device. The files are described according to their extra variable, or if none is provided, the file slot they were loaded into. The table below shows the GPUs used, which can be done by changing the Extra Variable to [GPU]. The drop-down menu changes what formula is used by the table, showing averages by default.

Other formula choices include: minimum, maximum, 90th percentile, 95th percentile, and 99th percentile. This option also changes the values shown in the bar plots (covered below).

Take note of the **PCAT vs API Power (W)** row, which shows the difference (in Watts) between the power measurements obtained from PCAT and total board power figures reported by a graphics vendor API; positive values indicate that PCAT measured more power than was reported by the API, while negative values suggest the API was reporting higher power values than PCAT observed. These differences are sensitive to alignment and sampling rate differences between PCAT and the reporting API.

Run Summary Average	NVIDIA GeForce RTX 2080 Ti	NVIDIA GeForce RTX 2060 SUPER	AMD Radeon RX 5700 XT
Capture Duration (s)	171.8	171.7	172.2
Frame Rate (fps)	61.5	38.9	41.1
Graphics Power (W)	173.3	82.1	179.2
Perf per Watt (F/J)	0.365	0.482	N/A
PCAT vs API Power (W)			
Render Present Latency (ms)	25.8	44.6	43.4
Render Queue (frames)	2.62	2.75	2.80
GPU0 Utilization (%)	98.5%	98.9%	98.4%
GPU0 Temp (°C)	68.8	63.5	74.3
GPU0 Clock (MHz)	1770.7	1808.1	1803.3
CPU Power (W)	50.6	40.5	39.8
CPU Utilization (%)	29.2%	19.9%	19.4%
CPU Temp (°C)	50.3	48.4	47.3
CPU Clock (MHz)	4677.3	4677.2	4675.0

Capture Health

This table shows the results of various checks performed on loaded capture files. Each test can suggest possible issues with a capture by testing for homogeneity and/or specific results in the reported application, resolution, runtime, flip model, and other ideally static attribute columns as

Capture Health	PROBLEMS FOUND - SEE TOOLTIPS		
Application	GOOD	GOOD	GOOD
GPU	GOOD	GOOD	GOOD
Resolution	GOOD	GOOD	CAUTION
Runtime	GOOD	GOOD	GOOD
Allows Tearing	GOOD	CAUTION	CAUTION
Process ID	GOOD	GOOD	GOOD
Swap Chain Address	GOOD	GOOD	GOOD
Sync Interval	GOOD	GOOD	GOOD
Present Flags	GOOD	GOOD	GOOD
Present Mode	GOOD	GOOD	CAUTION
Dropped Frames	GOOD	GOOD	CAUTION
MsBetweenDisplayChange	GOOD	GOOD	CAUTION

well as testing for dropped frames, a hardware-based flip model, frame synchronization, and more. Fields with potential issues are highlighted and display “CAUTION” text, and you can read the associated cell comments (hovering over cells with red triangles in the corner) to learn what issues might be affecting the quality of a capture.

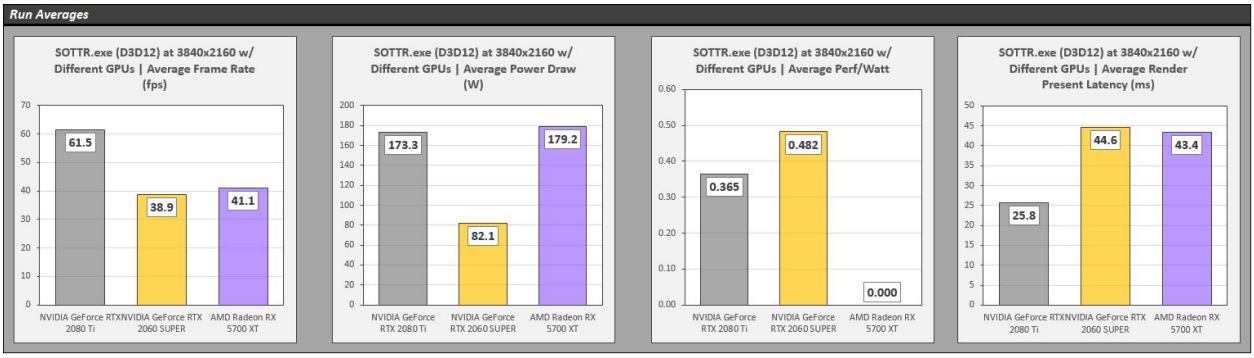
As shown, different results are returned for three files with varying types of potential issues. The first capture had no issues, the second was captured with G-SYNC enabled, and the third shows many issues arising from changing settings and window focus while capturing. Below is an outline of what each attribute is tested for during this process.

NOTE: The “ideal” qualities are prescribed with benchmarking in mind, where performance is prioritized above all else; using exclusive fullscreen modes, no variable refresh rate technology, no vertical synchronization policies, and so on. This table simply tries to explain possible issues, limitations, losses of accuracy, or less than ideal qualities that could affect comparisons in purely performance-focused testing. Regardless of the number of possible issues, the analysis template will render plots and calculate tables as best it can. Your typical gaming environment will probably deviate from these expectations in some way - for example, running in borderless fullscreen mode with G-SYNC and V-Sync enabled. If you are interested in comparing captures under your normal circumstances and understand the limitations therein, some of the results from this table can be safely ignored.

ATTRIBUTE	IDEAL QUALITY
Application	One unique value.
GPU	One unique value.
Resolution	One unique value that is not “WINDOWED”.
Runtime	One unique value that is not “Other”.
Allows Tearing	At least one value equals 1.
Process ID	One unique value.
Swap Chain Address	One unique value.
Sync Interval	One unique value that is less than 1.
Present Flags	One unique value.
Present Mode	One unique value that is hardware-based.
Dropped Frames	All values are equal to 0.
MsBetweenDisplayChange	No values are equal to 0.

Run Summary Bar Plots

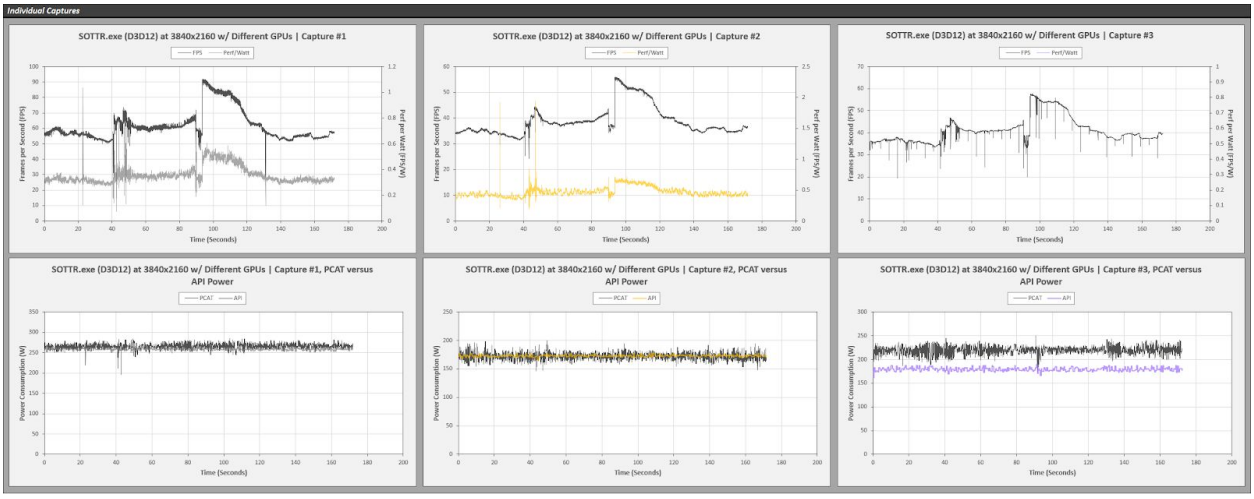
Beneath the file data tables is a row of bar plots which reflect four key metrics from the **Run Summary** table: frame rate, power draw, perf per watt, and render present latency. The drop-down menu controls what calculations are used, with the averages as the default option. Changes are immediately reflected in both the chart titles and values. The x-axis minimum for all plots are fixed to 0, and the x-axis maximum is automatically scaled to your data.



Individual Capture Plots

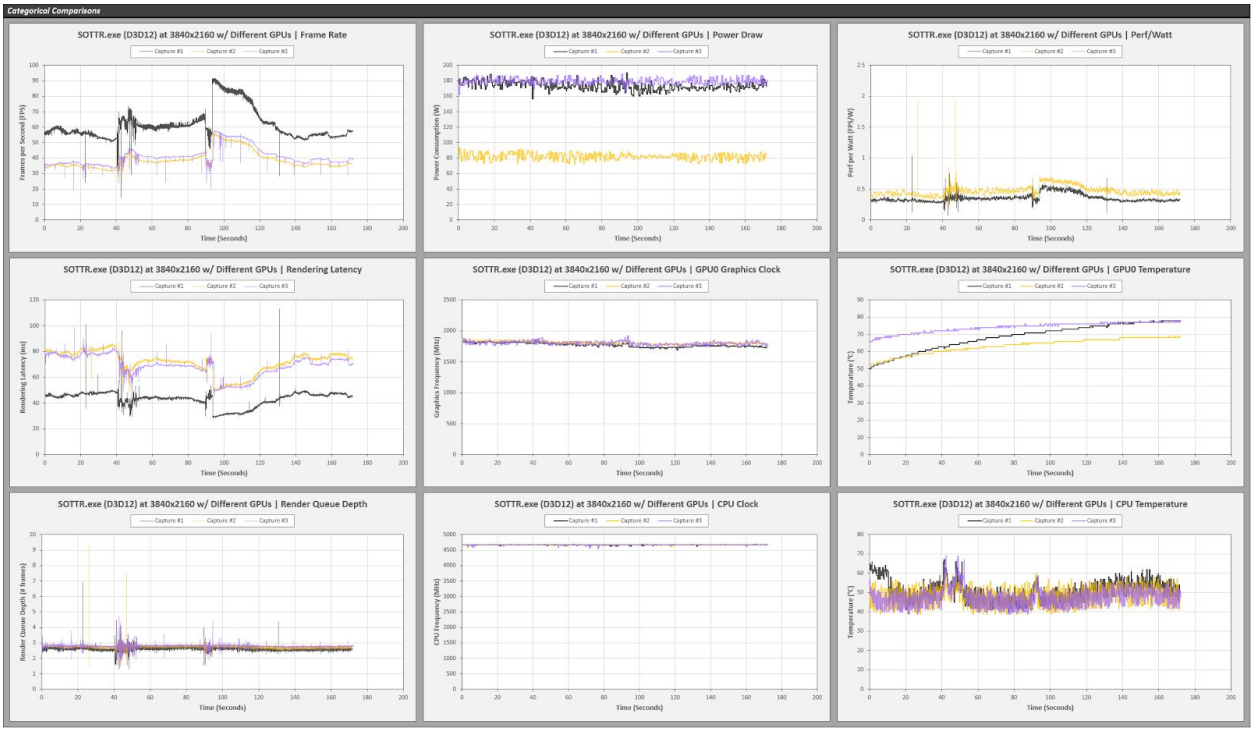
Each file has its own time-based scatter plots generated from frame rate and perf per watt measurements. These plots allow for isolated analysis which can be helpful when performance between the captures is highly similar, where key micro-scale trends might be obscured, or when performance is highly dissimilar, where a single large stutter event or a significant delta could suppress more nuanced behaviors by stretching the y-axis. The x-axis and y-axis minimums for all plots are fixed to 0, and the maximums are automatically scaled to your data.

In addition, if PCAT measurements were included in the FrameView log file, this data and API power data will be featured in another set of scatter plots. These graphs can be useful for understanding smaller (faster) power trends as PCAT captures data much more quickly than would be practical for API polling.



Categorical Comparison Plots

Further below is a block of categorical plots featuring all loaded capture files, providing head-to-head graphs for nine metrics: frame rate, power draw, perf per watt, render present latency, GPU0 core frequency, GPU0 core temperature, render queue depth, CPU frequency, and CPU temperature. The x-axis and y-axis minimums for all plots are fixed to 0, and the maximums are automatically scaled to your data.



Charting Performance Data

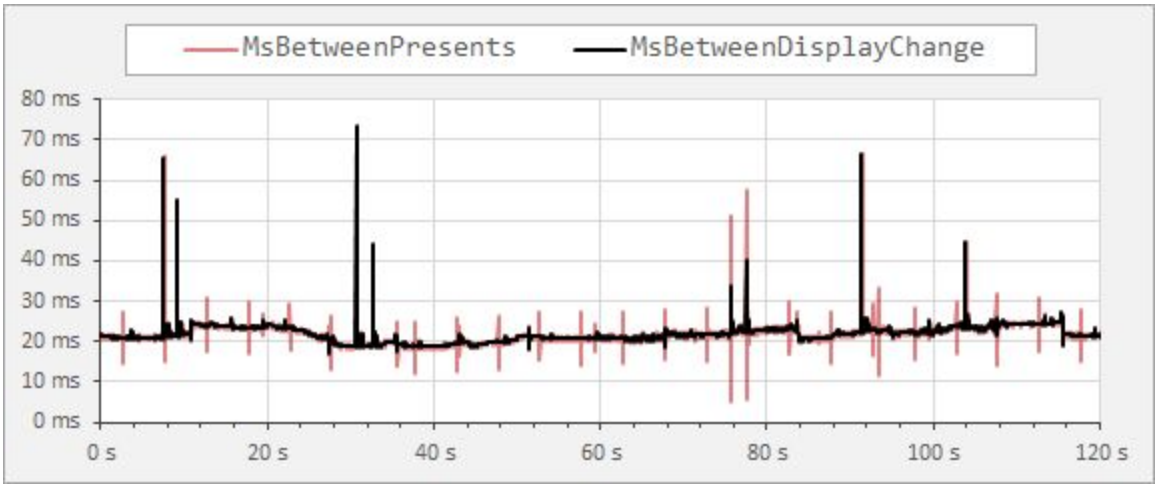
MsBetweenDisplayChange

(Displayed FPS) Should be used to chart displayed FPS. This data is captured from the end of the graphics pipeline and is an indicator of what the user actually sees displayed on screen.

MsBetweenPresents

(Rendered FPS) Can alternatively be used to chart rendered FPS. This data is captured from the beginning of the graphics pipeline and indicates the smoothness of the animation delivered to the GPU. This is the data that is typically provided by other benchmarking capture tools, but can't capture driver-side improvements like frame metering.

The plot below illustrates the difference between these metrics for the same capture:



Charting Percentile Data

FrameView Scan Logs provide percentiles which illustrate the frame rates that given percentages of frames can achieve, and FrameView Scan Reports provide frametime data to calculate the frametimes below which a given percentage of frametimes will fall. A 95th percentile frametime is the value below which 95% of the frametimes are found. Example: if the dataset has a rendered frametime 95th percentile of 16.67ms, then 95% of the frames were rendered faster than 16.67ms.

NOTE: Percentile data is highly sensitive to stutter. In order to obtain the most meaningful results, exclude loading screens, menus, and large frametime spikes when selecting a dataset.

Charting NVIDIA Power Data

NVIDIA Chip Power Consumption

GPUOnlyPwr(W) (API)

Should be used for charting the average GPU (chip) power consumption.

NVIDIA Chip Performance per Watt

Perf/W GPUOnly(F/J) (API)

Should be used for charting performance per watt data for NVIDIA GPUs (chip) where F is frames and J is joules (one joule is the equivalent of one watt of power radiated or dissipated for one second). So F/J would be frames per second (F/S) divided by watts (J/S).

$$F/J \;=\; (F/S) \,/\, (J/S)$$

For more details on measuring power of GPUs, please refer to the **NVIDIA GeForce GPU Power Primer**.

NVIDIA Board Power Consumption

NV-Total-USBCPwr(W) (API)

Should be used for charting Total Graphics Power (TGP). TGP is the maximum power in watts that a power supply should provide to the graphics board. TGP is also defined as the average power consumed by the entire graphics board subsystem while executing a very stressful "real world" application. TBP or Total Board Power is essentially the same as TGP. Using this data will be more accurate since it does not include the power used by devices that may be connected to the USB-C connector on NVIDIA GeForce RTX graphics cards.

NOTE: FrameView is not as accurate as interposer/riser card techniques for measuring ***idle*** chip or ***idle*** board power. It is accurate for load testing, so it is best to use FrameView when running real-world applications that stress the GPU.

NVIDIA Board Perf Per Watt (PPW)

Since FrameView captures both performance and power data, it allows users to create accurate perf-per-watt statistics to determine GPU efficiency by viewing the performance of the GPU alongside the power it uses. This metric is called performance-per-watt (PPW). The lower the power utilization and the higher the game performance, the better the perf-per-watt.

Perf/W Total-USBC(F/J) (API)

Should be used for charting performance per watt data for Total Graphics Power (TGP) where F is frames and J is joules (one joule is the equivalent of one watt of power radiated or dissipated for one second). So F/J would be frames per second (F/S) divided by watts (J/S).

$$F/J \;=\; (F/S) \,/\, (J/S)$$

For more details on measuring power of GPUs, please refer to the **NVIDIA GeForce GPU Power Primer.**

Charting AMD Power Data

AMD Board Power Consumption

AMDPwr(W) (API)

Should be used for charting the average GPU (board) power consumption for AMD GPUs.

While FrameView accurately reports both chip and board power for NVIDIA graphics cards, the AMD API used by FrameView currently only reports a value that appears to be in-between chip power and board power for AMD graphics cards. Therefore it's currently not possible to use FrameView to directly compare AMD GPU power to NVIDIA GPU power. It may be useful to ask AMD if they can report chip-only and full board power with the APIs, similar to NVIDIA. FrameView will be updated to capture total board power and chip power for AMD if they make such information publicly available in their API.

AMD Board Perf Per Watt (PPW)

Since AMD board power cannot be accurately measured with FrameView, performance per watt data can not accurately be calculated. Therefore, it's currently not possible to use FrameView to directly compare AMD GPU perf per watt data to NVIDIA. It may be useful to ask AMD if they can report chip-only and full board power with the APIs, similar to NVIDIA. FrameView will be updated to capture total board power and chip power for AMD if they make such information publicly available in their API. Note that AMD total board power can be measured by third-party combination hardware/software interposer testing methodologies including NVIDIA PCAT and others. PCAT works in conjunction with FrameView as described above.

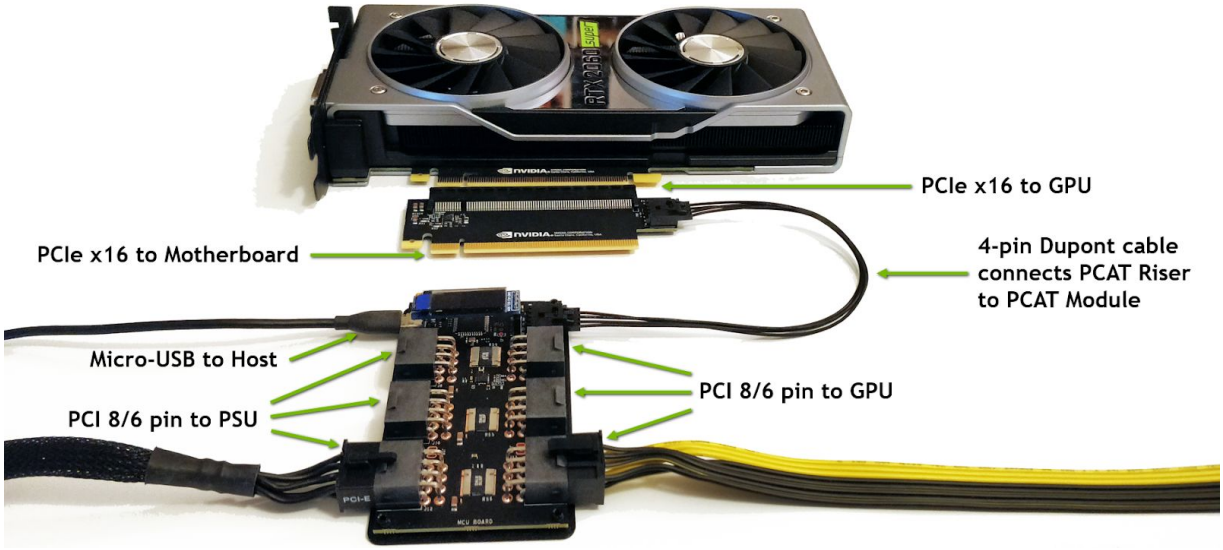
AMD perf per watt and total board power can only be measured properly with PCAT or another interposer. The correct values are represented by the **PCAT Power Total(W)** and **Perf/W Total(F/J) (PCAT)** headers in the FrameView log and summary files when using Power Capture Analysis Tool (PCAT).

Power Capture Analysis Tool (PCAT)

The PCAT Module contains three 8-pin power connectors on each side with a shunt between each pair of connectors that accurately reads and captures the power data from the PCIe power leads from the PSU to the GPU.

Power data is captured from the PCAT PCIe Riser over the 4-pin cable connecting it to the PCAT Module, and is then combined with the power data captured from the 6- or 8-pin PCIe power cables from the PSU. All of this data is sent to the PC across the micro-USB cable from the PCAT Module.

Contact an NVIDIA PR representative to get your hands on a PCAT Module as an alternative method for testing power.

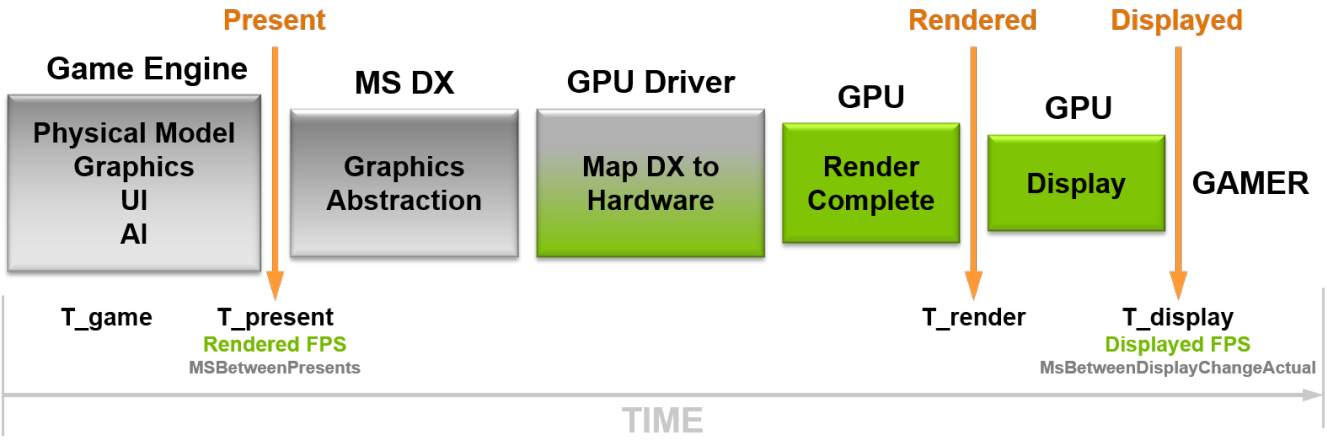


HOW FRAMEVIEW WORKS

To provide performance data for an accurate comparative analysis of GPUs, FrameView measures timestamps at the beginning of the graphics pipeline to provide a metric indicating the smoothness of the animation delivered to the GPU, and at the end of the pipeline to provide an indicator of what the user actually sees displayed on screen.

Frame Rendering Pipeline

The diagram below shows how game frames are created at the beginning of the pipeline and their path to the display.



FrameView provides performance data that is captured in the Present and Displayed portions of the game/graphics pipeline. Data from these two areas will always be reported in the logs. You can choose which you would like shown in the overlay using the FrameView settings in the interface. They are called Rendered FPS and Displayed FPS.

> **Rendered FPS (MsBetweenPresents)** measures timestamps from the beginning of the graphics pipeline and is a metric indicating the smoothness of the animation delivered to the GPU. This is the data that is typically provided by other benchmarking capture tools.
>
> **Displayed FPS (MsBetweenDisplayChange)** measures timestamps at the end of the game pipeline and is an indicator of what the user actually sees displayed on screen.

Stutter is the variation between T_game and T_display. This data is also reported by FrameView in the logs. The header is called MsUntilRenderComplete and it measures the time between present start and GPU work completion.

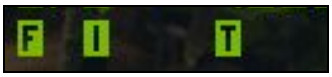
TROUBLESHOOTING

FrameView

Frames are capped at 30fps, 60fps, 75 fps (or any other framerate) in a game

The game may have a frame rate cap “framecap” (internal frame limiter) that prevents rendering faster than a specified rate. Check the game settings to see if a framecap is set.

This can also be caused by having V-sync enabled (ON), which will synchronize the frame rate to the refresh rate of the monitor. Check the game settings and disable V-sync to ensure that the frame rate is no longer tied to the monitor refresh rate. The FrameView overlay will show an “T” when V-sync is OFF (for tearing) and will show a “V” when V-sync is ON. It’s best to force Vertical Sync to Off in Manage 3D Settings found in the NVIDIA Control Panel.



NVIDIA Perf per Watt results are not showing in the FrameView overlay

You may need to reinstall FrameView again by clicking FrameViewSetup.exe.

The FrameView overlay is not being displayed over a game

Overlay information will not be shown in DX9/10 and Vulkan-based games. However, data capture is supported and the information will be properly logged.

If the overlay is not being displayed over DX12 or OpenGL games, close all applications including FrameView. Reopen FrameView and launch the application again.

Running FrameView and FRAPS Concurrently

Since FrameView and FRAPS both are both hooking into application processes, FrameView might not work with x64-bit applications if FRAPS is already running in the background.

1. Launch FrameView
2. Launch Game
3. Let FrameView overlay appear
4. Launch FRAPS
5. Close FRAPS before closing the game, and then follow steps 1-4 for the next run

Excel Analysis Template

What version of Microsoft Excel is required to use the analysis template?

The Excel template uses formulas and procedures that require at least Microsoft Excel 2010. Excel 2016 or later is recommended. Earlier versions of Excel will experience significant losses of functionality.

File loaded but there is no information in the tables or charts.

Verify that the loaded file was a FrameView Log file (not the summary file) and then check the Capture Health table to learn about potential integrity problems. If any modifications have been made to formulas, chart properties, named ranges, or VBA code in the template, reattempt with an unaltered copy of the original template.

The Capture Health table says “NOT FOUND” for one or more of my file’s attributes.

“NOT FOUND” is returned when a file has been loaded into the template but does not contain one or more column headers that are evaluated for the Capture Health check. The loaded file may have been generated with an older version of FrameView, created by another application (e.g., FRAPS, PresentMon, etc.), may have had key columns manually removed, or was otherwise not interpretable by the analysis template.

Excel slows down or crashes when importing large files into the template.

As with all software, Excel works with a limited economy of resources. Excel becomes less responsive when viewing large data files on their own; this template calculates dozens of various metrics and plots four bar charts and fifteen plots for up to three files with many thousands of data points, which places even greater demand on the application. While there is no explicit file size limit, we recommend limiting capture duration or trimming larger files down to 50 MB or less.

LEGAL

Notice

ALL INFORMATION PROVIDED IN THIS WHITE PAPER, INCLUDING COMMENTARY, OPINION, NVIDIA DESIGN SPECIFICATIONS, REFERENCE BOARDS, FILES, DRAWINGS, DIAGNOSTICS, LISTS, AND OTHER DOCUMENTS (TOGETHER AND SEPARATELY, “MATERIALS”) ARE BEING PROVIDED “AS IS.” NVIDIA MAKES NO WARRANTIES, EXPRESSED, IMPLIED, STATUTORY, OR OTHERWISE WITH RESPECT TO MATERIALS, AND EXPRESSLY DISCLAIMS ALL IMPLIED WARRANTIES OF NONINFRINGEMENT, MERCHANTABILITY, AND FITNESS FOR A PARTICULAR PURPOSE.

Information furnished is believed to be accurate and reliable. However, NVIDIA Corporation assumes no responsibility for the consequences of use of such information or for any infringement of patents or other rights of third parties that may result from its use. No license is granted by implication or otherwise under any patent or patent rights of NVIDIA Corporation. Specifications mentioned in this publication are subject to change without notice. This publication supersedes and replaces all information previously supplied. NVIDIA Corporation products are not authorized for use as critical components in life support devices or systems without express written approval of NVIDIA Corporation.

Trademarks

NVIDIA, the NVIDIA logo, and GeForce are trademarks or registered trademarks of NVIDIA Corporation in the United States and other countries. Other company and product names may be trademarks of the respective companies with which they are associated.

Copyright