Battery Life Analyzer

User Guide

March 2013

Revision 2.5

Document Number: 448043

Contents

Figures

Tables

Revision History

Document Number	Revision Number	Description	Revision Date
448043	0.7	Initial release of Battery Life Analyzer User Guide for Battery Life Analyzer Pre-Alpha	May-10
	1	Final release of Battery Life Analyzer User Guide for Battery Life Analyzer Pre-Alpha	Jul-10
	1.1	Updated release of Battery Life Analyzer User Guide for Battery Life Analyzer Pre-Alpha	Oct-10
	2	Battery Life Analyzer User Guide updated with version 2.0 features and bug fixes	Feb-11
	2.01	Updated Battery Life Analyzer User Guide with enhanced platform support features added in Battery Life Analyzer 2.01	Oct-11
	2.1	Updated Battery Life Analyzer User Guide with enhanced usability features added in Battery Life Analyzer 2.1.	Dec-11
	2.2	Updated Battery Life Analyzer User Guide with additional platform support features, such as, battery usage module, XHCI LTM module and enhanced usability features, such as, new command line options for C-State module that are incorporated in Battery Life Analyzer 2.2.	Mar-12
	2.3	Battery Life Analyzer User Guide is updated with new BLA features, such as, parallel data collection capability of selected modules and data collection during Connected Standby. Also, new capabilities of the Device Power Analysis module described along with minor changes/updates on other modules.	Jun-12
	2.4	Battery Life Analyzer User Guide is updated with improvements in Device Power Analysis Module, such as, device power resource mapping and D3 Cold analysis, Panel Self Refresh (PRS). Also new BLA features, such as, Power Feature Quick Check module is described along with minor changes/updates on other modules.	Oct-12
	2.5	Battery Life Analyzer User Guide is updated with new features such as LPSS LTR analysis, Connected Standby analysis module along with minor improvements to existing capabilities.	Mar-13

§

1 Introduction

Battery Life Analyzer (BLA) is a software tool that monitors various software and hardware activities that affect battery life. This document is a User Guide for BLA and will walk the user through set up, analysis, and system debug steps.

Please submit further questions that are not addressed in this document to: batterylifeanalyzer@intel.com.

1.1 Terminology

Term	Description
ALPE	Aggressive Link Power Management Enable
ASP	Aggressive Slumber
ASPM	Active State Power Management
BLA	Battery Life Analyzer
DIPM	Device Initiated Power Management
DPC	Deferred Procedure Call
EHCI	Enhanced Host Controller Interface
EP	End Point
ESP	External SATA Port
ETW	Event Tracing for Windows
ETL	Event Trace Log file
GUI	Graphical User Interface
HDD	Hard Disk Drive
HIPM	Host Initiated Power Management
HPCP	Hot-Plug Capability
ISR	Interrupt Service Routine
LTR	Latency Tolerance Reporting
RP	Root Port

Term	Description
UHCI	Universal Host Controller Interface
xHCI	Extensible Host Controller Interface
PCIe	PCI Express*
PSR	Panel Self Refresh
RTD3	Runtime D3
LPM	Link Power Management
SSD	Solid State Drive
CS	Connected Standby

1.2 Reference Documents

Document	Document No./Location
PCI Express Base Specification Rev 2.1	http://www.pcisig.com/specifications/pciexpress/base2
PCI Express 2.0 Latency Tolerance Reporting ECN	http://www.pcisig.com/specifications/pciexpress/specifications/ECN_LatencyTolnReporting_14Aug08.pdf
PCI Express 3.0 L1 PM Substates with CLKREQ ECN	http://www.pcisig.com/specifications/pciexpress/specifications/ECN_L1_PM_Substates_with_CLKREQ_23_Aug_2012.pdf
Energy-Efficient Platforms – Whitepaper (Considerations for Application Software and Services)	http://download.intel.com/technology/pdf/Green_Hill_Software.pdf
Energy-Efficient Platforms – Whitepaper (Designing Devices Using the New Power Management Extensions for Interconnects)	http://download.intel.com/technology/pdf/322304.pdf
Designing Energy Efficient SATA Devices	http://download.intel.com/technology/EEP/Designing_energy_efficient_SATA_devices.pdf
Universal Serial Bus Revision 2.0 Specification	http://www.usb.org/developers/docs/
Universal Serial Bus Revision 3.0 Specification	http://www.usb.org/developers/docs/

§

2 Online Help

BLA supports online help to provide information from the User Guide, Energy Efficient Platform White Paper and the Hardware Architecture White Paper. Selecting any one of these resources from the help menu opens up the document to the title page.

2.1 F1 Support

When a module is selected, the F1 key can be used to open the User Guide to the section pertaining to that module. When no module is selected, the F1 key will open the User Guide at the title page.

2.2 Contextual Help

Support is also available within the modules to help in analysis or further investigation. The relevant section of the Hardware or software white paper is displayed when a highlighted cell is right-clicked. Also, right-clicking a module in the Module explorer will display a context menu with an item to display. Selecting this menu item will display the User Guide for that module. Similar support is also available in the summary screen with a right-click on a row for a warning where the relevant section of the white paper is displayed with links to the possible areas for further investigation.

§

3 Command Line Operation

In addition to the Graphical User Interface (GUI), BLA has a very powerful Command Line Interface (CLI) that can be utilized especially when automating BLA Analysis using scripts. In order to invoke BLA CLI, open a Command Prompt window as the Administrator and type the BLA commands in the following format:

BLA <command> [options]

The CLI commands are listed below:

Table 1. BLA CLI Commands

Command	Action
Help | h | -h | /?	Display the CLI help
List | l	List the available modules
Options | o	List available options for a module
Install | i	Installs all dependencies
Uninstall | u	Uninstall all dependencies
Capture | c <module[:duration][,option_name=value] [+[module[:duration...]]>	Runs module for optional duration (seconds) with options specified
Concurrent Capture | cc <module[:duration][,option_name=value] [+[module[:duration...]]>	Concurrently captures data for specified modules for optional duration (seconds) with options specified. Note that, concurrent capture feature is only available for selected modules; refer to section 3.1.6 for more information.

BLA CLI also has an option of saving the output from the run in the directory specified with the –o <directory> option. Note that, the default CLI behavior is to save the files in the current working directory.

The commands ”List” and “Options” can be used at any time to show the available modules and their options. The command line interface allows for the sequential and parallel execution of the commands.

3.1 Usage

Before using BLA in command line mode to capture data, the install command needs to be run. This is done to pre-install the required drivers. By doing this, the drivers to not need to be re-installed for each subsequent capture. After data capture is complete, the uninstall command should to be run to remove all dependencies from the system.

3.1.1 List Command

The list command provided the list of modules supported in BLA. The command can be executed as follows:

“BLA list“ or “BLA l”

The various modules supported in BLA are as follows:

bat (Battery Usage) *
cs (Connected Standby) *
cst (CPU C-State) *
disk (Disk Activity)
gfx (Graphics Activity)
hdd (HDD Spin-Down)
sw_long (Long Duration Software Acti
lpss_ltr (LPSS LTR)
mem (Memory Activity)
pcie_lpm (PCIe LPM)
pcie_ltr (PCIe LTR)
pltr (Platform LTR)
check (Power Feature Quick Check)
dev_power (Device Power Analysis) *
sata_lpm (SATA LPM)
sata_lat (SATA Latency)
sw (Software Activity Analysis) *
ehci (EHCI Analysis)
xhci_lpm (XHCI LPM)
xhci _ltm (XHCI LTM)

Notice that, some modules listed with an “*” . This indicates that module can be run concurrently with other modules that are listed with an “*”. Please see section 3.1.6 Concurrent Capture Command for more information on how to perform concurrent Capture.

3.1.2 Options Command

This command lists the options available with each of the modules and can be invoked by typing “BLA options <module name>“ or “BLA o <module name> “.

For instance “BLA o sw” will display:

```
Options for Software Activity Analysis
    Name   Type        Description
    ------------------------------------------------------------------
    s      Boolean     Enable Symbol Translation
    t      Boolean     Enable Stack Trace Information
    m      Number      Set log file size limit in MB (0 = unlimited)

           Valid input for option types
           Type         Valid Input
           -----------------------------------------------------------
           Boolean       yes|y|true|t|no|n|false|f (Not case sensitive)
           Number        Any 32-bit integer value (0, 1, 2, 5462543442, ...)
           String       Any mix of word characters, ':', or '\'
```

3.1.3 Install Command

The install command is required to be run before data collection can proceed in the command line mode. The command is run as follows:

“BLA install” or “BLA i”

Any errors will be displayed on screen.

3.1.4 Uninstall command

The uninstall command can be used to remove all BLA dependencies from the system, It is run as follows:

“BLA uninstall” or “BLA u”

Any errors will be displayed on screen

3.1.5 Capture Command

This command is to capture data from the analysis of the system with the various modules in BLA.

“BLA capture <module[:duration][,option_name=value][+[module[:duration...]>

“BLA c <module[:duration][,option_name=value][+[module[:duration...]>

<module>: is one of the modules supported by BLA.

<duration>: number of seconds for data capture

<option_name>:An option for the module. Multiple options can be specified in a comma separated list.

<value> : value for module option.

Several modules can also be run with one capture command by sequentially specifying the modules in the command line separated by "+".

Starting with BLA version 2.2, C-state module supports additional options that allow BLA to save data periodically during the measurement time window. See the section 3.2 on an example of how to use this new feature. Additional options introduced by this feature are as follows:

<savehist>: Instructs BLA to save the data collection history to a file

<histlimit>: Number of records to save (last n records)

<poll>: data saving period in milliseconds (0 means feature is disabled)

3.1.6 Concurrent Capture Command

Concurrent capture command (cc) enables user to execute several modules at the same time. Note that, this feature is only available in the BLA Command Line Interface at this time. In order to find out which BLA modules can be run concurrently, use options command and look for modules that are listed with "*" as follows:

BLA –l

"BLA cc <module[:duration][,option_name=value][+[module[:duration...]>

Notice that, the meaning of the "+" in concurrent capture command to specify modules to be run concurrently (not sequentially as in capture command).

3.2 Example Run

This command line will run the software analysis module for 15 seconds with symbol translation enabled, run the C-State module for 10 seconds, and output captured data to c:\test directory.

"BLA c sw:15,s=true+cst:10 –o c:\test "

BLA version 2.2 C-State module allows periodic saving of the results during the measurement window. An example use of this command would be as follows:

“BLA c cst:300,savehist=All_cstatedata.csv,histlimit=3000,poll=100”

As a result of this command, BLA will create a new .csv file named All_cstatedata.csv and it will contain 3000 records that are sampled every 100ms.

The command below is an example usage for concurrent capture feature added in BLA 2.3. In this command, C state, SW activity, device power analysis and battery modules executed for 120 seconds:

“BLA.exe cc cst:120,savehist="cstdata.csv",poll=1000,histlimit=3000+sw:120+bat:120+dev_power,st="C:\BLA2.3.0.10018_BETA\etw\evt.etl",s=y,le=y”

§

4 Battery Life Analysis

4.1 Platform Hardware and Software requirements

BLA tool supports the following Intel® Architecture **mobile** platforms:

- Calpella (Intel® 5 Series Mobile Chipset and Intel® Core™ i7/i5/i3 Processor)
- Huron River (Intel® 6 Series Mobile Chipset and 2nd Generation Intel® Core™ i7/i5/i3 Processor)
- Chief River (Intel® 7 Series Mobile Chipset and 3rd Generation Intel® Core™ i7/i5/i3 Processor)
- Shark Bay (Intel® 8 Series Mobile Chipset and 4th Generation Intel® Core™ i7/i5/i3 Processor, 2-Chip and ULT)

Note: Platforms prior to the ones listed above are not supported and usage of BLA tool will result in an error. BLA does not support Desktop, Server, or Handheld platforms.

BLA supports the following Operating Systems:

- Microsoft Windows 7* RTM, SP1 (32 and 64-bit)
- Microsoft Windows 8* , last tested version is build 9200 (32 and 64 bit)

Note: Please refer to the release notes for known issues related to Windows 8.

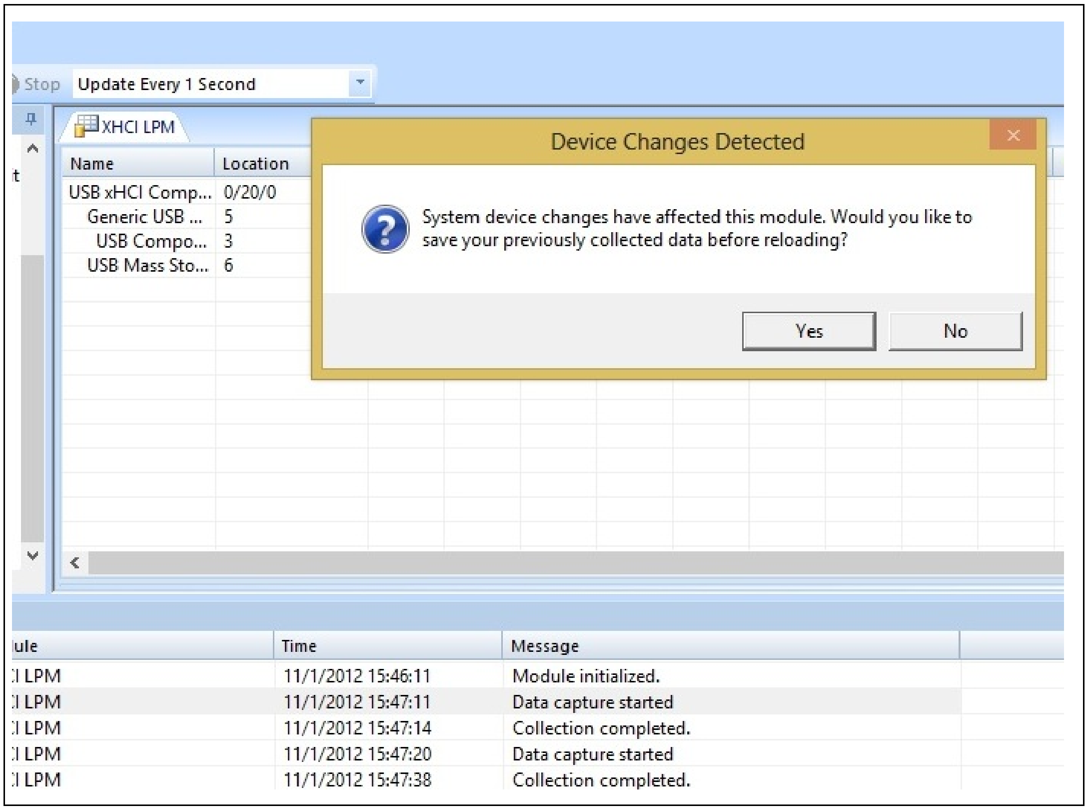
Note: BLA does not support virtual environments (Virtual PC*, Microsoft Windows XP* mode, etc.).

BLA will install all necessary drivers and DLLs each time it is invoked. BLA will also remove installed drivers and DLLs upon every graceful termination of the tool.

Note: BLA now detects all PnP activity while a module is loaded and provides the user with the opportunity to save the previously collected data before it reloads, see Figure 1.

Note: BLA now can perform various functions through connected standby. Please see 4.4 BLA Connected Standby Analysis section for more information on how to analyze platform behavior.

Figure 1. Popup Message for Device Change



BLA tool start-up requirements:

- **Users should not plug / unplug devices while BLA modules are being loaded:** BLA will ignore PnP events when the BLA modules are being loaded, therefore a device plugged or unplugged during this time may not be present in BLA analysis. If a PnP event happened during this time, user should exit out of BLA and restart it.
- In order to conduct new analysis, user should run BLA in Administrator Mode by right clicking on the BLA icon and selecting “Run as Administrator” option. If BLA is not started in Administrator Mode, it will not be able to collect new data but it can open ETL files saved from old data collections.
- If prompted that “A program needs your permission to continue” during application startup, BLA will need to install a driver for complete functionality. The user will need to click “Continue” in order to run the tool properly. Another prompt will appear asking the user, “Would you like to install this device software?” for which the user needs to click “Install”.
- Users starting BLA for the first time will be prompted to accept the License Agreement.

- BLA will expire after 1 year from the release date and end user will be reminded to obtain a more recent version of BLA when BLA starts.

4.2 Power and Battery Life Impact Estimate

For selected platforms, BLA provides an estimate for power impact to the platform idle power based on an algorithm that is beyond the scope of this documentation. In order to understand how it translates to a reduction in battery life, a user will need to know Total Battery Capacity (Watt*Hour) and Baseline Platform Power (BPP):

$$BPP\ (Watt) = \frac{Battery\ Capacity\ (Watt \times Hr)}{Baseline\ Rundown\ time\ (Hr)}$$

$$Battery\ Life\ Impact\ (Hr) = Baseline\ Rundown\ Time\ (Hr) - \frac{Battery\ Capacity(Watt \times Hr)}{BPP\ (Watt) - \ Power\ Impact\ Estimate\ from\ BLA\ (Watt)}$$

Note: Power Estimate provided by BLA is intended to give a guideline of potential power savings. It does not represent actual measured power on the system.

BLA Power estimates are available for the following platforms:

- Montevina
- Capella – Arrandale
- Hurronriver – Sandybridge – Dual core
- Hurronriver – Sandybridge – Quad core

4.3 BLA Modules Overview and Operation

There are various software and hardware activities that affect battery life and can use considerable amounts of power. Often, misbehaving devices or software can waste power even when the system is idle. BLA has several analysis modules that provide a complete picture of which components are causing additional power consumption. The *Module Explorer* window displays the modules that are loaded when BLA starts (Figure 2).

Analysis modules can be run individually with the Capture option (Manual Mode), or sequentially with the Capture All option (Automatic Mode).

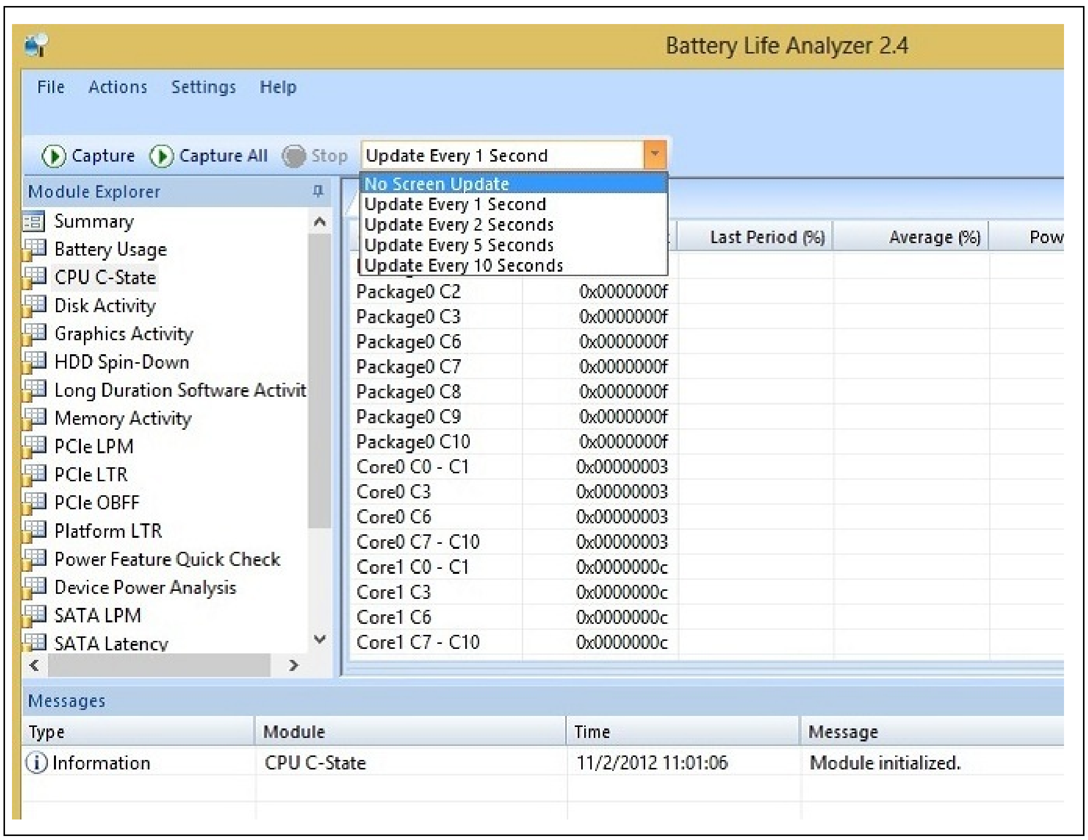
In Automatic mode, data is captured for all modules for a pre-determined amount of time (total amount of time this analysis takes is in about 5 minutes).

In Manual mode, the data is captured only for the module selected in the Module Explorer pane. Capture is started by pressing the Capture button and continues until the user stops capture with the Stop button. The user may select among the following possible data refresh rates using the drop-down selection in the toolbar: No screen updates, 1, 2, 5 or 10 seconds. If “No Screen Update” is selected, the data is not displayed until capture is stopped.

Note: Enabling run-time data updates during data collection will increase graphics activity and may result in increased CPU utilization and platform power consumption. Therefore it is recommended to use “No Screen Update” mode during data collection.

Note: “HDD Spin-Down Activity” and “Long Duration Software Activity” are only supported for data collection in Manual mode due to requiring considerably longer lengths of time to collect meaningful data.

Figure 2. Module Explorer Window



Hardware Analysis is supported by the following modules:

- Battery Usage
- CPU C-State
- HDD Spin-Down
- SATA LPM

- SATA Latency
- EHCI Analysis
- PCIe LPM
- PCIe LTR
- XHCI LPM
- XHCI LTM
- Platform LTR
- Battery Usage
- Device Power Analysis
- Memory Activity
- LPSS LTR
- Connected Standby

Software Analysis is supported by the following modules:

- Graphics Activity
- Disk Activity
- Software Activity Analysis includes Active Analysis, Driver Activity, P-State Analysis, and Idle Window Analysis)
- Long Duration Software Activity

In BLA version 2.4 a Power Feature Quick Check module added in order to enable checking various platform settings such as whether the “Connected Standby” is enabled and if the PEP driver is loaded (see Figure 3 below).

Figure 3. Platform Feature Quick Check

Power Feature Quick Check

Feature	Value	Information
Processor ID	0x206A0	
LPC ID	0x1C4F	
Memory Controller ID	0x104	
Operating System	Windows 7 - 64bit	
Connected Standby	N/A	Platform does not support CS
Intel PEP Driver	N/A	Platform does not support CS

Note: Users are encouraged to submit the output of this module, when filing bug reports as it will help BLA team to understand system configuration easily.

4.3.1 Enabling Periodic Poll and Analyzing Trends

Traditionally BLA hardware analysis modules collect only a running average value for the measurement period. If the screen refresh is enabled, it also displays the average value for last refresh period. Starting with BLA 2.2, users are able to enable periodic polling (poll period is specified in milliseconds) and collect desired number of measurement points. As of BLA version 2.5, this option is available for the following modules:

- CPU C-State
- LPSS LTR
- Device Power Analysis

4.3.2 Saving Results from Analysis

Analysis results can be saved from the "File" menu by selecting the "Save selected Module data" menu item. Hierarchical data can be saved as an HTML file, while all other data can be saved to the CSV format. Some modules, such as CPU C-State module collect history data, if module has history data collection enabled, additional options in the "File" menu will be presented to save the history data.

4.4 Connected Standby Analysis

The default behavior for BLA modules to stop data collection during connected standby and resume measurement when the platform comes out of connected standby state.

Starting with BLA version 2.5, there are two ways to analyze connected standby behavior. BLA Connected Standby module provides deep analysis of connected standby module, for details on how to enable data collection and use this module please refer to section 4.5.2.

In addition to Connected Stanby module, BLA CPU C State residency analysis module can also collect C state residency data when platform wakes up during connected standby to perform critical activities. Note that, users can perform this analysis using CLI or GUI throughout the connected stand by. In order to utilize this feature, C state analysis module should be executed with history limit and polling enabled mode so that the individual data points can be logged. Also, users should enable platform snapshot data collection flag in order to check for device LTR values and link states during Connected Standby. In order to learn how to use these features please read section 4.5.1.

Note: *When BLA C state analysis executed more than a few hours the counters may roll over. In order to prevent this behavior, user must enable history logging (see section 4.5.1).*

The amount of time the platform was in connected standby is not counted in the BLA execution duration that is, if user executes BLA for 120 seconds and during measurement platform was in connected standby mode for 180 seconds. This means BLA will finish execution 300 seconds from the start of the measurement.

As of version 2.4, BLA can execute CPU C-State, Software Activity Analysis, Device Power Analysis and Battery modules through connected standby concurrently (see Concurrent Capture Command in section 3.1.6).

4.5 Hardware Analysis

4.5.1 C-state Residency Analysis

In order to save energy when the CPU is idle it can enter low-power modes referred to as a C-States. Higher the C-state number (that is, deeper C-state), more logic within the CPU is turned off, enabling the platform to achieve more power savings. This power saving method comes at the cost of more time and energy required by the CPU to transition to the C0 state. Frequent C-state transitions will result in a net energy loss. To prevent such loss, the CPU implements automatic C-state demotion to avoid frequent transitions to deep C-states.

Note: C-state definition may vary from Processor to Processor. Table 2 is a guideline of what each C-state entails. For exact definition please refer to the *External Design Specification* for a specific Processor type.

Table 2. CPU C-states

C-State	Intel® Core™ i7 / i5 / i3 Processor Description	2nd / 3rd Generation Intel® Core™ i7 / i5 / i3 Processor Description
C0	CPU Active – Executing code	CPU Active – Executing code
C1	Halt, or Snoop Response	Halt
C2	N/A	Snoop response
C3	Core caches (L1/2) flushed.	Core caches (L1/2) flushed.
C6	Save CPU core state to SRAM and shut down CPU core voltage	Save CPU core state to SRAM and shut down CPU core voltage
C7	N/A	Last level cache (L3) flushed

When BLA starts, it will detect the platform type and adjust accordingly to display only the C-states supported on the platform.

Upon completion of C-state analysis, the tool will display (Figure 4):

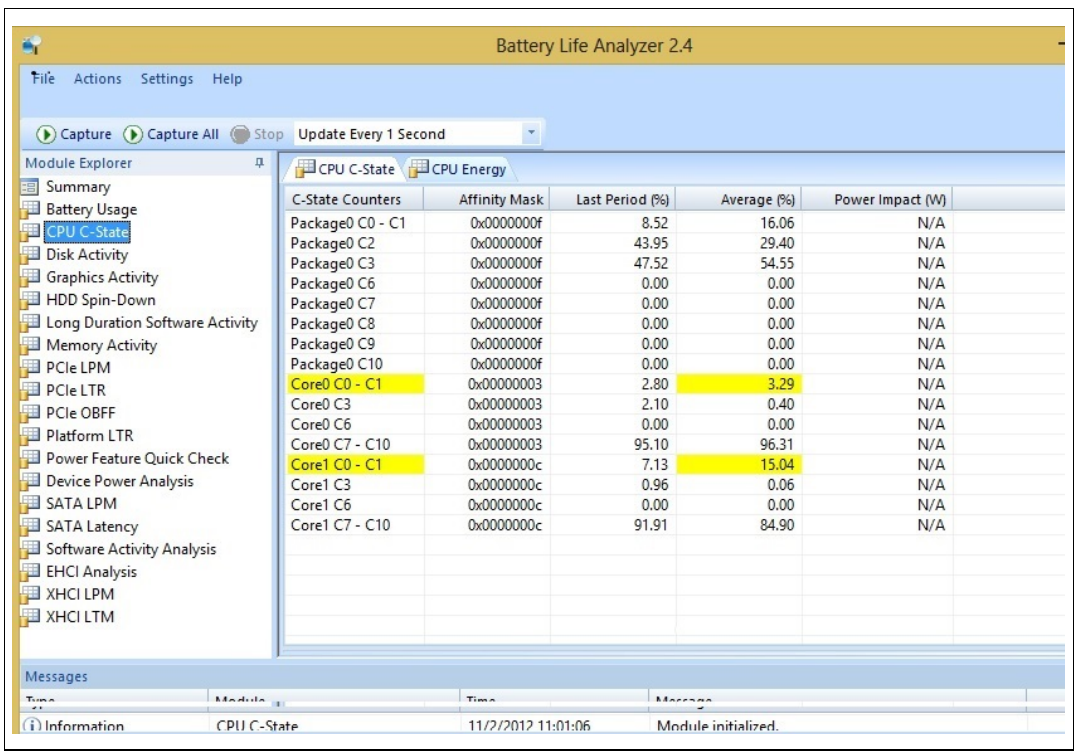
- Affinity Mask –the bitmap to the logical CPU that the OS recognizes.

- Last Residency Period percentage in each C-state –the period of the residency during the last update window. Only applies if runtime data updates are enabled. If data updates are turned off, this number will match the average residency period.
- Average Residency Period percentage in each C-state –the average residency period over the entire time of data collection. This field provides an estimate of the power impact of not staying in the deepest C-state where applicable.

The power impact of staying in the C0 (active) state is estimated in “Software Activity Analysis”. To maximize the battery life of mobile platforms, Intel recommends that deepest C-state residency in Idle should be greater than 95% per package and greater than 98% per core, and package residency in C0 should be less than 5% when the system is idle. These thresholds will not apply if the platform is being actively used, for example when applications are actively running.

In BLA version 2.2, a new CPU Energy tab added to CPU C-State Module that provides the CPU package power and DRAM power. Availability of these results depends on CPU and BIOS support. For each energy counter, BLA UI displays last period average power and a running average power since the start of the measurement.

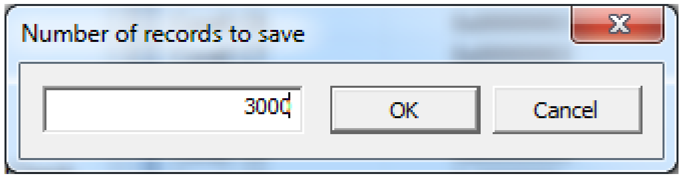
Figure 4. C-state Residency Module



In order to run BLA CPU C State Analysis in history enabled mode from the GUI user should set the following options:

1- Select CPU C-State Module and click on "Module Options" in the settings menu. Then, click on "Set data collection history limit" menu item and enter the number of records to save in the dialog box shown as below in Figure 5.

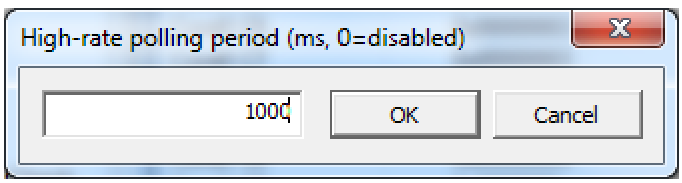
Figure 5. Setting history limit



The number of records to save option specifies the last n records to be saved during data collection.

2- After turning on the number of records to save option by setting it to more than one, user can set the polling period by accessing the "high rate polling period" menu item in "Module Options" in the settings menu as shown in Figure 6:

Figure 6. Setting polling period



Note that, this option is specified in milliseconds precision.

3- User can also select to enable platform snapshot data collection. Enabling this option will create a platform snapshot for PCIe LPM state, PCIe LTR values, Platform LTR values and SATA LPM state. This option is available only for logging all the snapshot data mentioned above in a text file.

4- After the measurement is compete, user can select to save the C-State history data and platform snapshot data in a .csv file from the File -> "Save data collection history" and File->"Save platform snapshot data" menu options.

In order to run BLA C-State module in history enabled mode from the CLI user should set the following options:

Savehist= <name of the history data file>

histlimit=<last n number of data points to save>

poll=<data collection period>

en_plat_data= <enable collecting platform snapshot data>

plat_data_loc=<path to the platform snapshot data file>

Example command:

BLA c cst:300,savehist="c:\test\All_cstatedata.csv",histlimit=3000,poll=100",en_plat_data=y, plat_data_loc="c:\test\plat_snapshot"

Note: *When BLA C state analysis executed more than a few hours the counters may roll over. In order to prevent this behavior, user must enable history logging.*

Note: *If during C state measurement the system transitioned to a suspend state, such as S3-Standby state, then after resume BLA will continue C State measurement however it will ignore the first sample collected after resume. In this scenario users will see 2 warning messages on the GUI as follows:*

"Suspend state resume detected. It is recommended that the application be reset"

"Simultaneous rollover of multiple counters (possible Sx.) Sample discarded at <date/time stamp>"

Also, the CPU C state residency data and the CPU energy data will ignore the first sample after resume and the average values will be calculated based on the system active state. Note that, Connected Standby analysis is different than the scenario described above, please refer to sections 4.4 and 4.5.2 to learn more about Connected Standby analysis.

4.5.2 Connected Standby Analysis

Important Note 1: This module requires BLA PEP driver (BLA specific Intel Power Engine Plug-in driver) in order to perform detailed analysis of the connected standby window. Please contact your Intel Application Engineer in order to obtain a copy of this driver.

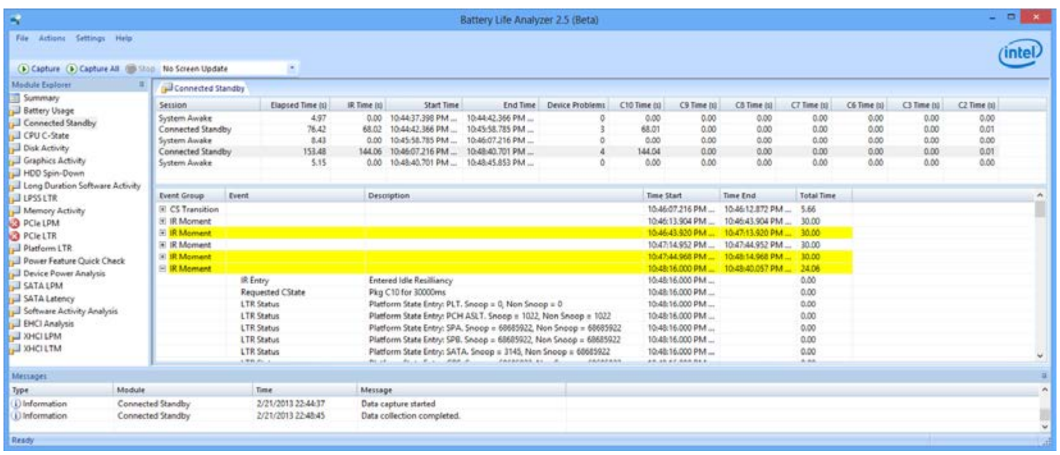
Important Note 2: If the time spent in C States (C10 Time, C9 Time etc. columns) are all measured 0, please follow the additional debugging steps described in section 4.5.2.1.

Using Connected Standby module users can analyze the details of connected standby window. During Connected Standby system may enter Idle Resiliency (IR) phase multiple times where system activity is minimized. BLA Connected Standby module provides visibility to each IR phase along with the important metrics that could impact the length of the IR phase or the deepest sleep state CPU can enter. Connected Standby module can be executed across multiple CS cycles and in parallel with the other BLA modules that can run concurrently (see section 3.1.6 for more information on how to use concurrent capture command).

In the BLA GUI the analysis result by Connected Stanby module are presented in a parent-child hierarchy. Each system active and connected standby window is listed in the parent window along with the following information:

- Elapsed Time: Total time within the individual system state
- IR Time: Applies to Connected Standby window only and represents the total time system was in IR phase during CS window.
- C State Residencies: Applies to CS entries only and represents the total time system was in the given C State during the CS window.
- Start Time: Start time of the associated entry.
- End Time: End time of the associated entry.
- Device Problems: Applies to CS window only and represents the total number of device related issues that impacted IR phase during CS session.

Figure 7. Connected Standby Module



See the above figure for an example output of Connected Standby module. Clicking on each row in the parent window displays the child events. For example, this measurement had two CS sessions. Clicking on each CS session, will display the events happened during that session. In the child window user can click on the "+" sign to expand details of that event. Event details are composed of subsystem level LTR values, C state residencies observed during the event and device issues. Also, the IR phase start and end time stamps and the total time taken are summarized for each IR phase in the child window. Note that, IR Moment event groups that BLA has detected a device problem are marked with yellow.

4.5.2.1 Additional Debugging Steps for Connected Standby Analysis

If the time spent in C States (C10 Time, C9 Time etc. columns) are all measured 0, then user can try reducing the diagnostic time interval by adding the following registry key:

[HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\hswultpep_bla\Diag]
"IdleDurationDbgThresholdInMS"=dword:<value>

Default value of this parameter is 1000ms. User can try reducing it shorter intervals such as 10ms:
"IdleDurationDbgThresholdInMS"=dword:0000000A

After the change, reboot the system and redo the measurement. If there is no change in C State residencies, then user can use BLA Device Power Analysis module to check if UART, I2C, SDHC, GFX, XHCI controllers does not do any D state transitions. If that is the case, then user should make sure correct RTD3 configuration for the device is present and device can perform D state transitions properly.

Alternatively, user can perform further CS Analysis using BLA CPU C–State module by enabling (1) history limit, (2) poling period and (3) platform snapshot data collection as described in section 4.5.1. This analysis will show the C state residencies, LTR and LPM information during CS with less frequently collected samples.

4.5.3 HDD Spin-Down analysis

On systems with traditional storage methods, the HDD draws the least amount of power when it's in the standby state (spun down) and remains in that state. Software activities, like frequent reads and writes to the registry or to files and flush operations, can reduce the percentage of time the HDD is spin-down and have a negative effect on power consumption.

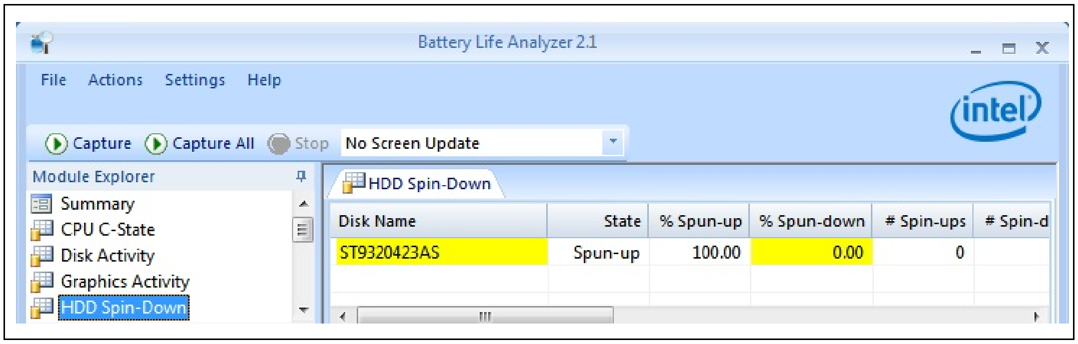
When BLA starts, it detects the types of disk drives present on the platform. Operations on drives determined to be a Solid State Drive (SSD) are not supported and will not be shown in the device table. Once detection is completed, information about the disk type, port and disk controller will be displayed in the output window. The recommended data collection window is 1-2 hours or longer.

Upon completion of the data collection, the tool will display the following (see Figure 8):

- Name of the disk drive.
- The last known power state.
- Percentage of time HDD spent in Active (Spun-Up) and Standby (Spun-Down) states.
- Number of spin-up and spin-down instances.
- Number of durations of spin-downs of various intervals (0-1 minute, 1-2 minutes, etc.).
- Name of the disk controller and port used.
- Estimate of the power impact of not staying in standby state.

Note: For platforms with an SSD attached, the HDD Spin-down module analysis will ignore the SSD as the concept of spin-down does not apply to it.

Figure 8. HDD Spin-Down Module



4.5.4 PCIe LPM Analysis

PCIe devices should be designed for maximum power management and should power down the PCIe Link as often as possible. If a device supports Active State Power Management (ASPM), it is imperative that those states are enabled. It is also important that a device allows the link to enter L1 or L0s as much as possible. *PCI Express* Base Specification Revision 2.1* defines several low power link states for when a device is in the Active D0 state, which allows individual serial links to have power reduced if the number of transactions on the link reduce (Table 3).

Table 3. PCIe* Link Power Management States

Link state	PCIe* Device State	Description
L0	D0	Link is fully active
L0s	D0	Low resume latency state intended for aggressive use during short intervals of Idle periods. All main power supplies, component reference clocks, and component internal PLLs are active, but transceiver circuitry and link layer logic is power gated
L1	D0, D1, D2, $D3_{hot}$	Higher resume latency state intended to be used during longer Idle periods. All main power supplies and component reference clocks are active, except the ones gated by CLKREQ#. Component internal PLLs are shut off. Transceiver circuitry and link layer logic is power gated
L1.1	D0, D1, D2, $D3_{hot}$	L1.1 is a substate of L1 state where both upstream and downstream ports must maintain common mode while they can turn off electrical idle detection circuitry.
L1.2	D0, D1, D2, $D3_{hot}$	L1.2 is a substate of L1 state where both upstream and downstream ports may power-down any active logic, including circuits required to maintain common mode.
L2	$D3_{cold}$	This link state is intended to for $D3_{cold}$ with AUX power support and WAKE# signaling for wake support.

Link state	PCIe* Device State	Description
L3	$D3_{cold}$	Link is off. No AUX power or wake support.

Upon completion of this analysis, the tool will display the following information (see Figure 9):

- The name of the PCIe* device as well as the root port location based on the number of PCIe ports supported on the platform.
- Percent of time each link spends in L0, L0s, L1, L2, and L3 states.
- Percentage of time the link is Active, Down, or Retraining.
- Root Port (RP) and End Point (EP) capabilities – RP and EP capabilities do not always match, however it's important to enable matching pairs on RP and EP side.
- Current link power management states that are enabled for each device.
- Estimate of the power impact for not staying in L1 state all the time. This number only includes the impact to the chipset and doesn't include the impact to the end-point device.
- Starting with BLA version 2.4, also the L1.1 and L1.2 substates capability discovery is included in the PCIe LPM module (see Figure 10). These are new states introduced in PCI Express 3.0 L1 PM Substates with CLKREQ ECN. Entry and exit from L1.1 and L2.2 substates happen to and from L1.0 substate directly and L1.0 substate corresponds to the conventional L1 link state.

Figure 9. PCI Express* LPM Analysis

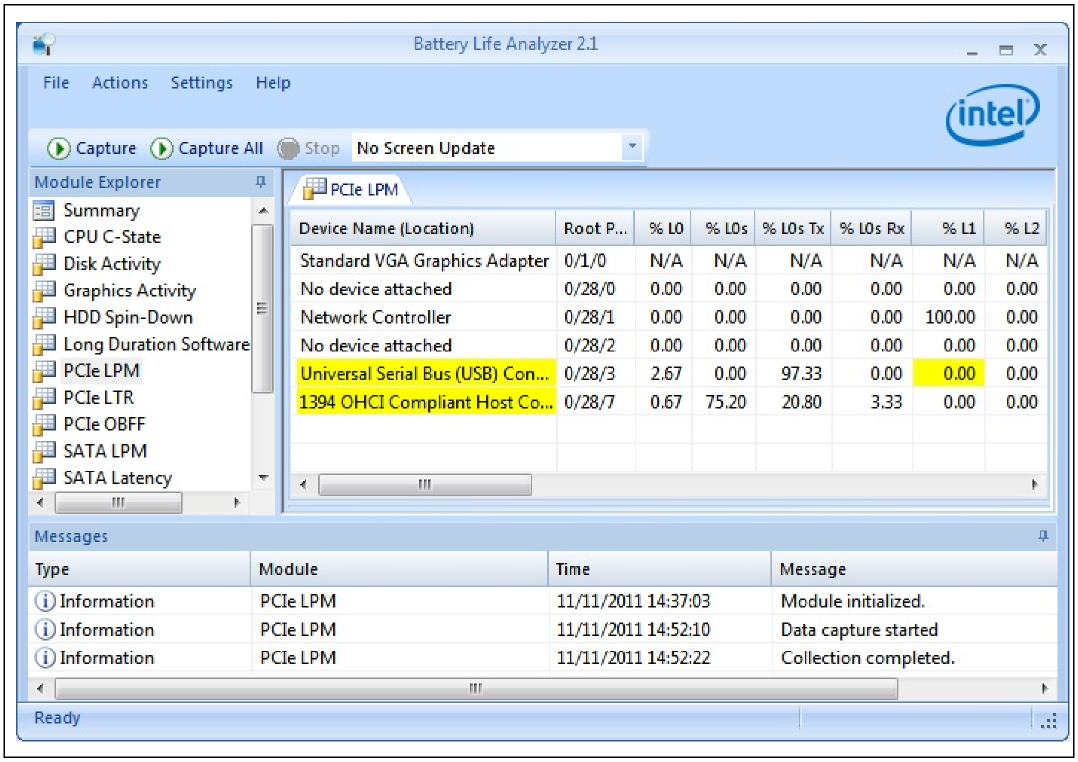
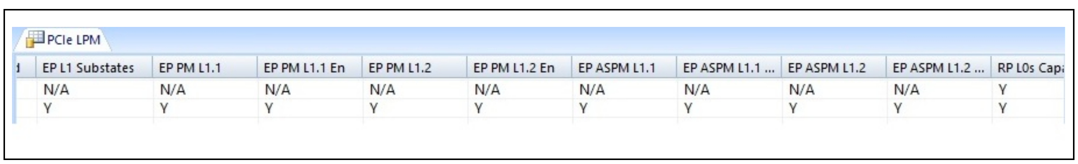


Figure 10. PCIe L1 Substates Capability Discovery



4.5.5 PCIe LTR Analysis

The Latency Tolerance Reporting (LTR) is a mechanism that enables PCIe End Points to explicitly convey their service latency requirements for interrupt and memory reads / writes to the root port. A platform will be able to optimize its power savings behavior based on the requirements provided by devices. In general, when devices have short latency tolerance requests, the platform does not enter into a deep power saving state to satisfy the requirements. To avoid excessive platform power impact, it is required that End Points implement an adequate power policy and optimize the latency tolerance messages based on its workload.

Please see the Energy-Efficient Platforms and PCI Express 2.0 Latency Tolerance Reporting ECN in Reference Documents for more detailed explanations of LTR.

Upon completion of the analysis the tool will display (Figure 11):

- The Device Name
- Root Port (RP) location – PCI bus, device, function number of the root port
- Device Class Code of the End Point
- If the Root Port is LTR capable
- If the Root Port has LTR enabled – Provided the Root Port is capable, this feature is configurable by double-clicking on the cell to enable/disable LTR functionality.
- If the End Point is LTR capable
- If the End Point has LTR enabled or not – Provided the End Point is capable, this feature is configurable by double-clicking on the cell to enable/disable LTR functionality.
- Flags indicating if the no-snoop and snoop LTR override enabled. If the device does not provide valid LTR messages, override values are used.
- Flags indicating, if the no-snoop and snoop LTR force override enabled. LTR values from the device are ignored and override LTR values are used.
- Snoop/no-snoop LTR override values. When the override enabled according to the override and force override flags, these LTR values are used.
- Flag indicating if LTR message received. This flag indicates if the platform received any LTR messages from the device, ensuring the LTR communication with the device is operational even when there is no LTR requirement for the device.
- A histogram of latencies reported of snoop and no-snoop requests
- Maximum and minimum of the snoop and no-snoop LTR values
- A histogram of latency tolerance reported for snoop/no-snoop traffic by the End Point – These histograms show the duration spent with each range of LTR values sent by the End Point. User can use these histograms to determine if the LTR policy of the device is adequate for the workload or not.
-
- Max snoop/no-snoop Latency Value set to the LTR capability structure of the End Point – These registers should be initialized by the BIOS and/or OS once their LTR support is in place. The BLA allows users to modify these register values by double clicking on the specific field so that users can test LTR capable devices before complete BIOS/OS support becomes available (Figure 12). When modifying the Max Snoop Latency, a user may alter either the Duration or the Latency and Scale values. BLA currently does not support altering of all three at once).

Note: These values are not recommendations from Intel, but instead are used to identify potential problems. They may not be actual problems.

Figure 11. PCI Express* LTR Analysis

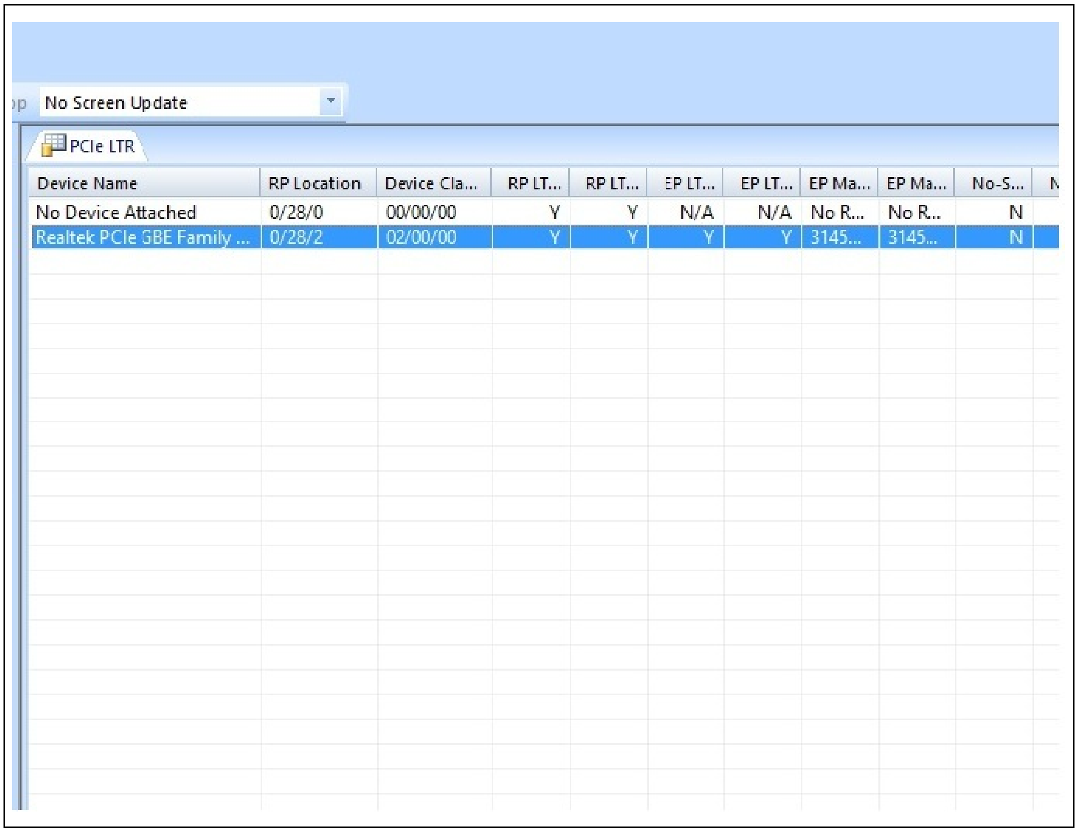
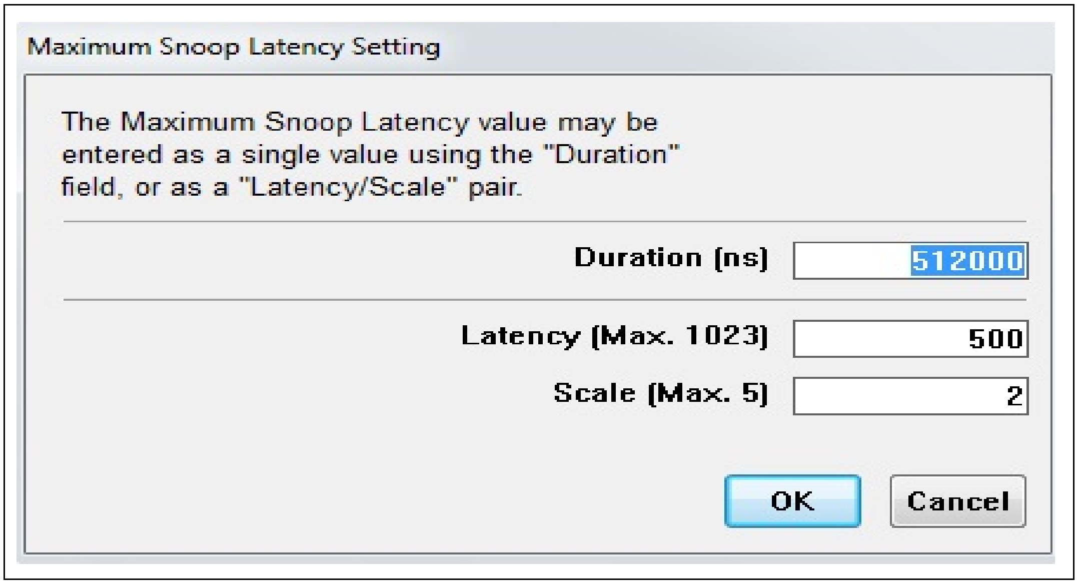


Figure 12. PCI Express* LTR Max Snoop Latency Setting



4.5.6 SATA Link Power Management Analysis

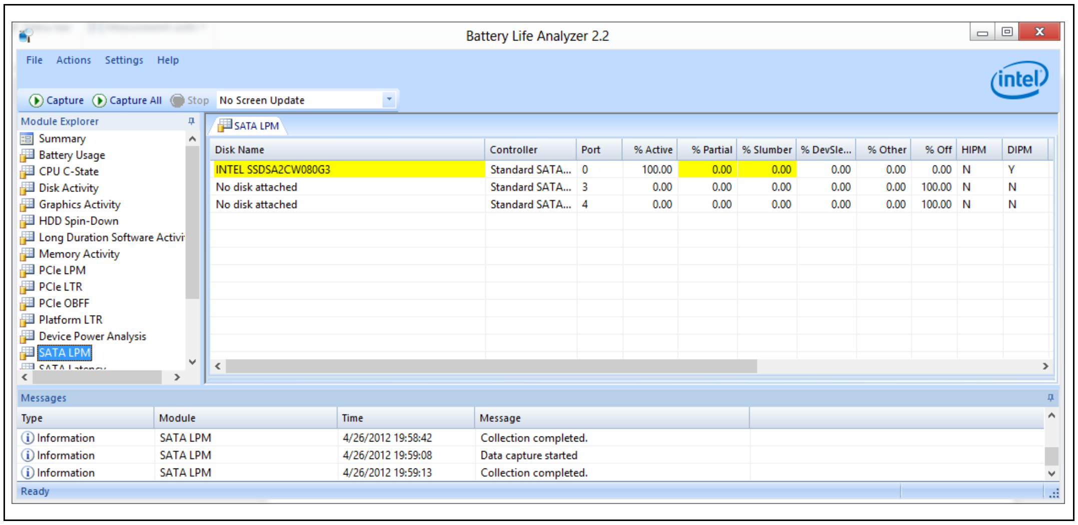
SATA devices can save power by allowing the link to enter the Partial or Slumber link power states while the interface is Idle. SATA defines three PHY layer power states while there is no traffic on the SATA link:

- Active - PHY logic and PLL are both on and active
- Partial – PHY logic is powered, most of internal circuitry is powered off in the host and device, but not the host PLL. Exit latency does not exceed 10 µs. This state is mainly used to save power when a device is processing commands.
- Slumber – PHY logic is powered, most of internal circuitry is powered off in the host and device, including the host PLL. Exit latency can be up to 10 ms; this state is used to save power between commands.
- DevSleep- Devlseep is a new lower latency low power mode in which both the host and device PHY can be completely powered off, but still maintain an exit latency similar to Slumber state than to a full shutdown.

If a SATA device supports those states, they should be enabled and the device should spend more than 85% in the deepest state. This module is capable of measuring the state residency of the SATA link. Upon completion of this analysis, the tool will display (Figure 13):

Note: LPM is not supported if ESP is enabled or if HPCP is enabled and MPSP is not set.

Figure 13. SATA LPM Analysis



4.5.7 SATA Latency

This module measures the LPM state entrance latency. (Data collection can take up to 3 minutes.) Upon completion of this data collection, the tool displays the following (Refer to Figure 14):

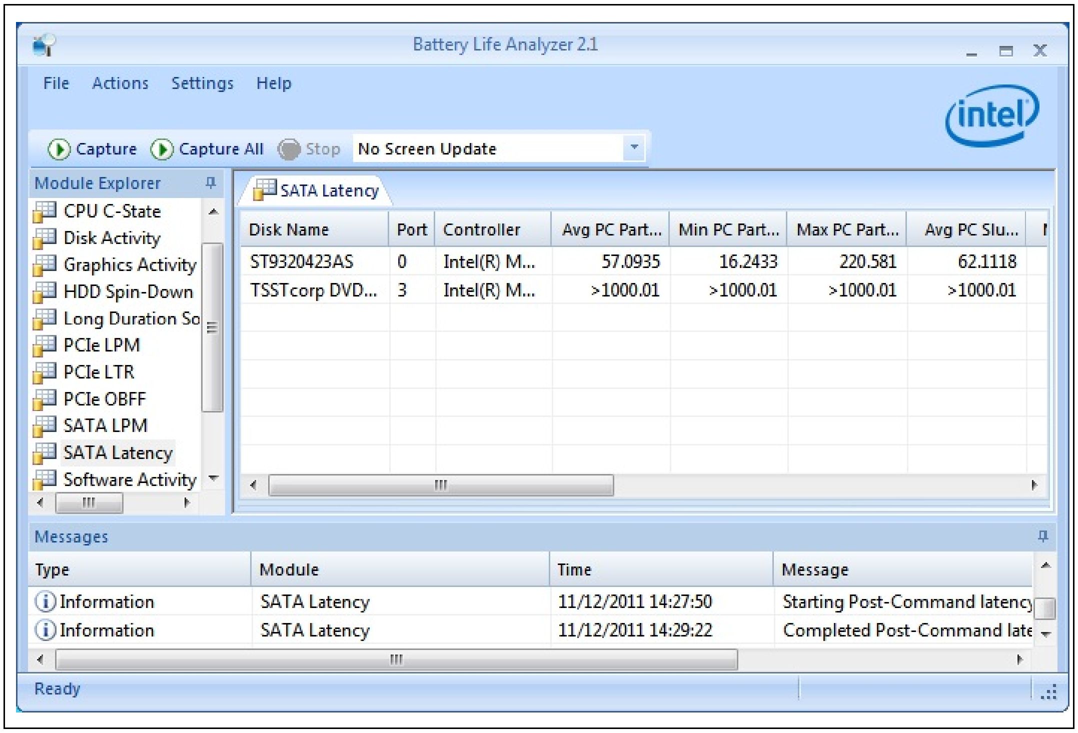
- In-command LPM state entry latencies
  - — Average, Minimum, and Maximum latency to enter Partial state during command execution
  - — Average, Minimum, and Maximum latency to enter Slumber state during command execution
- Post-command LPM state entry latencies
  - — Average, Minimum, and Maximum latency to enter Partial state after command execution
  - — Average, Minimum, and Maximum latency to enter Slumber state after command execution

Note: If the transition to the LPM state didn't happen during the command execution, BLA shows the time spent to complete the command prefixed with a greater than sign, ">", indicating that the command took longer than the maximum time measured. (BLA monitors LPM state transition up to 1 second after command completion.)

Note: The measurement of LPM state entry latency requires access to the media in each drive. If the system under the test has a CD-ROM or DVD drive, please insert media in the device (without content protection) before you run SATA latency analysis.

Note: DIPM must be enabled in attached devices to obtain valid results. It is recommended that the Intel RST drivers be used and that the system is idle during testing. If using the Microsoft AHCI drivers under Windows 7, DIPM support may be enabled by selecting the “Power Saver” OS power policy.

Figure 14. SATA LPM Latency Detection Output



4.5.8 EHCI Analysis

The USB bus is a polled bus with data and control transactions initiated by the host, not the device. Enhanced Host Controller Interface (EHCI) supports USB high-speed signaling at 480 Mbps. Each EHCI controller can support up to 8 USB ports. The Universal Host Controller Interface (UHCI) supports low-speed and full-speed USB devices at 1.5 Mbps and 12 Mbps respectively. Each UHCI controller can support up to 2 USB ports. Starting with the Calpella platform, UHCI was replaced with the Rate Matching Hub (RMH) to support low and full-speed devices. The RMH is connected to each EHCI controller and converts low and full-speed traffic into high-speed traffic. EHCI Analysis module looks at all UHCI and EHCI controllers and USB devices attached to them in an effort to analyze activity and power management features enabled for each. The analysis window will specify if the host controller is running or stopped and if the periodic and asynchronous scheduler is on/off. If the host controller is on, the tool will specify which device is keeping it running. USB devices designed with power management in mind usually do not keep the schedulers running.

EHCI provides support for two transfer types: asynchronous and periodic. Asynchronous schedules are used for control and bulk transfers, while periodic schedules are used for isochronous and interrupt transfers. A periodic schedule is based on a time-oriented frame list that services transfers during allocated time. An asynchronous schedule is a circular list of work items that get serviced in first come first serve fashion. High speed and low speed asynchronous schedules can generate traffic every 8-16 µs even when there is no data being transferred.

Control transfers are used to configure devices when attached. Bulk transfers are used to move large amounts of data (for example flash storage or communication devices). Interrupt transfers are used to convey human perceptible data or change in status (for example keyboard and mouse). Isochronous transfers are used to move fixed amount of bandwidth with a pre-negotiated time constraint.

EHCI Periodic Prefetch, also known as Prefetch Based Pause or Adaptive Activity ScoreBoard (AASB), is a feature that enables host controller hardware to safely prefetch information from periodic schedules to improve system memory access behavior.

Note: Not all devices support the Selective Suspend feature.

4.5.8.1 Basic Debug Guidance

This section provides best practices to employ when debugging USB device related issues. Note that, this is not intended to be the only means for debugging, but rather a guideline to help start the process.

The fastest way to identify USB devices that may be keeping a platform from entering suspend states is to examine the Selective Suspend duration for each host controller. The goal of this analysis is to find a host controller with the largest deviation from 100% for the "% Suspended" in the Selective Suspend. With the "% Suspended" value close to 100, it is safe to conclude that all devices on this specific host controller are okay.

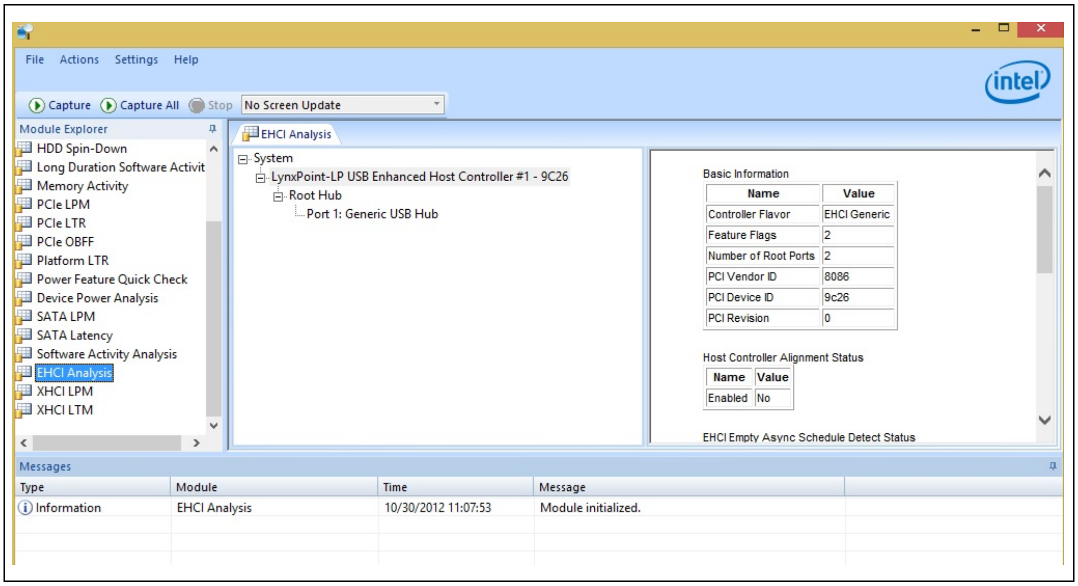
When a host controller with a low "% Suspended" state is found, the next step is to examine individual devices on the host controller. The specific data of interest for each device is the Selective Suspend field for the "% Suspended". If a device shows a low "% Suspended" it is possible this device is keeping the system from entering a Selective Suspend state.

It is possible that there are cases where no device for a host controller shows a significantly low Selective Suspend "% Suspended" value. In such cases, it is important to identify devices that may be entering and exiting Selective Suspend frequently (i.e., 100 ms of activity every 10s = 99% selective Suspend). Despite a devices' relatively high Select Suspend residency, the constant switching can cause a low residency at the host controller level. For these cases, search for a device with a Selective Suspend "% Suspended" residency not at 100%.

Once a host controller and device have been identified, the next step is to determine the impact to battery life. Examine the type of activity scheduler the host and device are utilizing through the Periodic and Asynchronous Schedule fields. If the

Asynchronous Schedule field has a high percentage for "% Running" then the overall impact is higher than the Periodic Scheduler's impact.

Figure 15. EHCI Analysis



4.5.9 XHCI Link Power Management Analysis

XHCI LPM (Link Power Management) enables a link to be placed into a lower power state when the link partners are idle. The longer a pair of link partners remain idle, the deeper the power savings that can be achieved. The supported Link Power States when the link is operating in USB 3.0 mode are shown below:

- U1 – Link Standby with Fast Exit. Entered and exited via hardware autonomous control.
- U2 – Link Standby with Slower Exit. Entered and exited via hardware autonomous control.
- U3 – Suspend. Entered only under software control and is directly coupled to the attached device's state.

The supported Link Power State when the link is operating in USB 2.0 mode is shown below:

- L1 – Link sleep state similar to L2 (below) but supports finer granularity in use. When in L1, the link state is identical to L2.
- L2 – Link suspend state.

This module supports XHCI LPM state analysis of the devices directly attached to an xHCI controller operating in USB 2.0 or 3.0 modes. Upon completion of this analysis, the tool will display the following information (Figure 16).

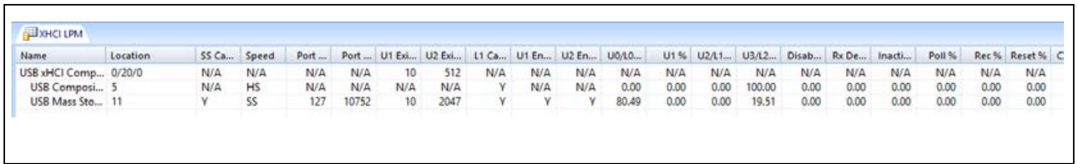
For xHCI host controller:

- The name of the xHCI host controller
- PCI bus location of the xHCI host controller
- U1/U2 device exit latency values from Structural Parameters 3 (HCSPARAMS3) register

For each USB device directly attached to the xHCI root port:

- The name of the device
- The port number the device is attached to
- SS Capable (whether the device is capable of running at Super Speed)
- Speed of the device (Current running speed of the device : LS for Low Speed, FS for Full Speed, HS for High Speed, SS for SuperSpeed)
- xHCI host controller root port settings
- U1 and U2 Port Timeout – Timeout value for U1 and U2 inactivity timer. These columns pertain to the USB3 devices. User can modify the settings by double clicking on the cell (Figure 17, Figure 18).
- Device LPM capabilities
- U1 and U2 Exit Latency – Time it takes to exit from either U1 or U2 link state. These columns pertain to the USB3 devices.
- L1 Capability – Indicates the device supports the LPM protocol. This column pertains to the USB2 devices.
- Device LPM settings
- U1 and U2 Enable – Indicates if the device is enabled to initiate U1 or U2 transition. These columns pertain to the USB3 devices. User can toggle the status by double clicking on the cell.
- Percent of time each link spends in U1, U2/L1, and U3/L2 states
- As of BLA version 2.5, if the device is not in one of the power saving states, BLA can also display detailed state of the device instead of indicating device is an unknown state, see columns following U2/L3% in XHI LPM module. These additional states are Disabled, Rx Detection, Inactive, Polling, Recovery, Reset, Compliance, Test and Resume States. In order to understand the meaning of these states please refer to the Universal Serial Bus Revision 3.0 Specification.

Figure 16. XHCI LPM Analysis


XHCI LPM

Name	Location	SS Ca...	Speed	Port ...	Port ...	U1 Exi...	U2 Exi...	L1 Ca...	U1 En...	U2 En...	U0/L0...	U1 %	U2/L1...	U3/L2...	Disab...	Rx De...	Inacti...	Poll %	Rec %	Reset %	C
USB xHCI Comp...	0/20/0	N/A	N/A	N/A	N/A	10	512	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	
USB Composi...	5	N/A	HS	N/A	N/A	N/A	N/A	Y	N/A	N/A	0.00	0.00	0.00	100.00	0.00	0.00	0.00	0.00	0.00	0.00	
USB Mass Sto...	11	Y	SS	127	10752	10	2047	Y	Y	Y	80.49	0.00	0.00	19.51	0.00	0.00	0.00	0.00	0.00	0.00	

Figure 17. XHCI LPM U1 Timeout Setting

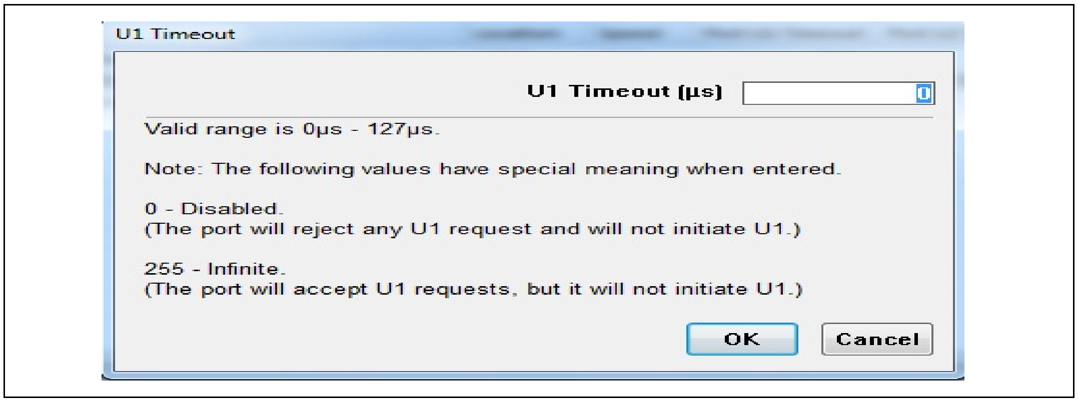
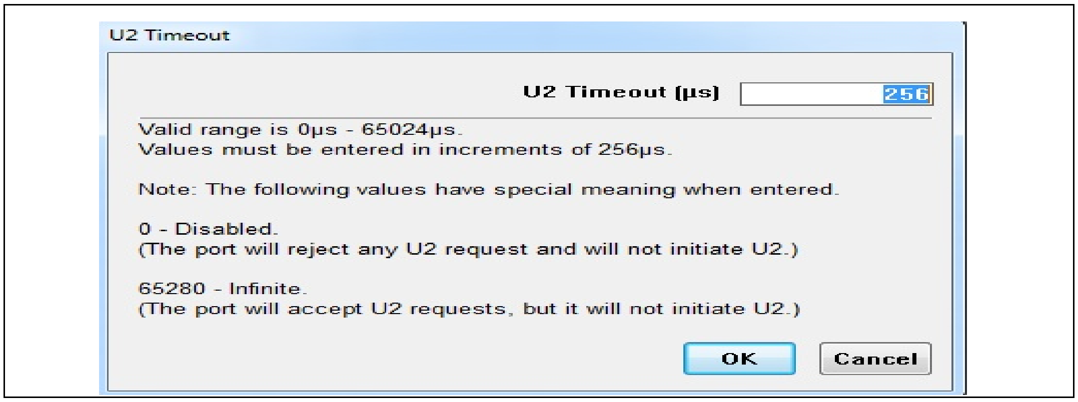


Figure 18. XHCI LPM U2 Timeout Setting



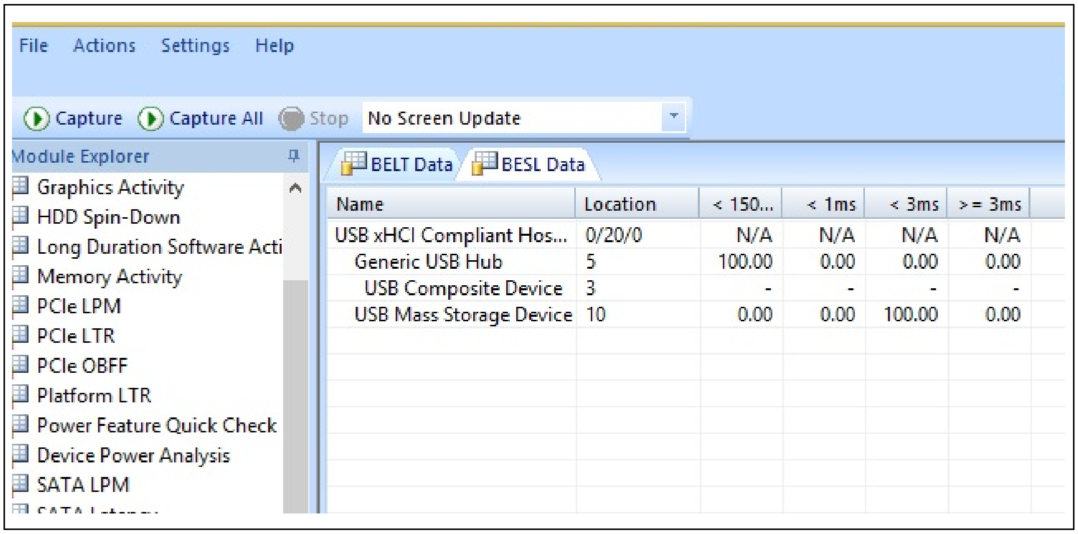
4.5.10 XHCI LTM (Latency Tolerance Messaging) Analysis

XHCI LTM (Latency Tolerance Messaging) module analyses the latency tolerance values set for the device or communicated from the device to PCH for xHCI USB controller. For USB 3.0 devices, BELT (Best Effort Latency Tolerance) values are tracked under BELT data tab. For USB 2.0 devices, BESL (Best Effort Service Latency) values are tracked under BESL data tab. Upon completion of this analysis, the tool will display:

- The name and location of the xHCI host controller or Device (indented) and port that the device is connected to.
- LTM capable and LTM enable flags for USB 3.0 devices

Histogram of LTM values

Figure 19: XHCI LTM Analysis



4.5.11 Platform LTR Module

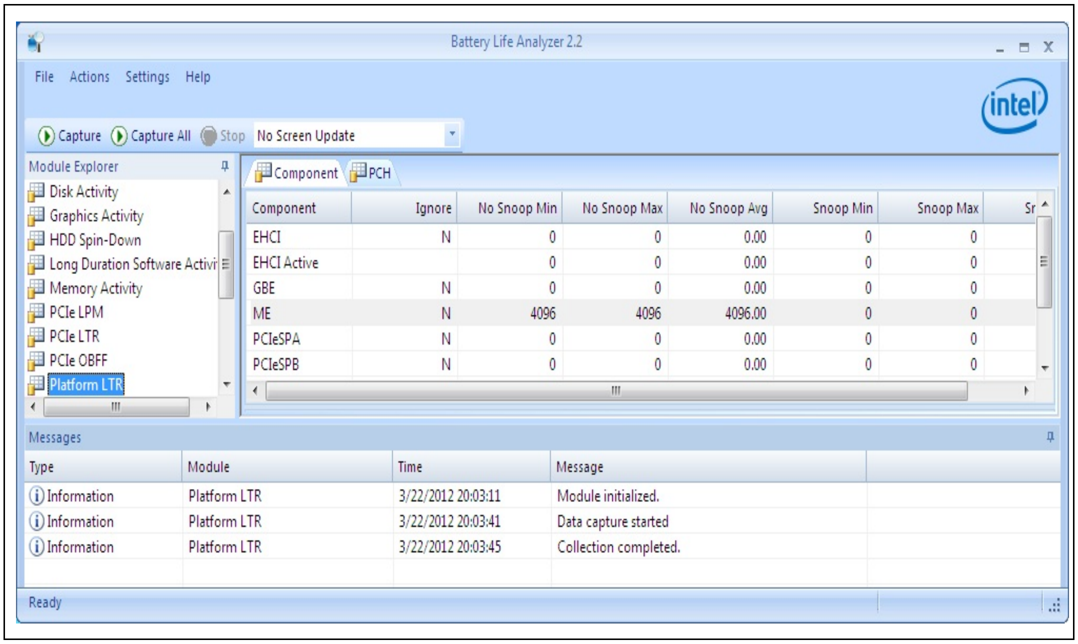
Platform LTR module provides the aggregated LTR values by PCH for various subsystems. For example, for the SATA line item the displayed LTR values are the aggregation of LTR values from all SATA devices attached to the subsystem. Upon completing the measurement these fields are populated on the Component tab:

- Ignore flag
- Minimum, maximum and average snoop and no-snoop LTR values.

All the data displayed by this module is read only and LTR values are displayed in nanoseconds precision. Note that, the XHCI Active and EHCI Active values are the values that are programmed to be used when those controllers are indicating that they are in active state.

The module also displays the PCH aggregated LTR value for all subsystems on the PCH tab. Upon completion of the analysis, minimum, maximum and average LTR no-snoop, fast snoop and slow snoop values are displayed on the PCH tab aggregated for all the PCH devices.

Figure 20: Platform LTR module



4.5.12 Battery Usage Module

The purpose of battery usage module is to provide user with an easy and consistent way to measure battery degradation even across standby mode. When the module loads the tool will display general information for the attached battery devices, specifically, these fields will be populated:

Battery Name: Name of the attached battery device.

Current Voltage: Battery voltage across the battery terminals in Volts.

Design Capacity: The theoretical full capacity of the battery when the battery is brand new in mWh.

Current Capacity: Current battery fully charged capacity in mWh.

Begin Charge and Last Charge fields will be populated with current battery charge level in mWh.

During data capture, if the system enters and exits standby mode, the module will not reload, simply will continue to run. Upon completion of this analysis, the tool will display:

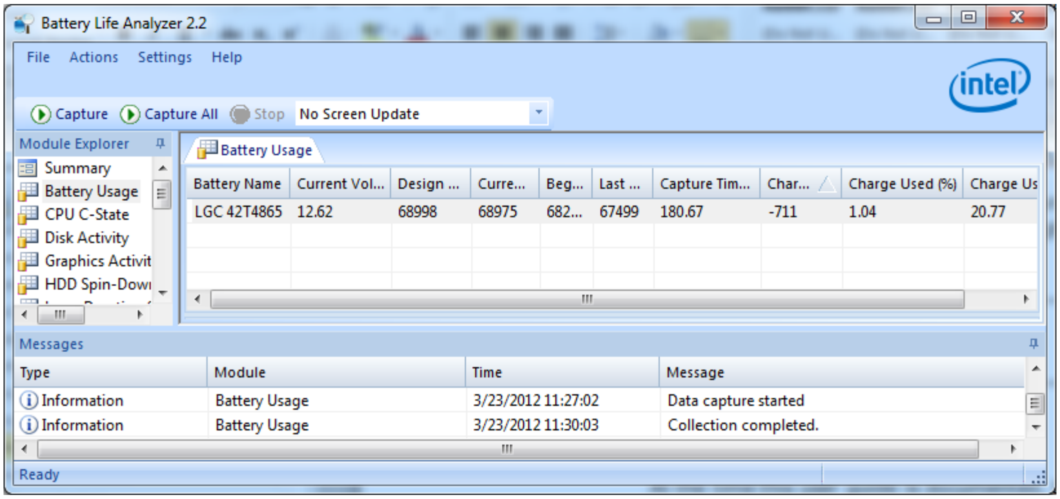
Begin Charge and Last Charge fields will be updated according to the battery reading when the measurement started and completed, respectively.

Capture Time: Duration of the measurement in seconds.

Charge Used: Difference between Last Charge (end of measurement) and Begin Charge (start of the measurement) in mWh.

Charge Used %: Ratio of Charge Used during measurement against Current Capacity
Charge Used / hour %: (Theoretical) Straight extrapolation of Charge Used % to hour.

Figure 21. Battery Usage Module



4.5.13 Device Power Analysis Module

Important Note: Please make sure to read the Manual Steps in section 4.5.13.1 before measuring PSR residency

The purpose of this module is to provide device sleep state analysis (D-States). With the ever increasing battery life requirements of the mobile platforms, the device power management is a critical factor in platform power management. Runtime D3 (RTD3) support allows devices to be completely powered off (D3 Cold) while the system is in S0 state. BLA Device Power Analysis module helps user to identify and characterize device power management capabilities. Upon completion of the measurement, module displays this information:

Controllers: Name of the controller that device is connected

Device Name: Name of the device

Bus;Dev;Func : Identifier of the device

Port#: Port id that device is connected

Hub#: Hub id that device is connected

D0%: D0 state residency

D1%: D1 state residency

D2%: D2 state residency

D3 Hot% : D3 Hot state residency

D3 Cold% : D3 Cold state residency

Removed% : Percentage of the time the device was removed during measurement. Note that, if the device was not attached during the beginning of the measurement, its PM statistics is not included in the results.

D0 Avg. Latency: Average latency of the D0 state entry

D1 Avg. Latency: Average latency of the D1 state entry

D2 Avg. Latency: Average latency of the D2 state entry

D3 Hot Avg. Latency: Average latency of the D3Hot state entry

D3 Cold Avg. Latency: Average latency of the D3Cold state entry

Hot Plug: Device Hot Plug capable

D0 Enabled : Whether the device supports D0

D1 Enabled : Whether the device supports D1

D2 Enabled : Whether the device supports D2

D3 Enabled : Whether the device supports D3Hot

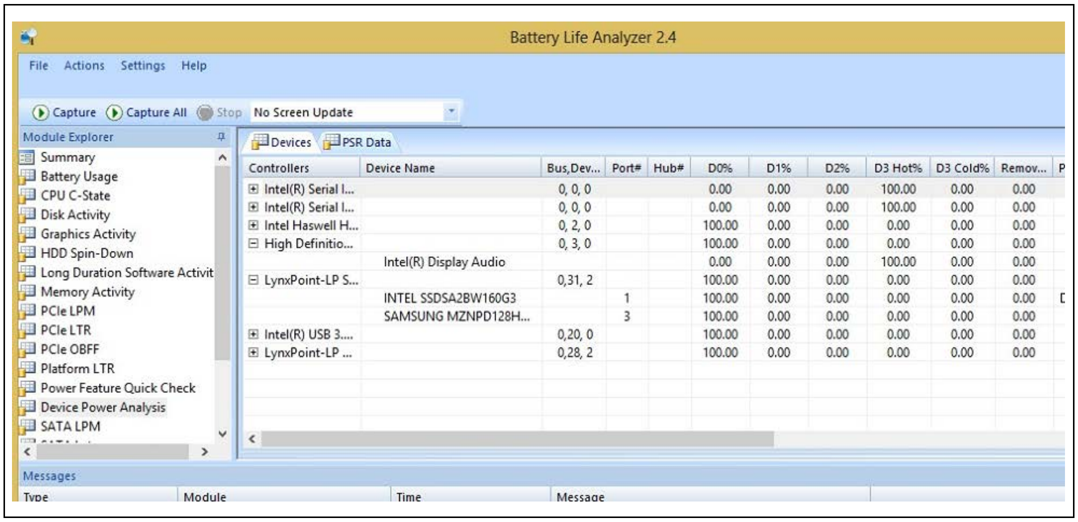
Pwr Resource: The mapping of the physical power resource to the device. Note that, this relationship is important for BLA to be able to determine D3Cold% and it is BIOS dependent.

Using the module settings menu, user can save detailed information collected during the measurement by enabling logging in .csv format and .etl format (related source ETW events captured during measurement).

In order to help testing RTD3 on SATA, BLA provides two options to set SATA idle timeout for AC and DC in “Settings”->”Module Options “ menu.

Also user can select save ETW events to file by turning on “Log ETW events to file” menu option in “Settings”->”Module Options “.Menu option.

Figure 22. Device Power Analysis



4.5.13.1 PSR Analysis

Important Note: Please make sure to read the Manual Steps in this section before measuring PSR residency

As of BLA 2.4, Device Power Analysis module is also able to analyze Panel Self Refresh (PSR) residency and capability in the “PSR Data” tab. When the user selects “capture”, module display PSR capability in the Messages pane, such as, “ PSR function initialized and reset” or “No PSR Device present on target system”.

User can enable history data collection by setting the “Polling rate for collecting PSR data” in “Settings” -> “Module Options” menu.

Similar to the other BLA analysis, once the capture is complete, user will see a running average of PSR% and the average of the last refresh period if screen refresh was enabled. Also, if polling rate is enabled, the results can be saved in .csv file with time stamps using “File”->”Save Data collection History” Menu option.

In order to be able to measure PSR residency user must follow the manual steps described below before the measurement. Upon completing the measurement the settings must be reverted to original, see the full instructions below:

<u>Manual Steps before the measurement</u>

In order to be able to measure the PSR residency for a given workload user need to disable RC6 sleep state. This can be achieved by performing one of the two options below.

Option 1: Disable RC6 in BIOS/CMOS settings

1. Enter BIOS/CMOS settings by pressing F2/Delete at boot time

2. Follow the path: Intel Advanced Menu -> Power & Performance -> GT - Power Management Control -> RC6(Render Stand by) –

3. Set the RC6 to Disabled State.

4. Save the settings and reboot.

Option 2: Registry Key Change

1. Boot to OS. Open Registry Editor.

2. Go to location: HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\Class\{4D36E968-E325-11CE-BFC1-08002BE10318}\000 & \001 (if available)

3. Find "FeatureTestControl" registry key. Copy the value of this registry key to a safe location as you will need this original value after the measurement. Then set bit 7 to 1 (note that, counting of the bit positions start with 0). Make sure only to change bit position 7 keeping the remaining bits in their original value.

4. Reboot

Manual Steps after the measurement

If Option 1 was selected before measurement then: Enable RC6 in BIOS/CMOS settings

1. Enter BIOS/CMOS settings by pressing F2/Delete at boot time

2. Follow the path: Intel Advanced Menu -> Power & Performance -> GT - Power Management Control -> RC6(Render Stand by) –

3. Set the RC6 to Enabled State.

4. Save the settings and reboot.

If Option 2 was selected before measurement then: Registry Key Change

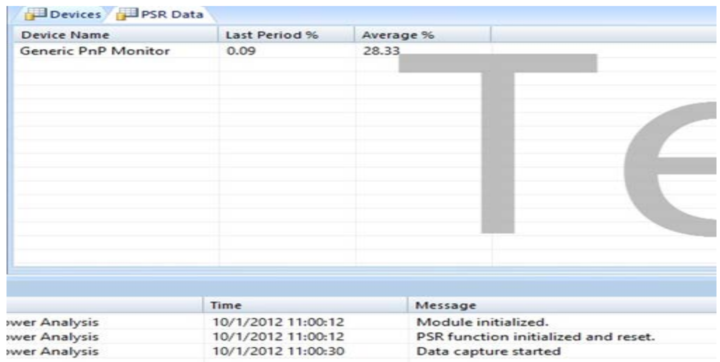
1. Boot to OS. Open Registry Editor.

2. Go to location: HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\Class\{4D36E968-E325-11CE-BFC1-08002BE10318}\000 & \001 (if available)

3. Find “FeatureTestControl” registry key and set bit position 7 to 0. Alternatively, just set the value of this key to the original value you saved before the measurement in step 3.

4. Reboot

Figure 23. PSR Analysis



4.5.14 Memory Activity Module

Understanding and optimizing memory activity is an important component of power management analysis. BLA Memory activity module provides detailed information about the memory bandwith of a workload and makes it easy to characterize the memory activity over time. This module provides memory read and write activity by memory channel. Upon completion of the measurement, module displays this information (Figure 24):

Channel: Memory Channel ID

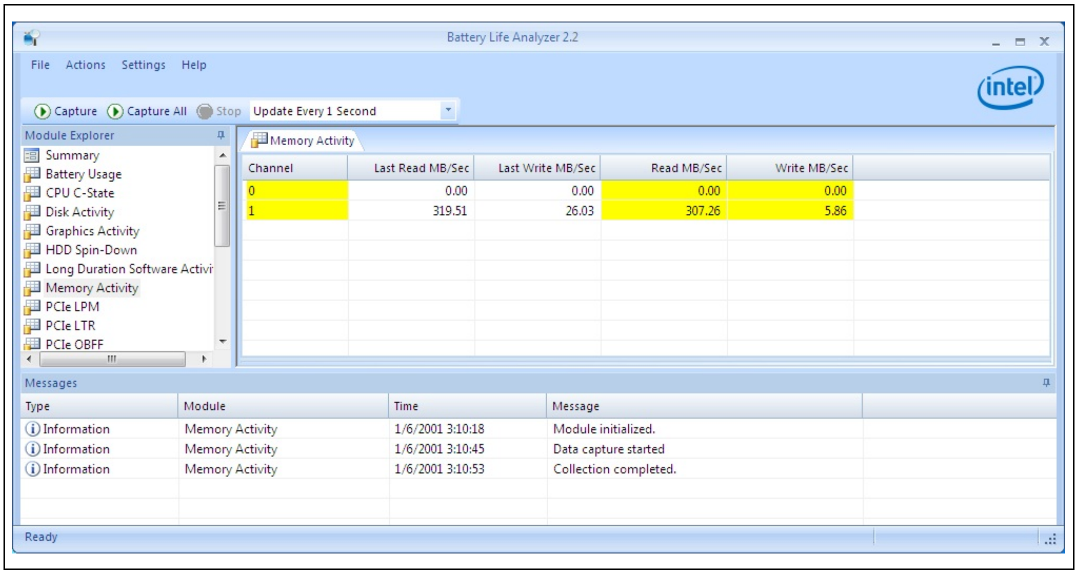
Last Read MB/Sec: Memory read per second within the last measurement window

Last Write MB/Sec: Memory write per second within the last measurement window

Read MB/Sec: Running average of read per second since the start of measurement

Write MB/Sec: Running average of write per second since the start of measurement

Figure 24. Memory Activity Module

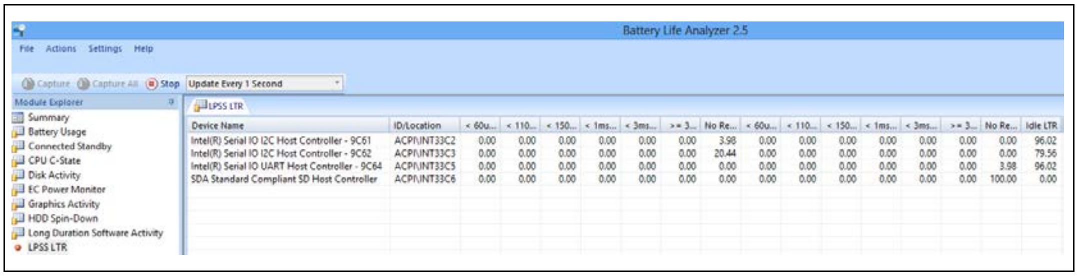


4.5.15 LPSS LTR Analysis

Similar to PCIe LTR module, as of BLA version 2.5 LPSS (Low Power Subsystem) LTR analysis is provided. BLA will load LPSS LTR module only when the LPSS controllers are enabled.

LPSS LTR module displays a histogram of the LTR values. Note that, there are two types of LTR mechanisms employed , namely Hardware(HW, also called Auto) and Software (SW) mechanisms. For example in Sharkbay ULT platform I2C controller implements HW LTR mechanism whereas SDA Standard Compliant SD Host Controller works in SW LTR mode. Depending on the LTR mechanism used by the controller, associated LTR values are displayed in the SW or HW(Auto) LTR buckets of the histogram. BLA also displays the % time no LTR requirement received in "No Requirement" column and % time BLA detected that the device was in D3 in "Idle LTR" column. Note that, in order to get the most accurate D-State residency metrics for LPSS devices, users should use BLA Device Power Analysis module.

Figure 25. LPSS LTR Analysis



4.6 Software Analysis

When idle, power efficient applications should have minimal impact on platform power consumption. Misbehaving software or services that issue frequent interrupts or periodic disk accesses can have a negative impact on the C-state residency and HDD spin-down. The first level of analysis as described in previous sections will be able to determine the platform power impact from hardware behavior changes (i.e., CPU utilization, HDD spin down duration, and various link state residency information). Software Activity analysis serves as a second level of analysis and looks at all the software components running on the system and highlights the ones that are causing the system to come out of deep C-states or that cause a hard drive to spin-up. The user can monitor the impact of specific software by temporarily disabling or removing it and checking the improvement in C-state residency and HDD spin down percentage.

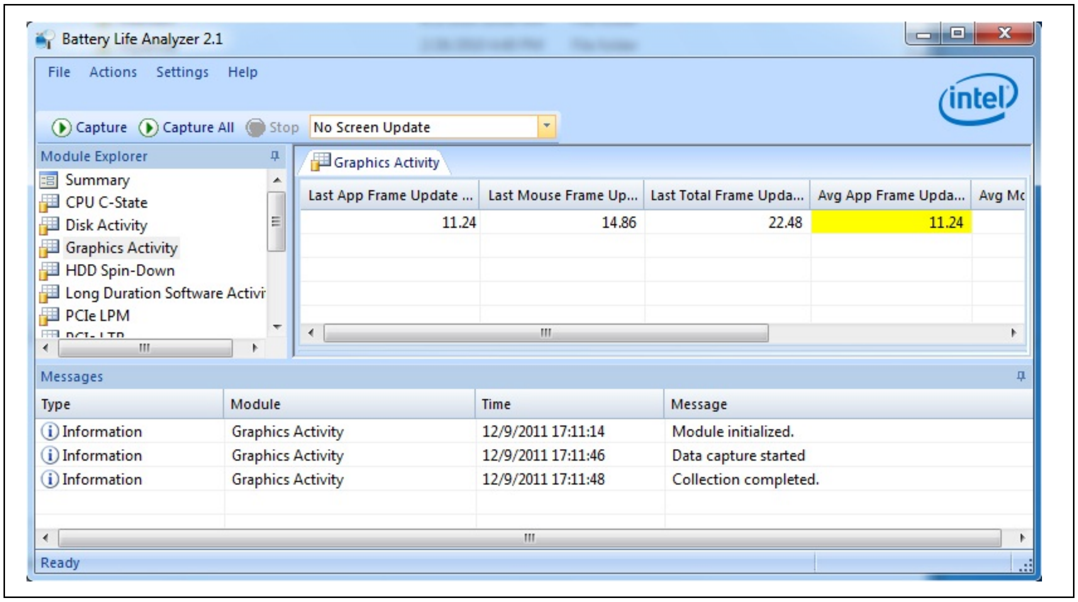
The Software Activity Analysis Module can save the traces as a single ETL file which can be post-processed using XPERF or similar tools. This feature also allows ETL files to be saved and opened in BLA.

4.6.1 Graphics Activity

This module can be used to identify any software that is frequently updating graphics when no user activity is taking place. Some percentage of graphics activity is expected when no obvious graphics update is happening; however it should be minimal (close to zero). The module will display (Figure 26):

- Average number of frames updated per second (over the duration of analysis) due to application activity.
- Number of frames updated per second due to application activity since last update. NOTE: This field only applies if real-time data updates are selected. Otherwise, this field will be equal to the average number of frames per second.
- Average number of frames updated per second (over the duration of analysis) due to mouse pointer activity.
- Number of frames updated per second due to mouse pointer activity since last update. NOTE: This field only applies if real-time data updates are selected. Otherwise, this field will be equal to the average number of frames per second.
- Average total number of frames updated per second (over the duration of analysis).
- Number of total frames updated per second since last update. NOTE: This field only applies if real-time data updates are selected. Otherwise, this field will be equal to the average number of frames per second.

Figure 26. Graphics Activity Module



4.6.2 Disk Activity

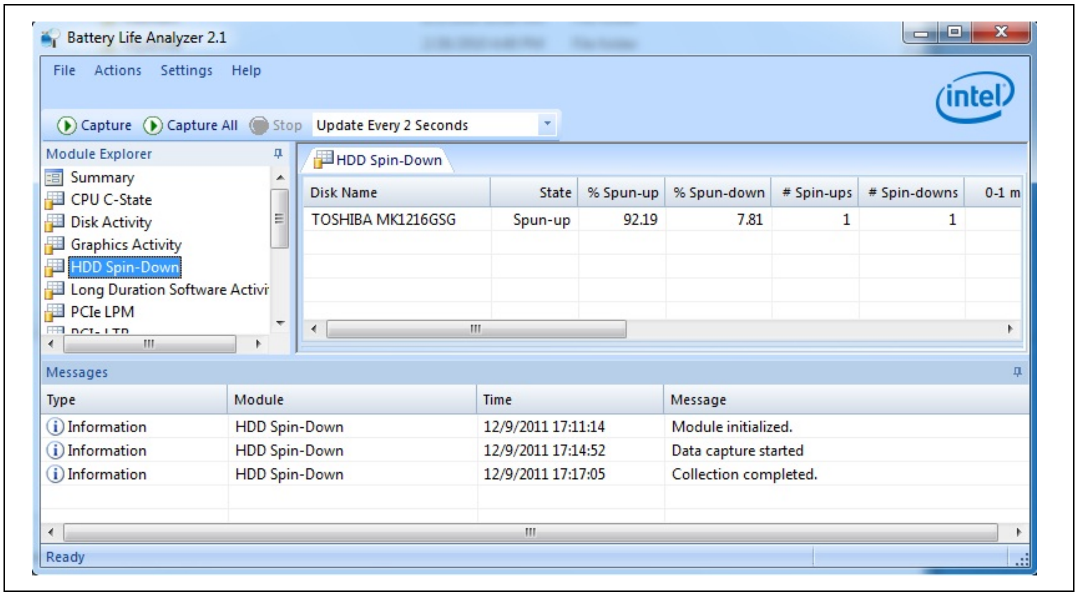
If the HDD Spin-down analysis is showing that the HDD is spending a lot of time spun-up, then the Disk Activity modules' analysis can be used to identify the software that is causing the disk spin-up. It is recommended to run this analysis with a short spin down timer setting (e.g., 1 minute) to catch the activities that prevent disks from spinning down. For example, in Microsoft Windows Vista, the spin down timer duration can be changed by going to Control Panel → Power Options → Change Plan Settings → Change Advanced Power Settings → Hard Disk → Turn Off Hard Disk After → Setting.

Note: On systems equipped with only an SSD, the Disk Activity Module analysis is not supported.

The module will display (Figure 27):

- Time of the disk spin-up event
- Process name that triggered disk spin-up
- Process ID of the process that caused disk spin-up
- Thread ID of the thread responsible for the disk spin-up
- Target file of the I/O that caused the disk spin-up
- I/O access type that caused disk spin-up
- Stack trace for the I/O that caused disk spin-up

Figure 27. Disk Activity Module



4.6.3 Active Analysis

Active Software analysis focuses on the CPU intensive usage and how the activity of each core is overlapping. Upon completion of this analysis, the following information will be displayed (Figure 28):

- Process Name - in general all the names will match Processes listed in the Task Manager provided the “Show Processes from All Users” feature is enabled. Threads of “System” processes will be shown separately in BLA.
- Percentage of platform usage – percentage of the duration when the entire platform is active
- Percentage of logical CPU usage – percentage of total CPU usage (all cores or threads depending on the CPU type)
- Number of context switches from Idle over the length of analysis to indicate C-state transitions cause by each software component.
- Timer Tick Period – Displays the average Timer Tick Period for the duration of the analysis for each process that reprogrammed the timer tick. For Platform Activity, it displays the average Timer Tick across all the processes in the system. If the process didn’t request timer tick period, this cell is left blank.
- Power Impact estimate where applicable

As general guidance, percent of CPU usage should be very low during idle. Also look for processes causing higher number of context switches from idle (C-switches from Idle, which is a good proxy for the C-state transition caused by each software) if the residency in the deepest C-state was low.

Note: To minimize the impact of BLA itself, it is recommended to run this analysis with "No Screen Update" and move the cursor out of the BLA window during data collection. This change will impact the ability to monitor real time data.

Note: "Platform Activity" process that always appears at the top of the list is a "pseudo process" and represents an "OR" of all software activities (see Figure 28).

Figure 28. Active Analysis

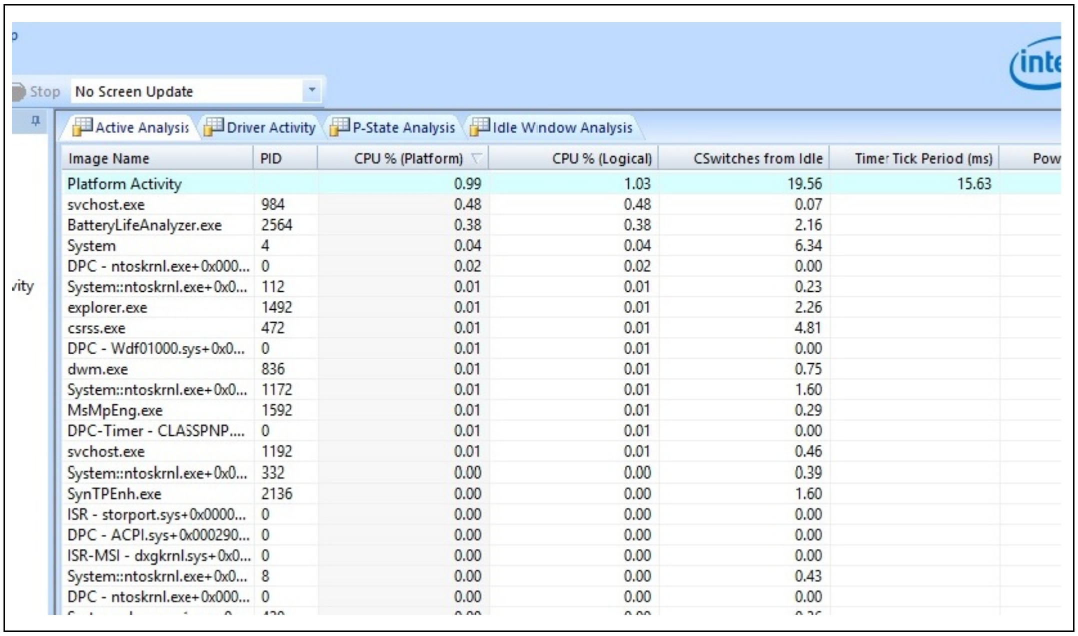


Image Name	PID	CPU % (Platform)	CPU % (Logical)	CSwitches from Idle	Timer Tick Period (ms)	Pow
Platform Activity		0.99	1.03	19.56	15.63	
svchost.exe	984	0.48	0.48	0.07		
BatteryLifeAnalyzer.exe	2564	0.38	0.38	2.16		
System	4	0.04	0.04	6.34		
DPC - ntoskrnl.exe+0x000...	0	0.02	0.02	0.00		
System::ntoskrnl.exe+0x0...	112	0.01	0.01	0.23		
explorer.exe	1492	0.01	0.01	2.26		
csrss.exe	472	0.01	0.01	4.81		
DPC - Wdf01000.sys+0x0...	0	0.01	0.01	0.00		
dwm.exe	836	0.01	0.01	0.75		
System::ntoskrnl.exe+0x0...	1172	0.01	0.01	1.60		
MsMpEng.exe	1592	0.01	0.01	0.29		
DPC-Timer - CLASSPNP....	0	0.01	0.01	0.00		
svchost.exe	1192	0.01	0.01	0.46		
System::ntoskrnl.exe+0x0...	332	0.00	0.00	0.39		
SynTPEnh.exe	2136	0.00	0.00	1.60		
ISR - storport.sys+0x0000...	0	0.00	0.00	0.00		
DPC - ACPI.sys+0x000290...	0	0.00	0.00	0.00		
ISR-MSI - dxgkrnl.sys+0x0...	0	0.00	0.00	0.00		
System::ntoskrnl.exe+0x0...	8	0.00	0.00	0.43		
DPC - ntoskrnl.exe+0x000...	0	0.00	0.00	0.00		

4.6.4 Driver Activity

Driver Activity Analysis focuses on the activity generated by the drivers. Types of events that are being analyzed are:

- Interrupt Service Routine (ISR) – is a software routine that hardware invokes in response to an interrupt. An ISR can happen when the CPU is idle and therefore can wake the whole platform.
- Deferred Procedure Call (DPC) – is a routine that allows high-priority calls to defer required, but lower priority, tasks for later execution. DPC routines are usually initiated by an ISR, and executed on the same logical CPU as ISR routines. DPCs only happen while the CPU package is already in C0. Although they cause the C0 duration to increase, DPCs do not have as significant a battery life impact as ISR type of routines do.

Upon completion of this analysis, the following information will be displayed (Figure 29):

- Name of the driver, offset of the routine
- Type of activity: ISR, DPC or DPC timer

- Number of calls generated by this driver per second
- Number of Unhandled Calls per second – Some devices share interrupt lines where the OS does not know which device generates the interrupt. This could result in unhandled interrupt calls. If a device generates a lot of interrupts, it's better to design it in such a way that it has its own interrupt line.
- CPU utilization percentage for the particular routine

Figure 29. Driver Activity

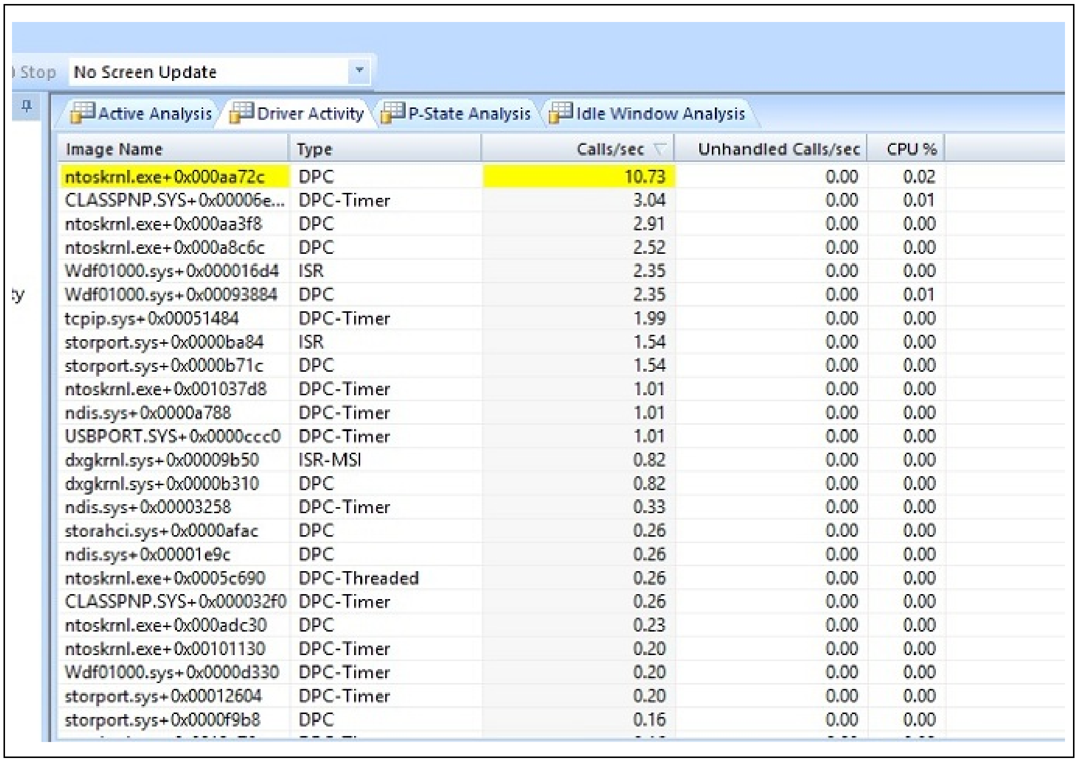


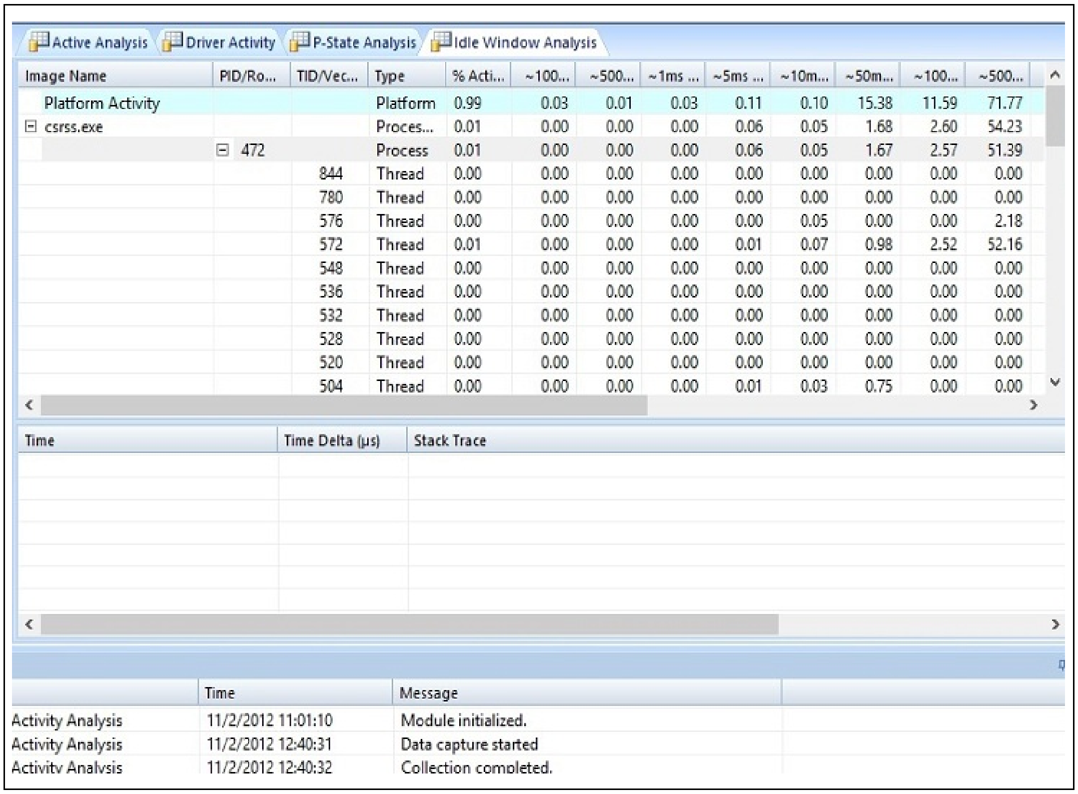
Image Name	Type	Calls/sec	Unhandled Calls/sec	CPU %
ntoskrnl.exe+0x000aa72c	DPC	10.73	0.00	0.02
CLASSPNP.SYS+0x00006e...	DPC-Timer	3.04	0.00	0.01
ntoskrnl.exe+0x000aa3f8	DPC	2.91	0.00	0.00
ntoskrnl.exe+0x000a8c6c	DPC	2.52	0.00	0.00
Wdf01000.sys+0x000016d4	ISR	2.35	0.00	0.00
Wdf01000.sys+0x00093884	DPC	2.35	0.00	0.01
tcpip.sys+0x00051484	DPC-Timer	1.99	0.00	0.00
storport.sys+0x0000ba84	ISR	1.54	0.00	0.00
storport.sys+0x0000b71c	DPC	1.54	0.00	0.00
ntoskrnl.exe+0x001037d8	DPC-Timer	1.01	0.00	0.00
ndis.sys+0x0000a788	DPC-Timer	1.01	0.00	0.00
USBPORT.SYS+0x0000ccc0	DPC-Timer	1.01	0.00	0.00
dxgkrnl.sys+0x00009b50	ISR-MSI	0.82	0.00	0.00
dxgkrnl.sys+0x0000b310	DPC	0.82	0.00	0.00
ndis.sys+0x00003258	DPC-Timer	0.33	0.00	0.00
storahci.sys+0x0000afac	DPC	0.26	0.00	0.00
ndis.sys+0x00001e9c	DPC	0.26	0.00	0.00
ntoskrnl.exe+0x0005c690	DPC-Threaded	0.26	0.00	0.00
CLASSPNP.SYS+0x000032f0	DPC-Timer	0.26	0.00	0.00
ntoskrnl.exe+0x000adc30	DPC	0.23	0.00	0.00
ntoskrnl.exe+0x00101130	DPC-Timer	0.20	0.00	0.00
Wdf01000.sys+0x0000d330	DPC-Timer	0.20	0.00	0.00
storport.sys+0x00012604	DPC-Timer	0.20	0.00	0.00
storport.sys+0x0000f9b8	DPC	0.16	0.00	0.00

4.6.5 Idle Window Analysis

Idle window analysis focuses on the frequency of software activity on the platform. It reports on the idle duration in-between activity of various processes running on the platform. The following information is reported: (Figure 30):

- Process Name
- Active duration percentage for each process and thread
- Distribution of the idle duration showing the percentage of time the process was idle in each time duration bucket.
- Power impact estimate where applicable

Figure 30. Idle Window Analysis

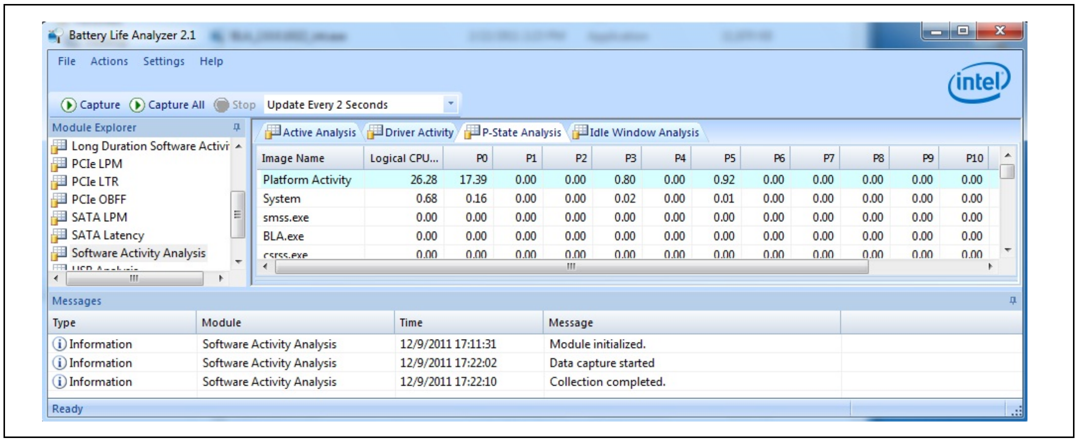


4.6.6 P-State Analysis

P-State Activity Analysis provides information about the various ACPI Performance or P-states that the OS logical processors go into while executing various processes in the system. Depending upon the number of P-states exposed by the BIOS and enumerated by the OS via ACPI, BLA will list the P-states in the display along with the % of CPU time taken in each P-state by each of the process. Upon completion of the P-state Analysis, the following information will be displayed (Figure 31):

- Process Name - Process whose activity is being measured. The Platform Activity process is a pseudo process as described earlier and is the "OR" of all the processes in the system.
- Logical CPU Percentage – This is the sum of all the percentages spent in various P-states by the Process.
- P0 (highest CPU frequency) through Pn (lowest CPU frequency) – The value "n" varies with the number of P-states exposed by the BIOS and reported by the OS. In the example in Figure 31, there are 9 P-states. Actual frequency (in MHz) of each P-state is displayed by hovering mouse cursor over the table header.

Figure 31. P-State Analysis

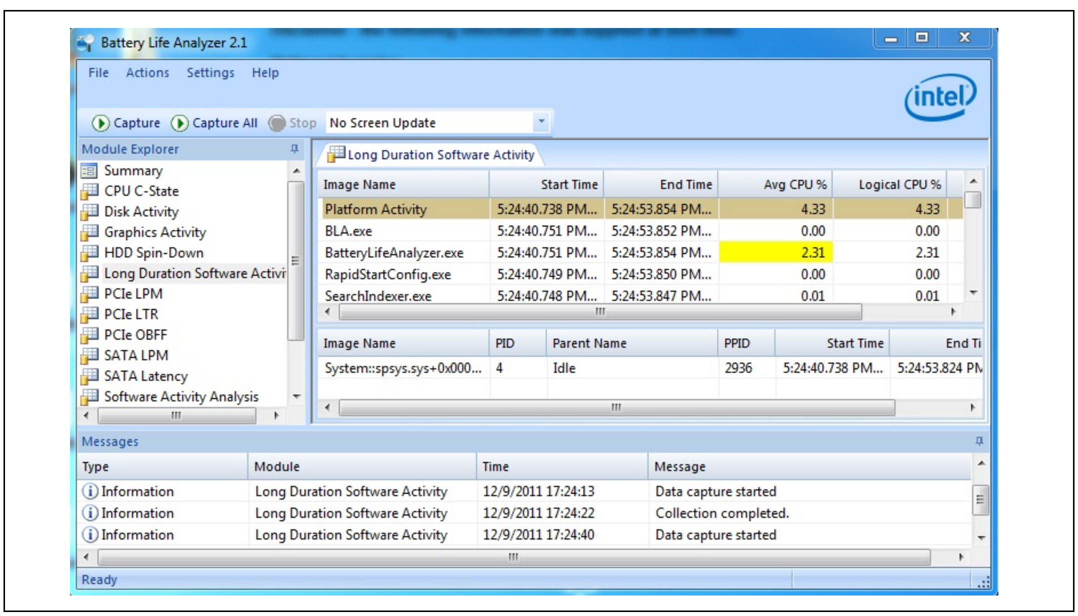


4.6.7 Long Duration Software Activity

Long Duration Software Activity analysis is run over a longer period of time and focuses on the processes that run infrequently but consume a lot of power during those infrequent times of activity. Recommended collection time is greater than 15 minutes, but varies depending on the application of interest. Figure 32 shows the display

Note: To minimize the impact of BLA itself, it is recommended the user move the cursor out of the BLA window during data collection.

Figure 32. Long Duration Software Activity



Note: Long Duration Software Activity module can only run in Manual Mode.

4.7 Symbol Support

Routine addresses (ISR/DPC routine addresses, system threads' start address, etc.) in Software Activity Analysis and Disk Activity Analysis are shown in the form of *<module-name> + offset* by default. If you wish to use the symbol name, use the "Set Symbol Path" menu item in the "Settings" menu. When _NT_SYMBOL_PATH environment variable is specified, BLA tries to get symbol information from the specified path. Otherwise, BLA will try to retrieve symbol information from Microsoft's public symbol server via the network.

Please refer to Microsoft Knowledge Base (http://support.microsoft.com/kb/311503) for detailed information about using Microsoft Symbol Server to obtain symbol files.

Note: It can take up to several minutes to translate module addresses to matching symbols.

§

5 Recommended Platform Analysis

5.1 Best Known Methods for Using the Battery Life Analyzer

Intel recommends the following analysis method with Battery Life Analyzer tool.

Step 1: Disable auto-sleep when running long duration analysis.

Step 2: Let the system remain Idle for more than 15 minutes after starting BLA tool but before pressing "Capture" button.

Step 3: Run analysis in either Manual Mode, selecting each module by hand, or in Automatic Mode (will take 5+ minutes).

Step 4: Examine the Analysis Summary page to review all highlighted warnings.

Step 5: For problems identified by software analysis, locate the program (or driver) by double clicking on the name.

Step 6: For problems identified by hardware analysis, try temporarily disabling the device in the Windows Device Manager and examine the impact.

5.2 Interpreting BLA results – Idle Analysis

Table 4 below provides guidelines on interpreting and analyzing the results from BLA. Upon completion of analysis, the user can either examine individual modules, or the Analysis Summary screen. Highlighted cells within each module can be used as indicators of potential issues and it is suggested that the user checks those first.

Table 4. Idle Analysis Interpretation

Module	Issue	Next Steps
CPU C-State	High Core C0 residency (Recommended C0 residency is less than 5%)	• CPU is executing instructions. • Identify component in "Active Analysis" or "Driver Activity" with high CPU utilization.
	High Package C2 residency	It is likely that bus master device is making constant memory access 1) Bus master device is making constant memory access and in this case, CPU needs to wake-up to C2 state to respond to memory access. 2) Identify bus master devices making frequent snoop activity. Disabling bus master device one by one can help identify offending device.
	Residency in the deepest C-state should be >95%	If residency in C0 is <5%, but deepest C-state residency is significantly lower than 95%: 1) CPU "auto demote" to lower C-state may be taking place. 2) Check "Cswitches from Idle" column of Active Analysis and identify process that causes high transition. 3) Check for ISR type of processes with high "Calls/sec" of "Driver Activity". 4) Check for processes with high "% Active" values and short durations in the "Idle Window Analysis". 5) If possible, disable in Device Manager to see if deeper C-state residency increases.
HDD Spin-down	Disk stays spun-up most of the time	1) Check the Disk Activity Module to identify software preventing spin down.
	Frequent spin-up/down (High number in 0~1 and 1~2 minutes bucket)	1) Consider extending HDD spin-down timer in OS policy to reduce frequent spin-ups, which negatively affects user experience and battery life.
PCIe LPM	L0s and/or L1 residency is 0	1) Check reported capabilities of the PCIe port "EP L0s Capable", "EP L1 Capable", "RP L0s Capable", and "RP L1 Capable". 2) Check that L0s and L1 are enables on both EP and RP where available.
	Residency in L1 is less than 98% for a link that is active	1) Identify link with low L1 residency.
	Residency in Down is not 100% for the unused link	1) Identify unused link that is not 100% down. 2) Check BIOS settings for correct power management implementation.
SATA LPM	Residency in Slumber for active links is less than 90%	1) Identify link with low Slumber Residency. 2) Check both host controller and device's capability & enable status.

Module	Issue	Next Steps
XHCI LPM	U1 and/or U2 residency is zero	Check the U1 and/or U2 Enable and Timeout settings
EHCI Analysis	Selective Suspend duration is not 100% for particular host controller	3) Look at selective suspend duration of each host controller. 4) Identify host controller with less than 100% selective suspend. 5) Check selective suspend duration of each device and/or hub under it. 6) Identify which device is preventing host controller from staying in selective suspend state. 7) For device identified above, look at the active duration of periodic and asynchronous schedules to get an idea of what kind of activity that device is making.
Long Duration Software Activity	Sporadic activities consume large amounts of power	1) Identify and understand the impact of sporadic activities.
Active Analysis	Process activity is high while no user activity is happening	1) Investigate process activity in the order of CPU utilization. 2) Check any activity that utilizes CPU for more than 0.5%.
Driver Activity	Interrupt activity exceeds the guideline defined in “Next Steps”	Check Frequency of interrupt (“Calls/sec” – “Unhandled Calls/sec”) and see if there are any ISR type of activity happening significantly more frequently than outlined in Table 5.
	CPU Utilization of each ISR/DPC activity is not minimal	1) Check any driver with greater than 0.1% of logical CPU utilization.
Idle Window Analysis	Percentage of time processes are active and relative duration of idle periods	1) Investigate process activities which high “% Active” values. 2) Investigate process which has high percentages of low duration idle periods.
Graphics Activity	Percentage of frame updated/second when no obvious graphics activity is happening is not close to zero	1) Stop applications as much as possible. In particular, applications that with icons in the task bar.
Timer Tick Analysis	Platform Activity Timer Tick Period is less than 15.6 ms	1) Check for processes other than Platform Activity that have a value in the Timer Tick Column. 2) If the process is svchost.exe, then most likely it is acceptable since a core OS component modified the Timer Tick. 3) If the process is not svchost.exe, the change made in the Timer Tick Value may be unnecessary and requires further analysis.

Table 5. ISR Activity Guidelines

Driver	Interrupt Source	Typical interrupt/sec @ Idle
ACPI.sys	Motherboard	< 1
dxgkrnl.sys	Graphics	<5 (depends on activity)
hal.dll	Timer Tick	64
i8042prt.sys	PS/2 KB/Mouse	0
iaStor.sys msahci.sys	SATA (HDD/DVD)	<10 (depends on activity)
Ndis.sys	Network (GbE/WiFi)	<30 (WiFi, associated, no traffic)
USBPORT.sys	USB (UHCI/EHCI)	0

Note: These values are not recommendations from Intel, but instead are used to indicate potential problems. They may not be actual problems.

§