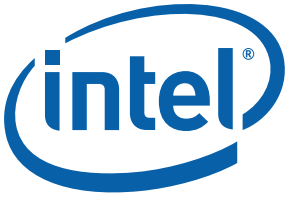

Intel® Energy Checker

SDK Companion Applications User Guide

Revision 2.0

December 15, 2010

Revision History

Revision	Revision History	Date
1.0	Initial Release	2009/10/12
2.0	2010.12.15 Release	2010/12/15

Contents

Figures

Tables

1 Companion Applications Overview

This *Intel® Energy Checker Companion Applications User Guide* is part of the Intel® Energy Checker (Intel® EC) Software Development Kit (SDK). The Intel EC SDK enables developers of Independent Software Vendors (ISVs) to easily import and export counters in their source code. Although the initial intent of the Intel EC SDK is to facilitate software energy efficiency analysis and optimizations, it can be used to expose any counter meaningful to each ISV and its customers.

The Intel® Energy Checker is also referred to as the Intel® EC. The Intel EC SDK is also referred to as the SDK.

Detailed descriptions of the SDK and use models can be found in the *Intel® Energy Checker Software Developer Kit User Guide*.

NOTE

This user guide assumes the reader is familiar with the 64-bit counters and Productivity Links (PLs) described in the Intel® Energy Checker SDK User Guide. Please refer to that document for a description of counters and PLs.

This user guide describes two companion applications provided with the Intel® Energy Checker SDK:

- **PL GUI Monitor** provides a graphical tool to dynamically examine Productivity Link (PL) counters created by applications instrumented with the Intel EC SDK.
- **PL CSV Logger** is a console application that can be used to dynamically dump PL counters to the console or to an output file that can be imported into most spreadsheet applications for further analysis.

This page intentionally blank.

2 PL2 GUI Monitor

The PL GUI Monitor (or monitor) allows users to graphically monitor one or more Productivity Links (PLs) in real-time. PL monitor runs under Microsoft* Windows*, Linux*, Solaris* 10, or MacOS* X systems with X11 (X Window) support. PL Monitor can be used for building IT dashboards, studying applications' performance, monitoring systems' behaviors, and debugging applications.

2.1 Introduction

The PL GUI Monitor's interface was inspired by the Engine Indicating and Crew Alerting System (EICAS) and Electronic Centralized Aircraft Monitor (ECAM) in modern aircraft. This choice is based on functional similarities (system monitoring), and on the strong conviction that these graphical interfaces were carefully designed to convey data rapidly to the users and to limit interpretation errors.

A monitor creates a window that shows one to several graphical meters and counter data. Figure 1 shows a typical configuration of a single monitor showing four gauges. Figure 2 shows a monitor with eight gauges.

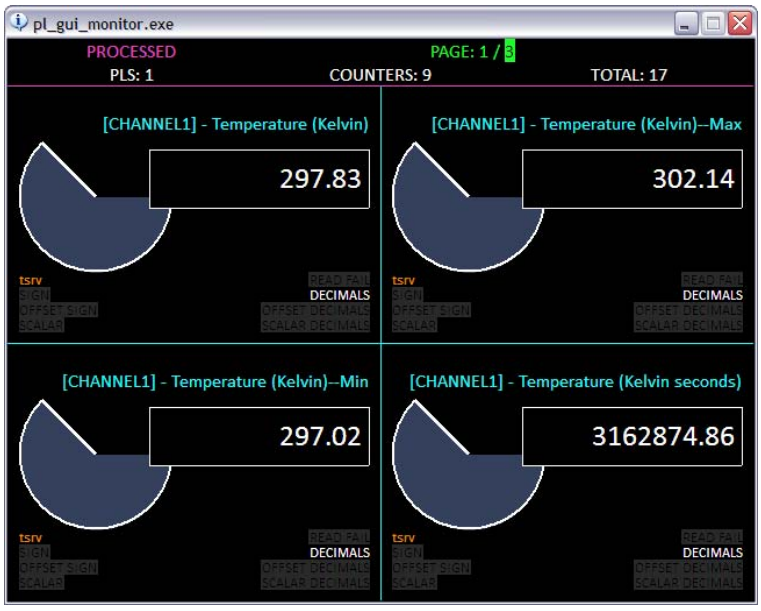

Figure 1: Typical monitor view with simulated data

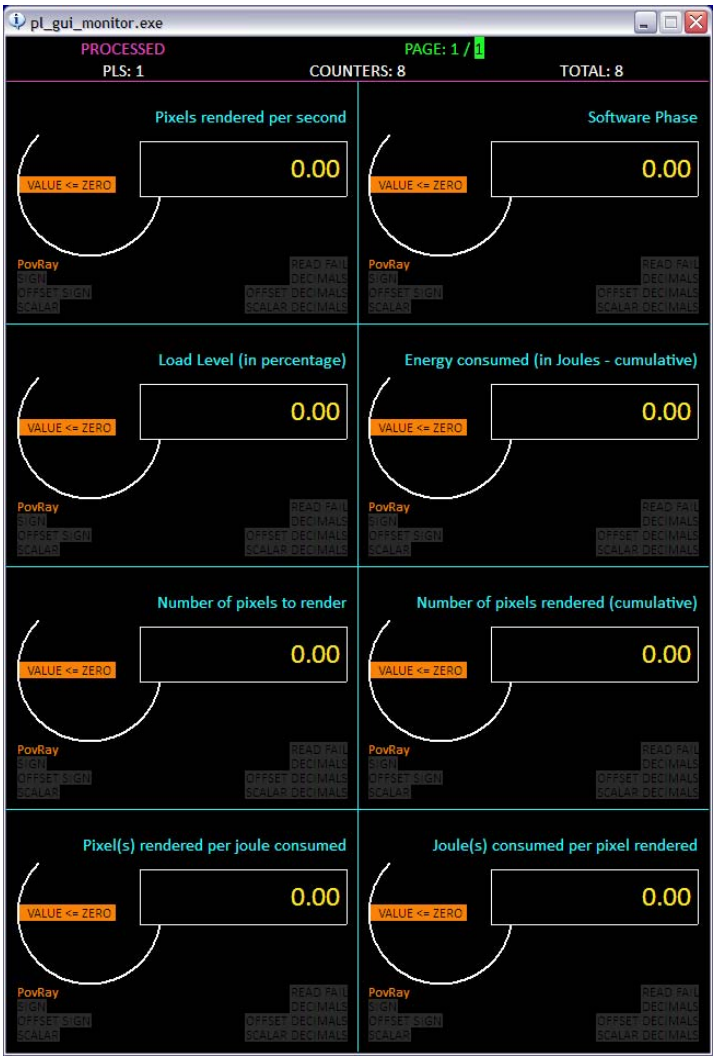

Figure 2. Monitor window with eight counters

2.2 Monitor Display Overview

Figure 3 shows a monitor's typical display. The display area is divided into two main zones:

- Header zone on top
- Gauge zone below

A magenta horizontal line separates the two zones.

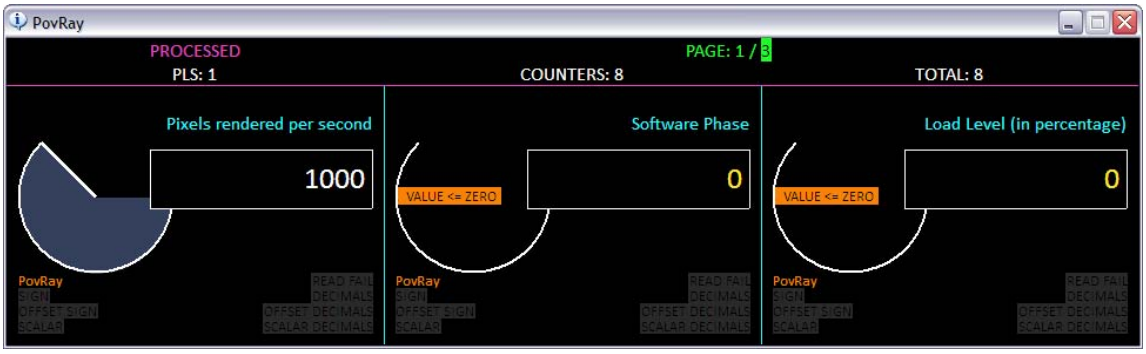

Figure 3: Monitor display with 1 row by 3 columns of gauges

The header zone's first line displays the following elements:

- The magenta `PROCESSED` indicator is displayed if the `--process` option was included on the command line when the monitor was started. If present, this indicates that displayed values have been *adjusted* (or processed) as defined by the counter. If absent, it indicates raw counter information is displayed.
- The active page indicator is shown in green.
- If there are more counters in a PL than can be shown in a given monitor, these counters are shown on successive pages. The total page count is shown in reverse green. The user can cycle through these pages using the commands described in Section 2.3 of this manual.

The header zone's second line displays the following elements in white:

- The number of PL's monitored.
- The number of active counters monitored.
- The total number of counters exposed by the monitored PL(s). This includes any suffix (attribute) counters that are suppressed when the processed values of counters are displayed.

Figure 4 below shows how eight counters can create a single processed counter in a PL.

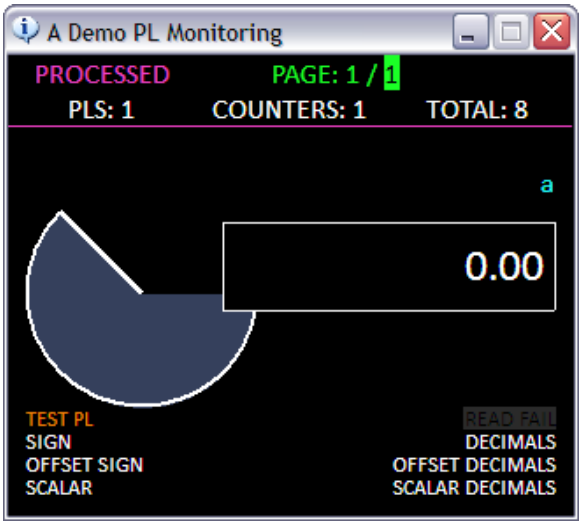

Figure 4: Gauge built-out of eight (8) PL counters

The gauge zone uses the rest of the window area below the header. If multiple gauges are displayed on a page, they are separated by a cyan grid as shown in Figure 3.

The upper portion of the gauge zone displays the following elements:

- In cyan, the counter name, (the non-descriptive "a" in the figure above).
- In white in a white outlined box, the counter value.
- In gray and white, a graphical dial with a white needle and a grey sweep angle (from 3:00 to approximately 11:00). This represents the counter's current value relative to its maximum value. Since most PL counters represent monotonically increasing values, the latest value will usually be the maximum value to date for that counter

The lower portion of each gauge displays the following elements:

- In amber, the PL name to which the counter belongs (TEST PL, in the case).
- In white on red background (if active), a `READ FAIL` indicator indicates a `pl_read` failure. Note that a `READ FAIL` condition is graphically re-enforced with other changes in the gauge's graphic (see Figure 7).
- Several suffix indicators are shown in white text, if that suffix indicator was detected during the PL analysis phase. Black text on a gray background indicates the counter was not detected during the PL analysis phase:
    - `SIGN` indicates if a `.sign` suffix counter was detected
    - `DECIMALS` indicates if a `.decimals` suffix counter was detected
    - `OFFSET SIGN` indicates if an `.offset.sign` suffix counter was detected
    - `OFFSET DECIMALS` indicates if an `.offset.decimals` suffix counter was detected
    - `SCALAR` indicates if a `.scalar` suffix counter was detected

- `SCALAR DECIMALS` indicates if a `.scalar.decimals` suffix counter was detected

Figure 5 below shows the graphical dial for a counter as it recedes from its maximum value, from 4 to 1 (read left to right, top to bottom).

Figure 5: Example dials for a declining counter

NOTE

If the counter in the example above had initially increased from 1 to 4, the dial and sweep angle portion of the gauge would appear like the upper left example since the counter's current value would match its maximum value. The maximum value is determined on the fly while the monitor is running.

NOTE

The sweep angle and needle are shown only if the counter's value is positive. If the value is negative or null, then the dial displays an amber `VALUE <= ZERO` *indicator (Figure 6).*

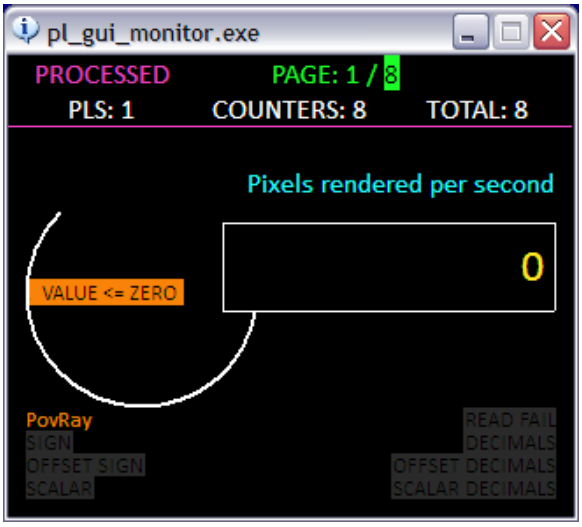

Figure 6: Negative or zero value display

If a read failure occurs, the error is textually displayed and is reinforced with visual cues as shown in Figure 7.

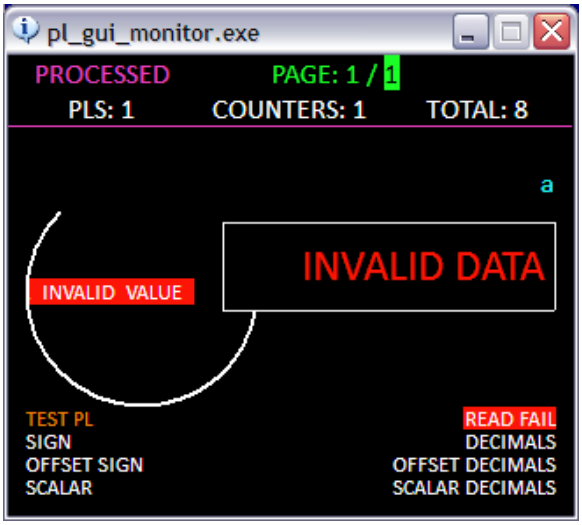

Figure 7: Gauge with a read error

NOTE

Counters should be uniquely named within a PL. A PL with two or more counters with identical names will not be interpreted properly by the monitor application.

At startup and during monitoring, each monitor instance prints to the standard output various data used for data capturing, debugging, and informational purposes. During normal operations, this data can be ignored and the console window (shown in Figure 8) can be minimized.

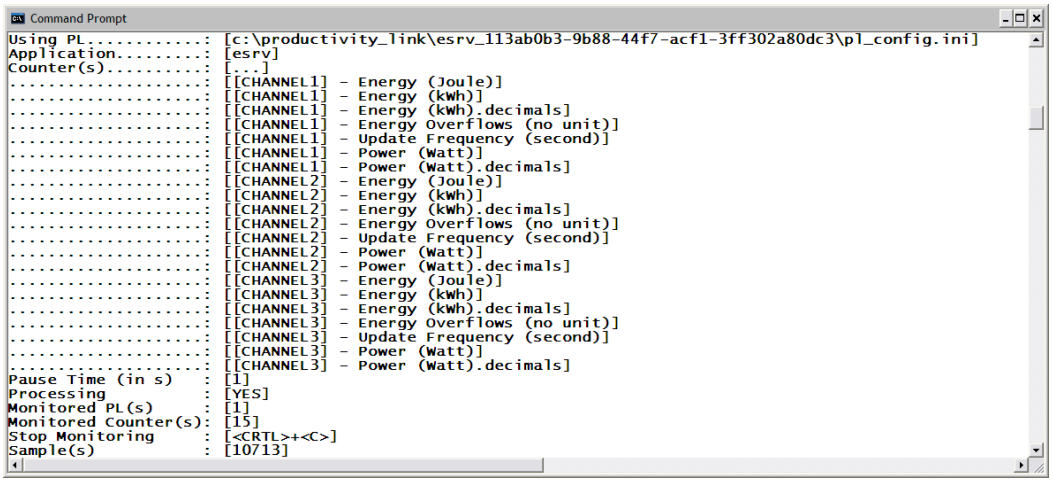

Figure 8: Monitor console window used for debugging

2.3 Interactive Commands

After the monitor is started, you can affect monitor operation using both keyboard command and mouse controls as described below.

2.3.1 Keyboard Commands

Table 1 summarizes the keyboard commands available to interact with the monitor.

Table 1. Keyboard Commands

Key Stroke	Action
<Escape>, <CTRL> + C	Quit the monitor
<ALT>+<F4>	Quit the monitor (Windows-only)
A	Display maximum values
I	Display minimum values
V	Display average (mean) values
M	Set the maximum indicator to the maximum allowed value
R	Reset all the counters, min, max, averages and samples count to the current counter value; does not affect the counters themselves, just the monitor
F	Freeze the display; press F again to unfreeze the display
C	Capture the counters' values and print them to the standard output
<HOME>	Set active page to first page
<END>	Set active page to last page
-, <LEFT ARROW>, <UP ARROW>, <PAGE UP>	Decrement active page; set to first page if on the last page
+, <RIGHT ARROW>, <DOWN ARROW>, <PAGE DOWN>	Increment active page; set to last page if on the first page

The monitor tracks the minimum, maximum, and mean average values while it is active. These values can be displayed using the keyboard commands. Figure 9 below illustrates what the gauges look like when these special modes are displayed. Note the indicator identifying each type of value.

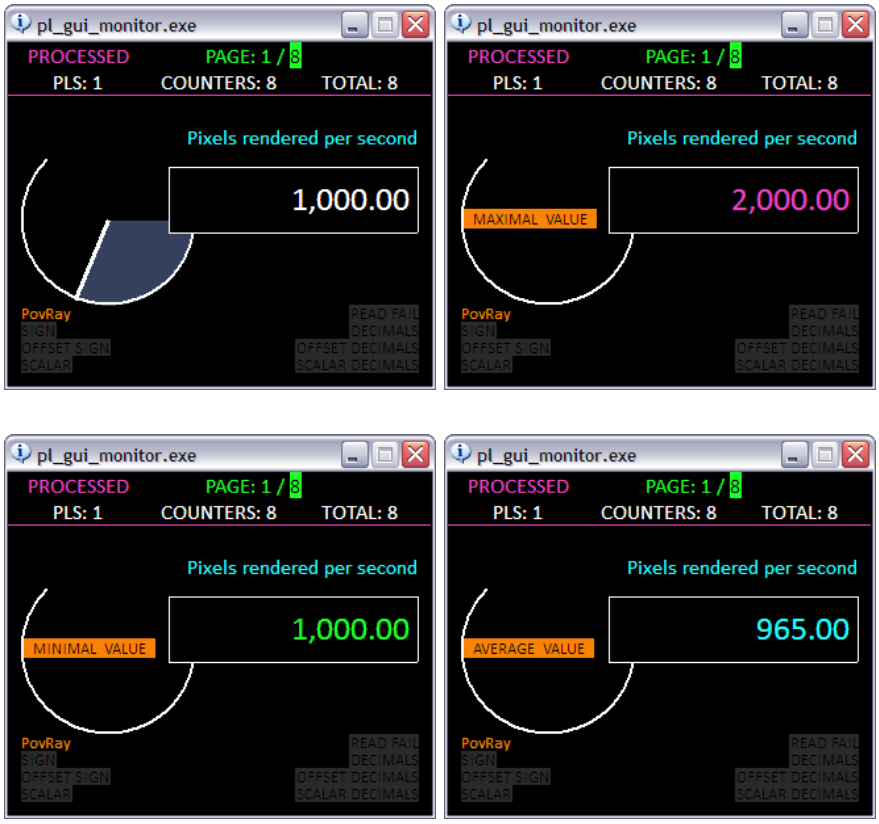

Figure 9: Examples of normal, min, max and average display mode

NOTE

If the minimum, maximum, or average display mode is active, then pressing 'N' (or any non-recognized key) switches the display back to normal mode.

2.3.2 Mouse Commands

In addition to the normal window manipulation capabilities (move, minimize and close windows), the user can switch between transparent and opaque display modes by double-clicking in the monitor's display window. This capability is only available if transparency was requested when the monitor was started (see section 2.6.4.4).

2.4 Starting the Monitor

Every Productivity Link (PL) created by the Intel Energy Checker SDK creates a configuration file (typically named *pl_config.ini*) in a special directory dedicated to

that PL. The monitor reads the information in the configuration file to present the PL counter information to the user.

In its simplest form, the monitor can be invoked with a single argument: the qualified path to a PL's configuration file (`pl_config.ini`). For example:

```
pl_gui_monitor c:\productivity_link\PovRay_76b5e071-34be-47fc-8f26-
   931157d58f5e\pl_config.ini.
```

If no PL configuration file is provided on the command line, the monitor opens a dialog box to select PL configuration file(s) manually on a Windows system. The PL configuration file must be specified on the command line for non-Windows systems.

NOTE

Use double quotes (") around the `pl_config.ini` *path if it contains spaces.*

NOTE

For details on PLs, PL counters and PL configuration files, please refer to the Intel® Energy Checker SDK User Guide.

The monitor is designed for IT professionals and power users. Due to the large number of options that can be provided on the command line, it is anticipated that the monitor will usually be invoked from a script/batch file or from a higher-level application.

It is generally preferable to have separate instances of the monitor application running to monitor separate sets of data. One monitor instance can monitor up to ten (10) PLs, as is shown in the following example (with only 2 PLs).

The following command line shows how to start two monitors by specifying the pl_config.ini file and its path. To open up to ten PLs, include all filenames/paths in the command line.

```
pl_gui_monitor c:\productivity_link\PovRay_76b5e071-34be-47fc-8f26-
   931157d58f5e\pl_config.ini c:\productivity_link\esrv_f6bde0a1-3ebe-55fc-
   8f26-9f1cc7de8f5e\pl_config.ini.
```

NOTE

If the same PL configuration file is specified multiple times, the PL only uses it once.

At startup, the monitor(s) attach to the specified PL(s), and start displaying the PL counters and their values with a 1 Hz refresh frequency.

2.5 Stopping the Monitor

The monitor may be terminated with one of the following keys or key combinations when the console monitor screen is active and the focus: Escape, <CTRL>+<C>, or <ALT>+<F4> (Windows only). The monitor window may also be terminated by closing the window via a mouse.

NOTE

Be careful when using <CTRL>+<C>, as other applications may be interrupted too. The other methods for terminating the monitor application are generally preferred.

2.6 Startup Command Line Options

The options described in this section affect how the monitor displays the various counter information. Use these options when you invoke a monitor instance.

2.6.1 Counter Processing

`--process` display processed counter values instead of raw values

By default, the monitor displays the PL counters in raw form. However, many counters have some suffix counters to help interpret the counter values. For example, a `.decimals` counter of 3 indicates that the raw value is one thousand (10^3) times the actual value.

Add the `--process` option to the command line to analyze the counters exported in the PL(s) dependencies amongst them. Dependency detection is confined to each PL. If such dependencies are found, the monitor can use the suffix counters to compute the base counters' real value. Refer to the *Intel Energy Checker SDK User Guide* for details and recommended usage of suffix counters, or use the `--process --help` command line option to display a short explanation of counters and suffix counters.

When processing is applied to counters, the header's top line displays `PROCESSED`, and one or more indicators are illuminated for the gauges having suffix counters. The processing option applies to all monitored PL(s) covered by a given monitor invocation. If the `--process` option is added, recognized suffix counters are not displayed in separate gauges; they are part of the counter the suffix(es) depend on.

NOTE

If a single monitor is started that tries to process (`--process` option invoked) more than 512 counters in all the PLs, an error message will be displayed (`cannot associate counters`). If multiple PLs were selected on a single command line to start the monitor, running each PL in a separate monitor instances may eliminate the issue.

2.6.2 Counter Value Formatting

`--format` format values with ',' and '.'

By default, the monitor displays unformatted counter values.

Add the `--format` option on the command line to format counter values with a thousands separator prior to displaying them. The comma (",") character indicates thousands; the period (".") is used for the decimal point. In Windows systems, the regional and linguistic settings in the operating system allow for alternate characters to be used for these purposes.

There are no parameters for this option. To enable formatting, include the following option on the command line (along with `--process`) as shown below.

```
--process --format
```

NOTE

Activating formatting reduces the number of digits displayed in the gauge's value box. If the value box does not display all the relevant digits, either change the gauge's size (using the `--geometry` option) or remove the formatting option.

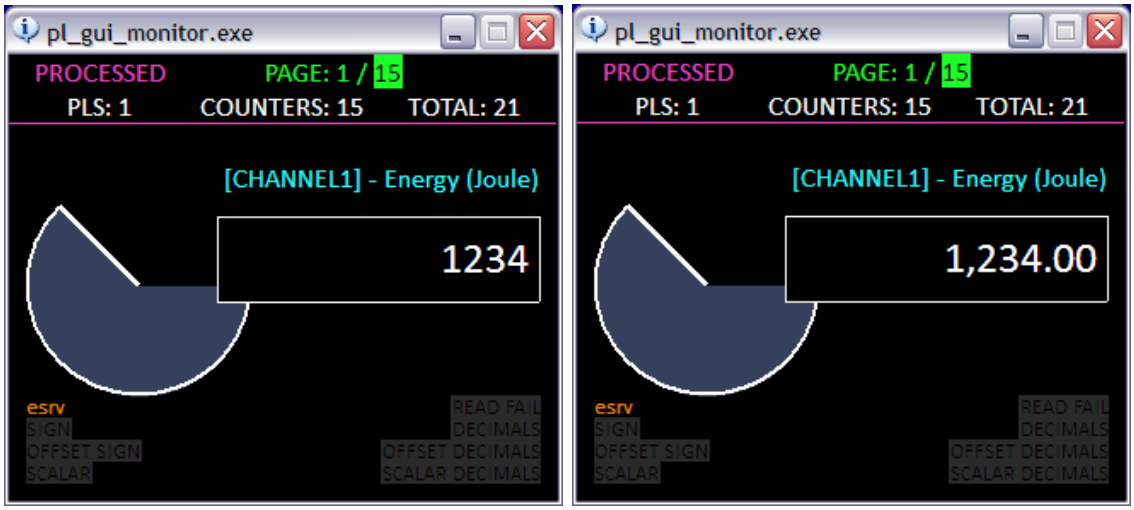

Figure 10: Unformatted *vs.* formatted counter value display

2.6.3 Sampling Frequency

`--t <seconds>` add `<seconds>` pause between reading the counters

By default, the monitor samples the counters at 1 Hz frequency (once per second).

To lower this frequency, use the `--t <second(s)>`. This option introduces a pause (in seconds) between each sample collection. The minimum value for the pause time is one (1) second. There is no upper limit for the pause time.

For example, to set the time between updates at five seconds, include the following command line option:

```
--t 5
```

NOTE

When used prior to the `--t` option, `--time_in_ms` option forces pl_gui_monitor to interpret the time interval as milliseconds rather than seconds. Using this option will translate into higher resource utilization and should therefore be avoided in general.

2.6.4 Display Options

The following command line options affect how the counters are displayed.

2.6.4.1. Window Title

`--title` add a title to the top of the monitor window

To set a custom title for the graphical window, use the `--title` option. For example, a monitor window used to track database statistics might include the following in the list of pl_gui_monitor options:

```
--title "Database Statistics"
```

2.6.4.2. Geometry Options

`--geometry` change how the gauges are arranged on the screen

A monitor displays its gauges in a single window, which might include multiple pages.

By default, the monitor displays the PL counters as a maximum of eight gauges in four (4) rows by two (2) columns, as is shown in Figure 11 below.

Figure 11: Default monitor window configuration

Each gauge is displayed in a 250 pixels wide by 200 pixels high area, and the monitor window is freely positioned by the operating system.

NOTE

Gauges/counters are displayed on the active page from left to right and top to bottom in their PL order (or the order in which they were provided to the `pl_open` *function by the exporting application).*

The `--geometry` option provides a way to change the display configuration by providing an options string typed between double quotes ("). This options string is parsed by the monitor at startup to adapt its display. Three (3) sets of options are recognized: gauges, size, and position.

Gauges Option

`--geometry "gauges=<rows>x<columns>"` set the gauge grid geometry

The syntax for the `gauges` option is: `gauges=<row(s)>x<column(s)>`. The minimum configuration is 1x1. The maximum configuration is limited by the screen resolution and the size of the gauges. Figure 12 shows the default configuration and two custom gauge geometry configurations for the same PL. Note that the number of gauges impacts the number of pages displayed.

As an example, a 3x2 arrangement would be configured with the following command line parameters:

```
--geometry "gauges=3x2"
```

NOTE

There is no space between the values and the `x` *character in the options string.*

Figure 12: Three different gauge configurations for the same PL

Size Option

--geometry "size=<w>x<h>" set the size of each gauge

The syntax for the `size` option is: `size=<width>x<height>` in pixels. The minimum size is 200x200 pixel(s). No test is performed on the maximum size. Changing the height of the gauge may introduce display deformation. Figure 13 shows gauge width changes.

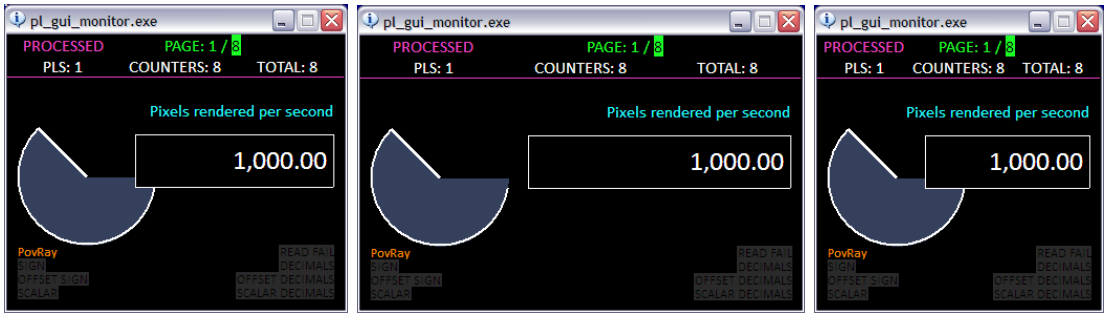

Figure 13: Different gauge widths (default, double, and minimum width)

For example, to set the geometry options for the middle window (the middle monitor) in Figure 13, include the following parameters on the command line when you create its monitor:

```
--geometry "gauges=1x1 size=500x200"
```

Position Option

--geometry "position=<vertical position>x<horizontal position>" position the monitor windows on the screen

The syntax for the position option is: `position=<vertical position>x<horizontal position>`, and both positions are strings. The available strings for vertical positions are: `top`, `center` and `bottom`. The available strings for horizontal positions are: `left`, `center` and `right`. For example, to position the monitor window in the bottom right of the screen, include the following parameters on the command line:

```
--geometry "position=bottomxright"
```

By default, if no positioning options are specified, the monitor window is positioned by the operating system.

In a non-Windows system, most window managers will support the `position` option, but some may not. If the `position` option is not supported by the OS, the following error message is displayed:

```
WARNING: --geometry Position May Not Be Honored By Window Manager Under X11.
```

2.6.4.3. Startup Page Display

--page display a selected page when the monitor starts

By default, the monitor displays the first page of gauges. The `--page <p>` command line option requests the monitor to display page `p` at startup. If `p` is out of range, the first page is displayed at start-up.

For example, to start with page five (5), include the following option on the command line

```
--page 5
```

2.6.4.4. Transparency

--transparency make the window transparent

The monitor window is displayed opaque by default. The `--transparency <n>` option on the command line sets the monitor window from fully opaque (n=0) to invisible (n=100). Figure 14 shows a monitor window invoked with the following option:

```
--transparency 60
```

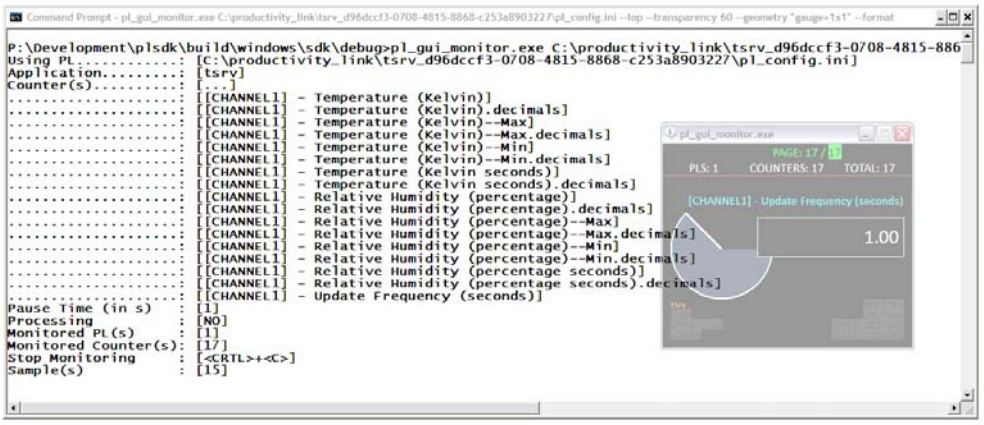

Figure 14: Transparent display (60% transparency)

Note that it is possible to toggle opaque and transparent display during runtime (see section 2.3.2 for details).

In non-Windows systems, transparency support requires participation from the window manager, so the following warning is displayed if transparency is requested:

```
WARNING: --transparency Option Requires A Layered Window Manager Under X11
   (such as Beryl or Compiz).
```

2.6.4.5. Display Font (Windows-only)

--font use the selected font for display

In a Windows system, the monitor defaults to the Microsoft Calibri* font. If this font is not present on the system, then the default system font is used instead. To use a specific font, add the `--font <font_name>` option to the command line, where `font_name` is the name of the font to use. For example, to use the Courier font, include the following on the monitor command line:

```
--font Courier
```

NOTE

If the font name contains spaces, then use double quotes (") around it. For example: "DejaVu LGS Sans".*

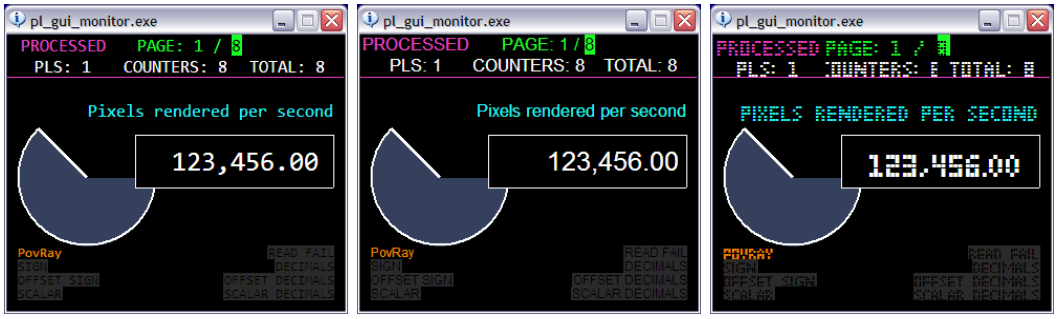

Figure 15: Various fonts used to display gauges

In non-Windows systems, the monitor requires the Helvetica font and does not provide a way to choose an alternate font. If the `--font` option is used under X11, the following error message is displayed:

```
WARNING: --font Option Is Not Supported Under X11.
```

If the Helvetica font is not installed, the following error is displayed on a non-Windows system:

```
WARNING: Please Install The Following Fonts On Your System:
WARNING:     *-adobe-helvetica-medium-r-normal*-24-*.
WARNING:     *-adobe-helvetica-medium-r-normal*-12-*.
```

2.6.4.6. Z Positioning (Windows-only)

`--top` stay on top of other windows

Under normal circumstances, the monitor window follows the standard depth (Z) positioning of the operating system. However, it sometimes is desirable to ensure the monitor application always remains on top. Using the `--top` option on the command line tells the monitor window to remain on top of other applications.

The `--top` option is only supported on Windows systems. If this option is specified on a non-Windows system, the following error message is displayed:

```
WARNING: --top Option Is Not Supported Under X11.
```

2.6.4.7. Anti-Aliasing Support (Windows-only)

`--gdiplus` apply anti-aliasing on arcs

Under Windows, the monitor draws gauge arcs using GDI non-anti-aliased graphic primitives by default. Using the `--gdiplus` option on the command line applies anti-aliasing while drawing gauge arcs. Figure 16 shows an anti-aliased arc on the left side compared with the default mode on the right side.

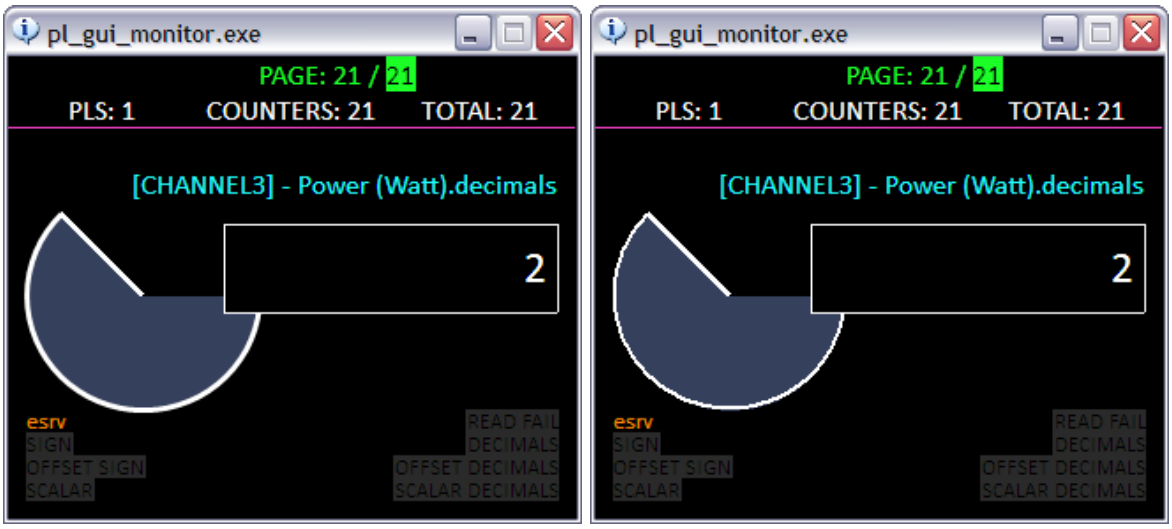

Figure 16: Gauge arc display with and without anti-aliasing

Anti-aliasing requires GDI+ on the system. If the `--gdiplus` option is used and GDI+ is not installed, this option will be ignored.

If the `--gdiplus` option is specified on a non-Windows system; the following error message is displayed:

```
WARNING: --gdiplus Option Is Not Supported Under X11.
```

2.6.5 Trend Indicator

`--trend` trend gauge data

When specified, the `--trend` option forces pl_gui_monitor to display a trend indicator for each gauges. A trend value is computed for each gauge based on its current value. The trend value is an instantaneous estimation of the future value monitored by a gauge.

The forecast window is set to one sample interval by default. It is possible to increase the forecast window size by specifying a value (in sample intervals) after the `--trend` option. For instance, `--trend 5` will display a forecasted value for five (5) sample intervals. A trend indicator is useful to visually estimate the dynamic of the monitored counters.

A trend indicator is displayed as a magenta arc on the perimeter of the gauges as shown in figure Figure 17 and Figure 18. The forecasted value is represented by the magenta arc's end which is not in contact with the gauge needle. When the forecasted value is less than the current value, the arc is drawn counter-clockwise. It is drawn

clockwise when the forecasted value is increasing. When the forecasted value is higher than the maximal value memorized by a gauge, a single magenta notch is displayed.

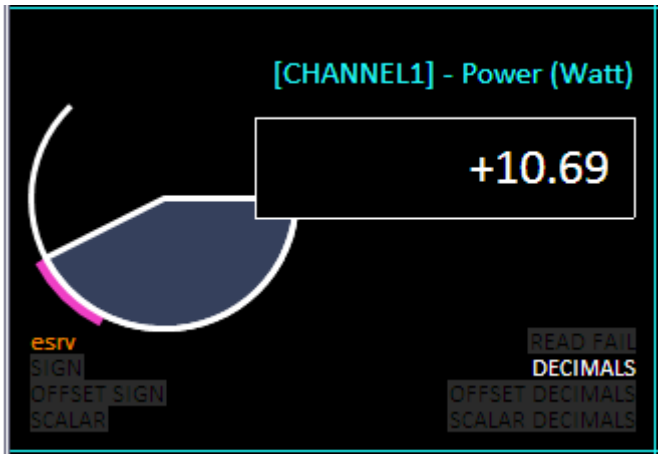

Figure 17: Typical trend display

NOTE

Trend indicators are displayed only when pl_gui_monitor detects a value change of more than one (1). It is not possible to change the threshold in this version of the program.

NOTE

Trend indicators are only displayed, never outputted.

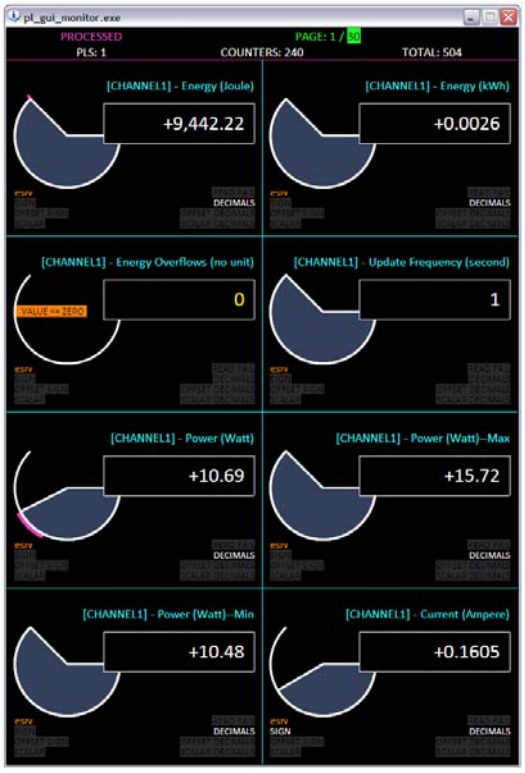

Figure 18: Trend indicator visually highlights dynamic counters

2.6.6 Output Options

2.6.6.1. HTML5/Canvas

`--html5_canvas` export display data into html5 file

To export the display of the active page into an HTML5 file, use the `--html5_canvas` option. For example, a monitor window used to track database statistics might include the following in the list of pl_gui_monitor options:

```
--html5_canvas "C:\Inetpub\wwwroot\default.htm"
```

pl_gui_monitor uses Canvas to generate a rendering program that reproduces the display of the active page. Therefore, the browser used to display the output HTML5 page needs to support Canvas, otherwise no display will be visible. As per publication date, the following web browsers were tested successfully:

- Microsoft* Internet Explorer 9 Beta
- Mozilla* FireFox 3.6.11
- Google* Chrome 7.0.517.41
- Apple* Safari 5.0.1 (7533.17.8)
- Opera Software ASA* Opera 10.63 (3516)

The HTML5 code generated by pl_gui_monitor incorporates an auto-update feature. The re-load time interval is set as the sampling interval of pl_gui_monitor (which is one second by defaults). It is recommended to use lower sampling frequencies, so the system resources are not impacted too severely. The following HTML tag is used:

```
<meta http-equiv="refresh" content="1">
```

NOTE

Different web browser may have different interpretation and/or implementation of the HTML5/Canvas standards. Thus, the result of the Canvas code rendering may slightly vary.

2.6.6.2. Example HTML5/Canvas Displays

Below are some example web browser displays in a Windows* environment. Note that Microsoft* Internet Explorer versions prior to version 9 do not have support for Canvas and therefore cannot be used to display the HTML5/Canvas output of pl_gui_monitor.

Figure 19: Microsoft* Internet Explorer 9 Beta

Figure 20: Mozilla* FireFox 3.6.11

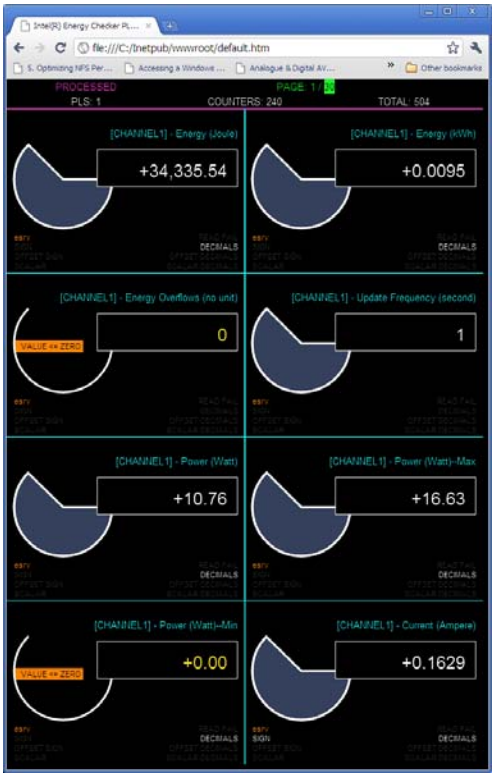

Figure 21: Google* Chrome 7.0.517.41

Figure 22: Apple* Safari 5.0.1 (7533.17.8)

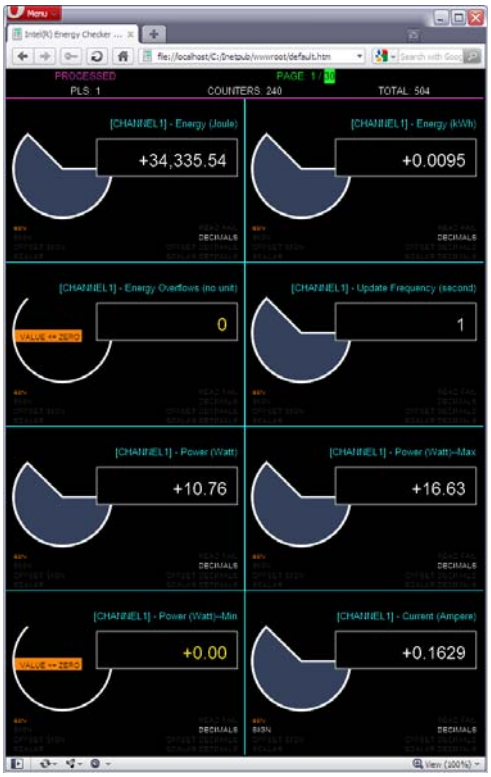

Figure 23: Opera Software ASA* Opera 10.63 (3516)

2.7 Example Monitor Displays

Most of the PL GUI Monitor screen shots in this guide are from a Windows* environment. Below are some example PL GUI Monitor displays in MacOS* X, Solaris* 10, and Linux* environments:

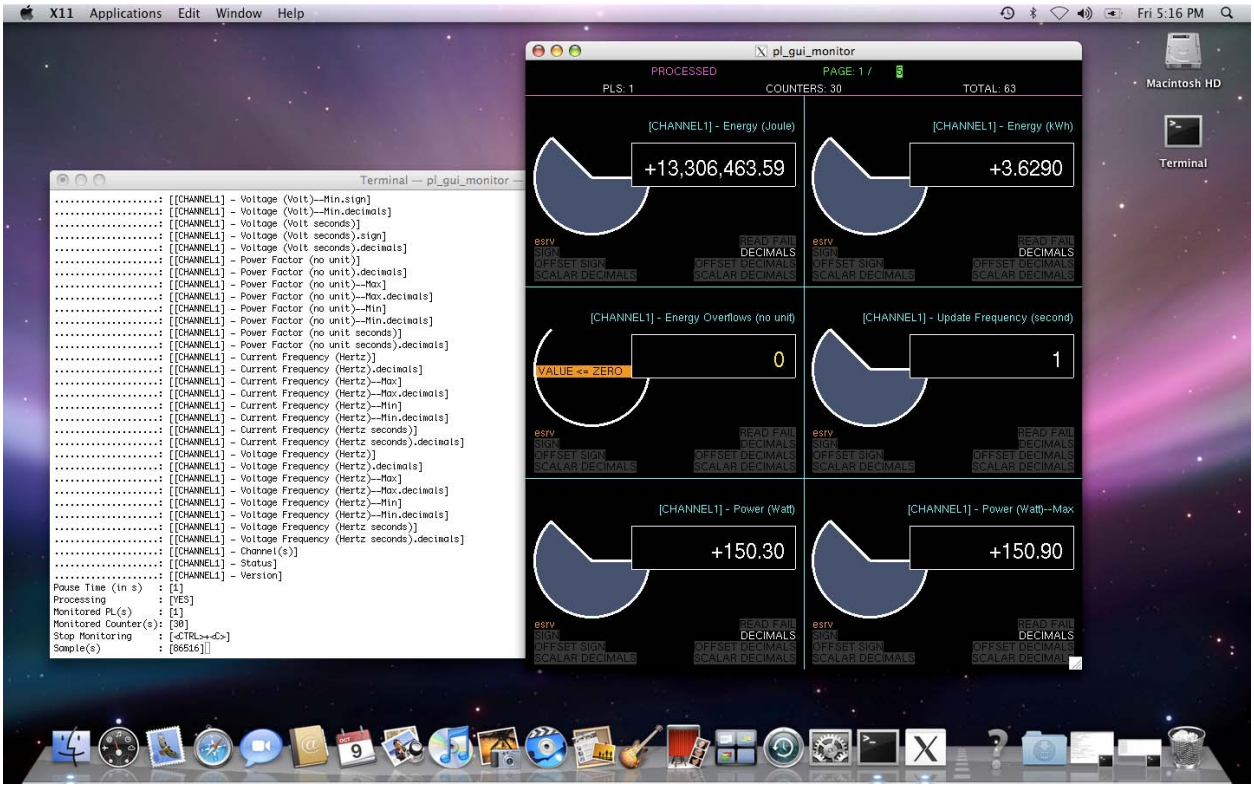

Figure 24: Example monitor display in MacOS X

Figure 25: Example monitor display in Solaris 10

Figure 26: Example monitor display in Linux

2.8 Example Monitor Command Line and Help

2.8.1 Example Monitor Command Line Invocation

The following command line invoked the monitor window shown in Figure 27

```
1  pl_gui_monitor c:\productivity_link\PovRay_76b5e071-34be-47fc-8f26-
2  931157d58f5e\pl_config.ini --process --format --geometry "gauges=4x4 size=200x200
3  position=topxright" --font "Arial" --gdiplus --top --transparency 10 --page 2
```

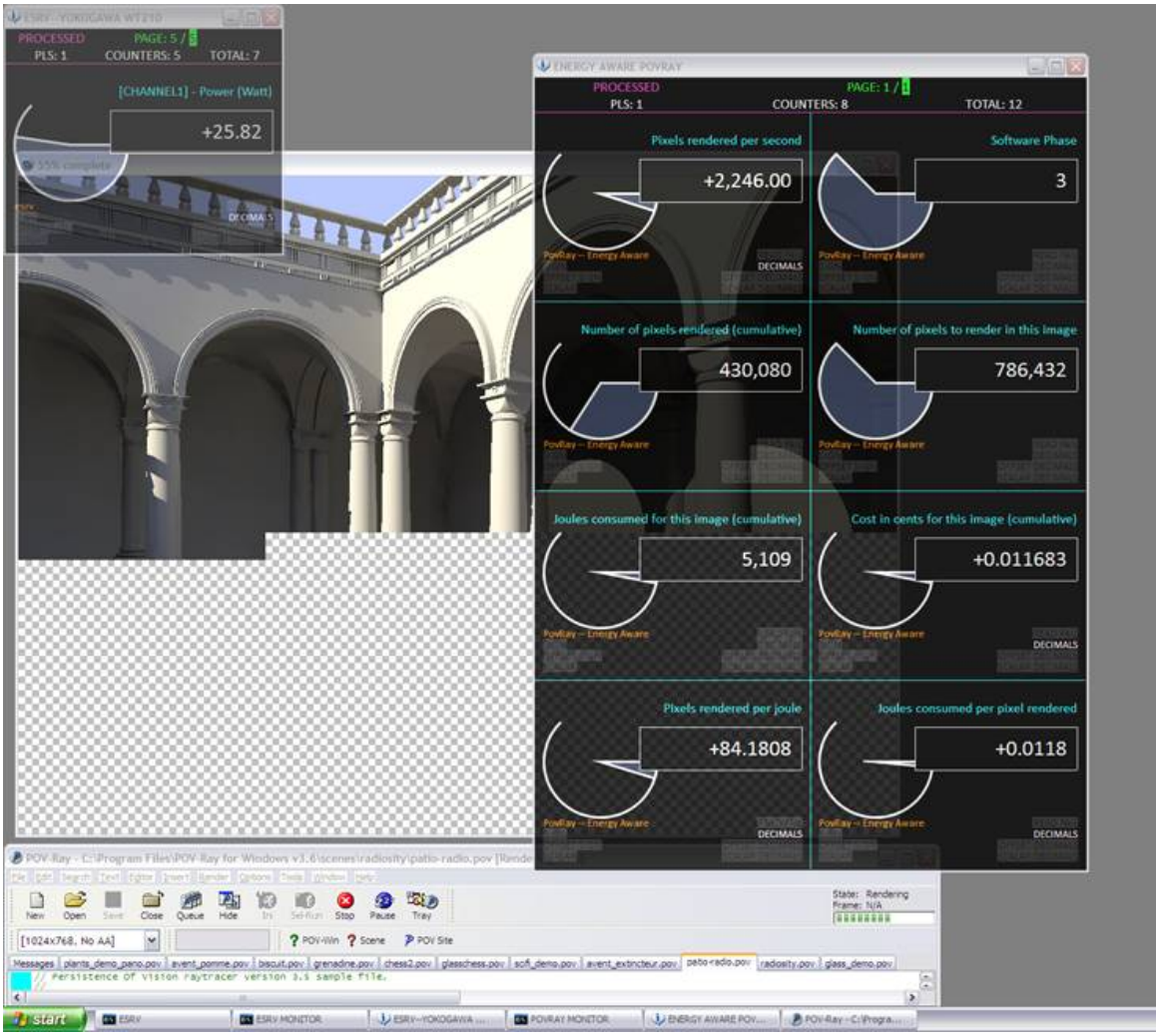

Figure 27: Example monitor invocation

2.8.2 Monitor Syntax and Help

The `--help` option displays basic help on the monitor's syntax. The listing below was obtained using this option.

```
 1
 2  Start PL GUI monitor monitor and display data in graphical format.
 3
 4  Usage:  pl_gui_monitor [[PL_ini] ... ] [--t <n>] [--font <font_name>] ...
 5             [--geometry "options_string"] [--title <name>] ...
 6             [--format] [--process] [--top] [--transparency <x>] --gdiplus
 7             [--page <p>] [--trend [n]] [--html5_canvas <o>]
 8  Or      pl_gui_monitor [--version]
 9  Or      pl_gui_monitor [--help]
10
11     [PL_ini] is one or more fully qualified path(s) to PL configuration file(s).
12     --t with <n> sets the pause time between each log entry creation.
13        n is expressed in seconds.  n is equal to 1 by default.
14     --font with <font_name> sets the font (Windows*-only).
15        Calibri* font is used by default.  If the font doesn't exist, then
16        default Windows font is used.
17     --title with <name> sets the title bar's content to name.
18        Use this option to identify easily the multiple monitors' data sources.
19        For example: "ESRV Monitoring YOKOGAWA WT3000"*.
20     --geometry with <"options_string"> sets the display options.
21        Use  pl_gui_monitor  --geometry  --help  for  more  information  on  the  display
22  settings.
23     --format displays values using digit group separator (a.k.a. thousands separator).
24     --process forces additional analysis on counter values with prescribed suffixes.
25        Use pl_gui_monitor --process --help for more information on the suffixes.
26     Use --top displays the monitor window always on top (Windows-only).
27     --transparency with <x> sets the transparency of the monitor window.
28        x is expressed in percentage.  0 means opaque, 100 means invisible.
29        Double-click in the monitor window to toggle opaque / transparent display.
30     --page with <p> sets the active page displayed.
31        if p is out-of-range, then first page is displayed.  Range depends on geometry
32  options.
33     --trend with optional [n] activates the counters trend computation and display.
34        n indicates the number of samples used to forecast the trend (1 by default).
35     --html5_canvas  with  <o>  generates  an  html  page  named  (named  <o>,  o  may  be  a
36  qualified path).
37        The  html  page  is  generated  at  the  sampling  frequency  and  reflects  the  active
38  page.
39        This file can be published on the Web for remote monitoring.
40        The page has an auto-reload statement set to the sampling frequency.
41        Note: the picture rendering requires a Javascripts* and canvas* API compatible
42  browser.
43     --gdiplus draws gauges' arc with anti-aliasing (Windows-only).
44     --version prints version information.
```

```
   --help prints this help message.

The monitor application continues to display data once per second (or
whatever time interval is selected with the --t option).


NOTE
   The following keys are active:
      A:  display max values.
      I:  display min values.
      V:  display average values.
      R:  reset max, min, average values and samples count.
      M:  set max values to their type max.
      F:  freeze the display.
      C:  print counters and values to standatd output.
      +, dow, right, page down:  increase active page number.
      -, up, left, page up:  decrease active page number.
    ESC:  stop monitor.

   * Third-party trademarks are the property of their respective owners.
```

The `--geometry --help` option displays additional help on the monitor's geometry options. The listing below was obtained using this option.

```
Additional help information on the display geometry settings.

   The following options are recognized in the geometry options string
   when the --geometry parameter is supplied:

      gauges
      size
      position

Gauges
------
      The monitor displays 4 rows of 2 gauges per page.  Different
   configurations can be specified as row(s) x column(s).  For example,
   to request 3 rows and 1 column, use "gauges=3x1" string.
   Dimensions cannot be less than 1 or more than what the screen can accommodate.

Size
----
      The monitor displays gauges in 250 x 200 pixels.  Different
   gauge size can be specified as width x height pixel(s).  For example,
   to request gauge width of 400, and gauge height of 250 pixels, use
   "size=250x200" string.
   Width cannot be less than 200 pixels, height cannot be less than 200 pixels.

```

```
Position
--------
      By default, the monitor is positioned by Windows*.  Different positioning
   can be specified as vertical position x horizontal position.  For example,
   to request centered monitor, use "position=centerxcenter" string.
   Available vertical positions are: top, center and bottom.
   Available horizontal positions are: left, center and right.


NOTE
       Gauges' geometry is dynamically computed based on the geometry options
   provided with the --geometry parameter.
       Gauges and size options can be combined.  For examples
       --geometry "gauges=1x1 size=500x400"
```

The `--process --help` option displays basic help on the monitor's counter processing mechanism. The listing below was obtained using this option.

```
Additional help information on the counter suffixes supported by the monitor.

   The following suffixes are recognized by monitor when the --process
   parameter is supplied:

      .decimals (defaults to 0 if suffix counter doesn't exist)
      .scalar (defaults to 1 if suffix counter doesn't exist)
      .scalar.decimals (defaults to 0 if suffix counter doesn’t exist)
      .sign (defaults to 0 if suffix counter doesn't exist)
      .offset (defaults to 0 if suffix counter doesn't exist)
      .offset.decimals (defaults to 0 if suffix counter doesn't exist)
      .offset.sign (defaults to 0 if suffix counter doesn't exist)

   The following formula is used to compute a "total" counters' real values:

      real_total = total
                 / (10 ^ total.decimals)
                 * total.scalar
                 / (10 ^ total.scalar.decimals)
                 * (total.sign ? -1 : 1)
                 + total.offset
                 / (10 ^ total.offset.decimals)
                 * (total.offset.sign ? -1 : 1)
      ;

Sign Suffix
-----------
      For maximum portability across different architectures,
   the Intel EC SDK always uses positive values for its counters,
   but negative values can easily be represented with PL counters
```

```
   and interpreted by other applications if the following
   convention is adhered to:
      1.  Add a supplemental counter with a ".sign" suffix
      2.  Write the static sign to the supplemental counter
          using the following convention:
              1 means a negative number.
              0 means a positive number (or zero).
   If the sign suffix is omitted, the number is assumed to be
   positive.

Decimals Suffix
---------------
      The EC SDK always uses unsigned long long values for its counters,
   but floating point values with a fixed number of decimal places can
   easily be represented with PL counters and interpreted by other
   applications if the following convention is adhered to:
      1.  Add a supplemental counter with a ".decimals" suffix
      2.  Write the static number of decimal places to the supplemental
          counter
      3.  Adjust counters by 10 ^ n before writing to PL

   For example, the ESRV utility makes use of this with its
   "Energy (kWh)" counter.  To represent the value to two decimal
   places, the actual number of kiloWatt-hours is multiplied by 100 and
   stored as an unsigned long long.  In addition, at program startup,
   ESRV writes a value of 2 to the "Energy (kWh).decimals" counter to
   indicate that "Energy (kWh)" should be interpreted as having two
   decimal places.

   All applications writing counters representing fixed decimal numbers
   should use the .decimals suffix on a supplemental static counter as
   appropriate.  The number of decimal places should not be changed
   after it is written.

Offset Suffix
-------------
      In some cases, the native source of a counter may represent a
   differential value from some fixed offset value.  Rather than
   constantly adding the differential value to the offset value each
   time the counter is written out, an application developer could
   elect to define a static offset counter and let any consumers of the
   PL data add in the offset where needed.  To do this, the following
   convention has been established:
      1.  Add a supplemental counter with a ".offset" suffix
      2.  Write the static value of the offset to the supplemental
          counter
   For example, if an application is counting the number of visitors
   over 5000 to a theme park, a "Visitor Total" value of 800 might
   represent a total attendance of 5800 visitors.  By adding a
```

```
   "Visitor Total.offset" counter and writing a value of 5000 to that
   offset counter, the application developer can indicate what number
   to add to the counter value to get the real value.
      The Offset suffix may have additional suffixes.  Currently, the
   .sign and .decimal suffixes are supported by this logger.

Scalar Suffix
-------------
      In most cases, the decimals suffix provides sufficient ability
   to scale the way numbers are reported out.  However, there are some
   cases where the counter values written through PL need some sort of
   scaling factor applied to them.  In such cases, the following
   convention should be used:
      1.  Add a supplemental counter with a ".scalar" suffix
      2.  Write the static value of the scaling factor to the
          supplemental counter
   For example, software for an egg producer could report how many
   dozens of eggs were produced with an "Egg Dozens" counter.
   That same value could be written to an "Eggs" counter, provided
   that "Eggs.scalar" had been set up with a value of twelve.
   When units are measured by the thousands or millions, using a
   scalar factor of 1000 or 1,000,000 as appropriate will represent
   the proper scale of the units.

   The scalar suffix also supports the .decimals suffix (i.e.,
   .scalar.decimals) to allow scalar values to represent values
   other than integers.


NOTE
    The static counters can be used in a compound manner.
    For example, consider the following:
       MyTotal counter has a value of 5
       MyTotal.decimals has a value of 1
       MyTotal.scalar has a value of 72

   In this example, the actual value is 36:
       5 / (10 ^ 1) * 72 = 36

```

This page intentionally blank.

3 PL CSV Logger

The PL CSV Logger (or logger) is a console application that allows users to simultaneously log one or multiple Productivity Links (PLs) in real time into a Comma Separated Value (CSV) text file. PL logger was created as both a tool and as a reference application. The source code of the logger is provided in the SDK and demonstrates how to use suffix counters.

3.1 Starting the Logger

Every Productivity Link (PL) created by the Intel Energy Checker SDK generates a configuration file (typically named `pl_config.ini`) in a special directory dedicated to that PL. The logger application reads the information in this configuration file to dump the PL counter information to a file.

In its simplest form, the logger can be invoked with a single argument: the qualified path to a PL's configuration file (`pl_config.ini`). For example:

```
pl_csv_logger c:\productivity_link\PovRay_76b5e071-34be-47fc-8f26-
   931157d58f5e\pl_config.ini.
```

NOTE

Use double quotes (") around the `pl_config.ini` *path if it contains spaces.*

NOTE

For details on PLs, PL counters and PL configuration files please refer to the Intel Energy Checker SDK User Guide.

The monitor is designed for IT professionals and power users. Due to the large number of options that can be provided on the command line, it is anticipated that the monitor will usually be invoked from a script/batch file or from a higher-level application.

It is possible to log multiple PLs with various logger instances. However, it is generally preferable to use a single logger to gather related data from multiple PLs so the data from various PLs are roughly synchronized when recorded. This will eliminate the need to correlate individually recorded data from multiple loggers. Up to ten (10) PLs can be monitored by a single monitor instance.

Examples:

```
pl_csv_logger c:\productivity_link\PovRay_76b5e071-34be-47fc-8f26-
931157d58f5e\pl_config.ini c:\productivity_link\esrv_f6bde0a1-3ebe-55fc-8f26-
9f1cc7de8f5e\pl_config.ini
```

```
    ./pl_csv_logger /opt/productivity_link/esrv_f1fe2268-0d94-43ad-916c-
c3ea1d18a656/pl_config.ini
```

NOTE

If the same PL configuration file is specified multiple times, the logger uses the configuration file only once.

At startup, a logger attaches to the PL(s) specified by the user and starts logging the PL(s)' counters and their values with a 1 Hz refresh frequency (once per second). By default, the logger prints the logged data to the standard output (stdout).

3.2 Specifying an Output File

If only PL configuration files are specified on the command line, the logger will dump its output to the standard output (console screen unless redirected). However, the typical use case for the logger is to dump time-stamped counter information to a dedicated CSV file.

To identify the output filename, use the `--output <filename>` parameter, where `<filename>` is a fully qualified output filename (typically with a ".csv" extension) without the angle brackets. For example, to log the data to a "C:\my data\mylogdata.csv" file on a Windows machine, include the following on the command line:

```
--output "C:\my data\mylogdata.csv"
```

NOTE

Use double quotes (") around the `<filename>` *path if it contains spaces.*

3.3 Logger Output Overview

The output of the logger scales with the number of counters in the PLs. The first sub-section presents the output when a dedicated output file is specified and the second sub-section shows a more sophisticated example with more counters.

3.3.1 Simple Logger Example (TSRV)

The listing below shows the logger's typical output to stdout when a PL with 20 counters is used and the logger output is explicitly directed to a separate file with the `--output` option:

```
1
2    @@@@   @@@@  @    @        @       @@@@   @@@@   @@@@  @@@@@@ @@@@@
3   @    @ @    @ @    @        @       @    @ @    @ @    @ @      @    @
4   @      @      @    @        @       @    @ @      @      @      @    @
5   @        @@@@  @    @        @       @    @ @      @      @@@@@@ @@@@@
6   @           @  @   @        @       @    @ @  @@@ @  @@@ @      @  @
7   @    @      @   @ @         @       @    @ @    @ @    @ @      @   @
8    @@@@  @@@@@     @          @@@@@@  @@@@   @@@@   @@@@  @@@@@@ @    @
9
```

```
Using PL.........: [c:\productivity_link\tsrv_d85a26a0-ebbd-4466-bb30-0caa607973
43\pl_config.ini]
Application......: [tsrv]
Counter(s).......: [...]
.................: [C:\productivity_link\tsrv_d85a26a0-ebbd-4466-bb30-0caa607973
43\[CHANNEL1] - Temperature (Kelvin)]
.................: [C:\productivity_link\tsrv_d85a26a0-ebbd-4466-bb30-0caa607973
43\[CHANNEL1] - Temperature (Kelvin).decimals]
.................: [C:\productivity_link\tsrv_d85a26a0-ebbd-4466-bb30-0caa607973
43\[CHANNEL1] - Temperature (Kelvin)--Max]
.................: [C:\productivity_link\tsrv_d85a26a0-ebbd-4466-bb30-0caa607973
43\[CHANNEL1] - Temperature (Kelvin)--Max.decimals]
.................: [C:\productivity_link\tsrv_d85a26a0-ebbd-4466-bb30-0caa607973
43\[CHANNEL1] - Temperature (Kelvin)--Min]
.................: [C:\productivity_link\tsrv_d85a26a0-ebbd-4466-bb30-0caa607973
43\[CHANNEL1] - Temperature (Kelvin)--Min.decimals]
.................: [C:\productivity_link\tsrv_d85a26a0-ebbd-4466-bb30-0caa607973
43\[CHANNEL1] - Temperature (Kelvin seconds)]
.................: [C:\productivity_link\tsrv_d85a26a0-ebbd-4466-bb30-0caa607973
43\[CHANNEL1] - Temperature (Kelvin seconds).decimals]
.................: [C:\productivity_link\tsrv_d85a26a0-ebbd-4466-bb30-0caa607973
43\[CHANNEL1] - Relative Humidity (percentage)]
.................: [C:\productivity_link\tsrv_d85a26a0-ebbd-4466-bb30-0caa607973
43\[CHANNEL1] - Relative Humidity (percentage).decimals]
.................: [C:\productivity_link\tsrv_d85a26a0-ebbd-4466-bb30-0caa607973
43\[CHANNEL1] - Relative Humidity (percentage)--Max]
.................: [C:\productivity_link\tsrv_d85a26a0-ebbd-4466-bb30-0caa607973
43\[CHANNEL1] - Relative Humidity (percentage)--Max.decimals]
.................: [C:\productivity_link\tsrv_d85a26a0-ebbd-4466-bb30-0caa607973
43\[CHANNEL1] - Relative Humidity (percentage)--Min]
.................: [C:\productivity_link\tsrv_d85a26a0-ebbd-4466-bb30-0caa607973
43\[CHANNEL1] - Relative Humidity (percentage)--Min.decimals]
.................: [C:\productivity_link\tsrv_d85a26a0-ebbd-4466-bb30-0caa607973
43\[CHANNEL1] - Relative Humidity (percentage seconds)]
.................: [C:\productivity_link\tsrv_d85a26a0-ebbd-4466-bb30-0caa607973
43\[CHANNEL1] - Relative Humidity (percentage seconds).decimals]
.................: [C:\productivity_link\tsrv_d85a26a0-ebbd-4466-bb30-0caa607973
43\[CHANNEL1] - Update Frequency (seconds)]
.................: [C:\productivity_link\tsrv_d85a26a0-ebbd-4466-bb30-0caa607973
43\[CHANNEL1] - Channel(s)]
.................: [C:\productivity_link\tsrv_d85a26a0-ebbd-4466-bb30-0caa607973
43\[CHANNEL1] - Status]
.................: [C:\productivity_link\tsrv_d85a26a0-ebbd-4466-bb30-0caa607973
43\[CHANNEL1] - Version]
Pause Time       : [1]
Processing       : [NO]
Logged PL(s)     : [1]
Logged Counter(s): [20]
Saving Log Into  : [temp.csv]
```

```
To Stop Logging  : [<CTRL>+<C>]
Samples          : [3]
```

As shown above, a banner is first printed (`CSV LOGGER`) followed by information on the logged PL(s). For each PL, the name of the application is displayed, followed by a list of all the counters in the PL, an indication if processing is applied to the raw counter values, and the output filename in which the data is stored (`temp.csv` in this example, `stdout` if no `--output` parameter is provided).

The CSV output file is composed of one (1) header line and multiple data line(s). The data output to the CSV file is shown in the listing below:

NOTE

Lines 1-28 in the listing below (in bold) are actually one very long line that wrapped 27 times. Each of the succeeding two lines (in italics) comprise a single line that has wrapped in this listing.

```
"Time Stamp", "C:\productivity_link\tsrv_d85a26a0-ebbd-4466-bb30-0caa60797343\[C
HANNEL1] - Temperature (Kelvin)", "C:\productivity_link\tsrv_d85a26a0-ebbd-4466-
bb30-0caa60797343\[CHANNEL1] - Temperature (Kelvin).decimals", "C:\productivity_
link\tsrv_d85a26a0-ebbd-4466-bb30-0caa60797343\[CHANNEL1] - Temperature (Kelvin)
--Max", "C:\productivity_link\tsrv_d85a26a0-ebbd-4466-bb30-0caa60797343\[CHANNEL
1] - Temperature (Kelvin)--Max.decimals", "C:\productivity_link\tsrv_d85a26a0-eb
bd-4466-bb30-0caa60797343\[CHANNEL1] - Temperature (Kelvin)--Min", "C:\productiv
ity_link\tsrv_d85a26a0-ebbd-4466-bb30-0caa60797343\[CHANNEL1] - Temperature (Kel
vin)--Min.decimals", "C:\productivity_link\tsrv_d85a26a0-ebbd-4466-bb30-0caa6079
7343\[CHANNEL1] - Temperature (Kelvin seconds)", "C:\productivity_link\tsrv_d85a
26a0-ebbd-4466-bb30-0caa60797343\[CHANNEL1] - Temperature (Kelvin seconds).decim
als", "C:\productivity_link\tsrv_d85a26a0-ebbd-4466-bb30-0caa60797343\[CHANNEL1]
 - Relative Humidity (percentage)", "C:\productivity_link\tsrv_d85a26a0-ebbd-446
6-bb30-0caa60797343\[CHANNEL1] - Relative Humidity (percentage).decimals", "C:\p
roductivity_link\tsrv_d85a26a0-ebbd-4466-bb30-0caa60797343\[CHANNEL1] - Relative
 Humidity (percentage)--Max", "C:\productivity_link\tsrv_d85a26a0-ebbd-4466-bb30
-0caa60797343\[CHANNEL1] - Relative Humidity (percentage)--Max.decimals", "C:\pr
oductivity_link\tsrv_d85a26a0-ebbd-4466-bb30-0caa60797343\[CHANNEL1] - Relative
Humidity (percentage)--Min", "C:\productivity_link\tsrv_d85a26a0-ebbd-4466-bb30-
0caa60797343\[CHANNEL1] - Relative Humidity (percentage)--Min.decimals", "C:\pro
ductivity_link\tsrv_d85a26a0-ebbd-4466-bb30-0caa60797343\[CHANNEL1] - Relative H
umidity (percentage seconds)", "C:\productivity_link\tsrv_d85a26a0-ebbd-4466-bb3
0-0caa60797343\[CHANNEL1] - Relative Humidity (percentage seconds).decimals", "C
:\productivity_link\tsrv_d85a26a0-ebbd-4466-bb30-0caa60797343\[CHANNEL1] - Updat
e Frequency (seconds)", "C:\productivity_link\tsrv_d85a26a0-ebbd-4466-bb30-0caa6
0797343\[CHANNEL1] - Channel(s)", "C:\productivity_link\tsrv_d85a26a0-ebbd-4466-
bb30-0caa60797343\[CHANNEL1] - Status", "C:\productivity_link\tsrv_d85a26a0-ebbd
-4466-bb30-0caa60797343\[CHANNEL1] - Version", "Sample #"
Mon Oct 05 16:37:54 2009, 29845, 2, 29904, 2, 29814, 2, 55329095, 2, 6100, 2, 61
00, 2, 6000, 2, 11211600, 2, 1, 1, 1, 20090515, 1
Mon Oct 05 16:37:55 2009, 29864, 2, 29904, 2, 29814, 2, 55358959, 2, 6000, 2, 61
```

```
00, 2, 6000, 2, 11217600, 2, 1, 1, 1, 20090515, 2
Mon Oct 05 16:37:56 2009, 29904, 2, 29904, 2, 29814, 2, 55388863, 2, 6000, 2, 61
00, 2, 6000, 2, 11223600, 2, 1, 1, 1, 20090515, 3
```

The CSV output file can be imported into most spreadsheet applications

3.3.2 Advanced Logger Example (ESRV)

The listing below shows the logger's typical output for a more sophisticated PL on a different system. This listing also shows what happens when a separate output file is not designated with the `--output` command line option.

```

  @@@@   @@@@  @    @        @       @@@@   @@@@   @@@@  @@@@@@ @@@@@
 @    @ @    @ @    @        @       @    @ @    @ @    @ @      @    @
 @      @      @    @        @       @    @ @      @      @      @    @
 @       @@@@  @    @        @       @    @ @      @      @@@@@@ @@@@@
 @           @  @   @        @       @    @ @  @@@ @  @@@ @      @  @
 @    @      @   @ @         @       @    @ @    @ @    @ @      @   @
  @@@@  @@@@@     @          @@@@@@  @@@@   @@@@   @@@@  @@@@@@ @    @


Using PL.........: [/opt/productivity_link/esrv_b230cee8-59bc-49e9-ab17-
bfb36939ef9d/pl_config.ini]
Application......: [esrv]
Counter(s).......: [...]
.................: [[CHANNEL1] - Energy (Joule)]
.................: [[CHANNEL1] - Energy (Joule).decimals]
.................: [[CHANNEL1] - Energy (kWh)]
.................: [[CHANNEL1] - Energy (kWh).decimals]
.................: [[CHANNEL1] - Energy Overflows (no unit)]
.................: [[CHANNEL1] - Update Frequency (second)]
.................: [[CHANNEL1] - Power (Watt)]
.................: [[CHANNEL1] - Power (Watt).decimals]
.................: [[CHANNEL1] - Power (Watt)--Max]
.................: [[CHANNEL1] - Power (Watt)--Max.decimals]
.................: [[CHANNEL1] - Power (Watt)--Min]
.................: [[CHANNEL1] - Power (Watt)--Min.decimals]
.................: [[CHANNEL1] - Current (Ampere)]
.................: [[CHANNEL1] - Current (Ampere).sign]
.................: [[CHANNEL1] - Current (Ampere).decimals]
.................: [[CHANNEL1] - Current (Ampere)--Max]
.................: [[CHANNEL1] - Current (Ampere)--Max.sign]
.................: [[CHANNEL1] - Current (Ampere)--Max.decimals]
.................: [[CHANNEL1] - Current (Ampere)--Min]
.................: [[CHANNEL1] - Current (Ampere)--Min.sign]
.................: [[CHANNEL1] - Current (Ampere)--Min.decimals]
.................: [[CHANNEL1] - Current (Ampere seconds)]
.................: [[CHANNEL1] - Current (Ampere seconds).sign]
.................: [[CHANNEL1] - Current (Ampere seconds).decimals]
```

```
.................: [[CHANNEL1] - Voltage (Volt)]
.................: [[CHANNEL1] - Voltage (Volt).sign]
.................: [[CHANNEL1] - Voltage (Volt).decimals]
.................: [[CHANNEL1] - Voltage (Volt)--Max]
.................: [[CHANNEL1] - Voltage (Volt)--Max.sign]
.................: [[CHANNEL1] - Voltage (Volt)--Max.decimals]
.................: [[CHANNEL1] - Voltage (Volt)--Min]
.................: [[CHANNEL1] - Voltage (Volt)--Min.sign]
.................: [[CHANNEL1] - Voltage (Volt)--Min.decimals]
.................: [[CHANNEL1] - Voltage (Volt seconds)]
.................: [[CHANNEL1] - Voltage (Volt seconds).sign]
.................: [[CHANNEL1] - Voltage (Volt seconds).decimals]
.................: [[CHANNEL1] - Power Factor (no unit)]
.................: [[CHANNEL1] - Power Factor (no unit).decimals]
.................: [[CHANNEL1] - Power Factor (no unit)--Max]
.................: [[CHANNEL1] - Power Factor (no unit)--Max.decimals]
.................: [[CHANNEL1] - Power Factor (no unit)--Min]
.................: [[CHANNEL1] - Power Factor (no unit)--Min.decimals]
.................: [[CHANNEL1] - Power Factor (no unit seconds)]
.................: [[CHANNEL1] - Power Factor (no unit seconds).decimals]
.................: [[CHANNEL1] - Current Frequency (Hertz)]
.................: [[CHANNEL1] - Current Frequency (Hertz).decimals]
.................: [[CHANNEL1] - Current Frequency (Hertz)--Max]
.................: [[CHANNEL1] - Current Frequency (Hertz)--Max.decimals]
.................: [[CHANNEL1] - Current Frequency (Hertz)--Min]
.................: [[CHANNEL1] - Current Frequency (Hertz)--Min.decimals]
.................: [[CHANNEL1] - Current Frequency (Hertz seconds)]
.................: [[CHANNEL1] - Current Frequency (Hertz seconds).decimals]
.................: [[CHANNEL1] - Voltage Frequency (Hertz)]
.................: [[CHANNEL1] - Voltage Frequency (Hertz).decimals]
.................: [[CHANNEL1] - Voltage Frequency (Hertz)--Max]
.................: [[CHANNEL1] - Voltage Frequency (Hertz)--Max.decimals]
.................: [[CHANNEL1] - Voltage Frequency (Hertz)--Min]
.................: [[CHANNEL1] - Voltage Frequency (Hertz)--Min.decimals]
.................: [[CHANNEL1] - Voltage Frequency (Hertz seconds)]
.................: [[CHANNEL1] - Voltage Frequency (Hertz seconds).decimals]
.................: [[CHANNEL1] - Channel(s)]
.................: [[CHANNEL1] - Status]
.................: [[CHANNEL1] - Version]
Pause Time       : [1]
Processing       : [NO]
Logged PL(s)     : [1]
Logged Counter(s): [63]
Saving Log Into  : [stdout]
"Time Stamp", "/opt/productivity_link/esrv_b230cee8-59bc-49e9-ab17-bfb36939ef9d/
[CHANNEL1] - Energy (Joule)", "/opt/productivity_link/esrv_b230cee8-59bc-49e9-ab
17-bfb36939ef9d/[CHANNEL1] - Energy (Joule).decimals", "/opt/productivity_link/e
srv_b230cee8-59bc-49e9-ab17-bfb36939ef9d/[CHANNEL1] - Energy (kWh)", "/opt/produ
ctivity_link/esrv_b230cee8-59bc-49e9-ab17-bfb36939ef9d/[CHANNEL1] - Energy (kWh)
```

```
.decimals", "/opt/productivity_link/esrv_b230cee8-59bc-49e9-ab17-bfb36939ef9d/[C
HANNEL1] - Energy Overflows (no unit)", "/opt/productivity_link/esrv_b230cee8-59
bc-49e9-ab17-bfb36939ef9d/[CHANNEL1] - Update Frequency (second)", "/opt/product
ivity_link/esrv_b230cee8-59bc-49e9-ab17-bfb36939ef9d/[CHANNEL1] - Power (Watt)",
 "/opt/productivity_link/esrv_b230cee8-59bc-49e9-ab17-bfb36939ef9d/[CHANNEL1] -
Power (Watt).decimals", "/opt/productivity_link/esrv_b230cee8-59bc-49e9-ab17-bfb
36939ef9d/[CHANNEL1] - Power (Watt)--Max", "/opt/productivity_link/esrv_b230cee8
-59bc-49e9-ab17-bfb36939ef9d/[CHANNEL1] - Power (Watt)--Max.decimals", "/opt/pro
ductivity_link/esrv_b230cee8-59bc-49e9-ab17-bfb36939ef9d/[CHANNEL1] - Power (Wat
t)--Min", "/opt/productivity_link/esrv_b230cee8-59bc-49e9-ab17-bfb36939ef9d/[CHA
NNEL1] - Power (Watt)--Min.decimals", "/opt/productivity_link/esrv_b230cee8-59bc
-49e9-ab17-bfb36939ef9d/[CHANNEL1] - Current (Ampere)", "/opt/productivity_link/
esrv_b230cee8-59bc-49e9-ab17-bfb36939ef9d/[CHANNEL1] - Current (Ampere).sign", "
/opt/productivity_link/esrv_b230cee8-59bc-49e9-ab17-bfb36939ef9d/[CHANNEL1] - Cu
rrent (Ampere).decimals", "/opt/productivity_link/esrv_b230cee8-59bc-49e9-ab17-b
fb36939ef9d/[CHANNEL1] - Current (Ampere)--Max", "/opt/productivity_link/esrv_b2
30cee8-59bc-49e9-ab17-bfb36939ef9d/[CHANNEL1] - Current (Ampere)--Max.sign", "/o
pt/productivity_link/esrv_b230cee8-59bc-49e9-ab17-bfb36939ef9d/[CHANNEL1] - Curr
ent (Ampere)--Max.decimals", "/opt/productivity_link/esrv_b230cee8-59bc-49e9-ab1
7-bfb36939ef9d/[CHANNEL1] - Current (Ampere)--Min", "/opt/productivity_link/esrv
_b230cee8-59bc-49e9-ab17-bfb36939ef9d/[CHANNEL1] - Current (Ampere)--Min.sign",
"/opt/productivity_link/esrv_b230cee8-59bc-49e9-ab17-bfb36939ef9d/[CHANNEL1] - C
urrent (Ampere)--Min.decimals", "/opt/productivity_link/esrv_b230cee8-59bc-49e9-
ab17-bfb36939ef9d/[CHANNEL1] - Current (Ampere seconds)", "/opt/productivity_lin
k/esrv_b230cee8-59bc-49e9-ab17-bfb36939ef9d/[CHANNEL1] - Current (Ampere seconds
).sign", "/opt/productivity_link/esrv_b230cee8-59bc-49e9-ab17-bfb36939ef9d/[CHAN
NEL1] - Current (Ampere seconds).decimals", "/opt/productivity_link/esrv_b230cee
8-59bc-49e9-ab17-bfb36939ef9d/[CHANNEL1] - Voltage (Volt)", "/opt/productivity_l
ink/esrv_b230cee8-59bc-49e9-ab17-bfb36939ef9d/[CHANNEL1] - Voltage (Volt).sign",
 "/opt/productivity_link/esrv_b230cee8-59bc-49e9-ab17-bfb36939ef9d/[CHANNEL1] -
Voltage (Volt).decimals", "/opt/productivity_link/esrv_b230cee8-59bc-49e9-ab17-b
fb36939ef9d/[CHANNEL1] - Voltage (Volt)--Max", "/opt/productivity_link/esrv_b230
cee8-59bc-49e9-ab17-bfb36939ef9d/[CHANNEL1] - Voltage (Volt)--Max.sign", "/opt/p
roductivity_link/esrv_b230cee8-59bc-49e9-ab17-bfb36939ef9d/[CHANNEL1] - Voltage
(Volt)--Max.decimals", "/opt/productivity_link/esrv_b230cee8-59bc-49e9-ab17-bfb3
6939ef9d/[CHANNEL1] - Voltage (Volt)--Min", "/opt/productivity_link/esrv_b230cee
8-59bc-49e9-ab17-bfb36939ef9d/[CHANNEL1] - Voltage (Volt)--Min.sign", "/opt/prod
uctivity_link/esrv_b230cee8-59bc-49e9-ab17-bfb36939ef9d/[CHANNEL1] - Voltage (Vo
lt)--Min.decimals", "/opt/productivity_link/esrv_b230cee8-59bc-49e9-ab17-bfb3693
9ef9d/[CHANNEL1] - Voltage (Volt seconds)", "/opt/productivity_link/esrv_b230cee
8-59bc-49e9-ab17-bfb36939ef9d/[CHANNEL1] - Voltage (Volt seconds).sign", "/opt/p
roductivity_link/esrv_b230cee8-59bc-49e9-ab17-bfb36939ef9d/[CHANNEL1] - Voltage
(Volt seconds).decimals", "/opt/productivity_link/esrv_b230cee8-59bc-49e9-ab17-b
fb36939ef9d/[CHANNEL1] - Power Factor (no unit)", "/opt/productivity_link/esrv_b
230cee8-59bc-49e9-ab17-bfb36939ef9d/[CHANNEL1] - Power Factor (no unit).decimals
", "/opt/productivity_link/esrv_b230cee8-59bc-49e9-ab17-bfb36939ef9d/[CHANNEL1]
- Power Factor (no unit)--Max", "/opt/productivity_link/esrv_b230cee8-59bc-49e9-
ab17-bfb36939ef9d/[CHANNEL1] - Power Factor (no unit)--Max.decimals", "/opt/prod
uctivity_link/esrv_b230cee8-59bc-49e9-ab17-bfb36939ef9d/[CHANNEL1] - Power Facto
```

```
r (no unit)--Min", "/opt/productivity_link/esrv_b230cee8-59bc-49e9-ab17-bfb36939
ef9d/[CHANNEL1] - Power Factor (no unit)--Min.decimals", "/opt/productivity_link
/esrv_b230cee8-59bc-49e9-ab17-bfb36939ef9d/[CHANNEL1] - Power Factor (no unit se
conds)", "/opt/productivity_link/esrv_b230cee8-59bc-49e9-ab17-bfb36939ef9d/[CHAN
NEL1] - Power Factor (no unit seconds).decimals", "/opt/productivity_link/esrv_b
230cee8-59bc-49e9-ab17-bfb36939ef9d/[CHANNEL1] - Current Frequency (Hertz)", "/o
pt/productivity_link/esrv_b230cee8-59bc-49e9-ab17-bfb36939ef9d/[CHANNEL1] - Curr
ent Frequency (Hertz).decimals", "/opt/productivity_link/esrv_b230cee8-59bc-49e9
-ab17-bfb36939ef9d/[CHANNEL1] - Current Frequency (Hertz)--Max", "/opt/productiv
ity_link/esrv_b230cee8-59bc-49e9-ab17-bfb36939ef9d/[CHANNEL1] - Current Frequenc
y (Hertz)--Max.decimals", "/opt/productivity_link/esrv_b230cee8-59bc-49e9-ab17-b
fb36939ef9d/[CHANNEL1] - Current Frequency (Hertz)--Min", "/opt/productivity_lin
k/esrv_b230cee8-59bc-49e9-ab17-bfb36939ef9d/[CHANNEL1] - Current Frequency (Hert
z)--Min.decimals", "/opt/productivity_link/esrv_b230cee8-59bc-49e9-ab17-bfb36939
ef9d/[CHANNEL1] - Current Frequency (Hertz seconds)", "/opt/productivity_link/es
rv_b230cee8-59bc-49e9-ab17-bfb36939ef9d/[CHANNEL1] - Current Frequency (Hertz se
conds).decimals", "/opt/productivity_link/esrv_b230cee8-59bc-49e9-ab17-bfb36939e
f9d/[CHANNEL1] - Voltage Frequency (Hertz)", "/opt/productivity_link/esrv_b230ce
e8-59bc-49e9-ab17-bfb36939ef9d/[CHANNEL1] - Voltage Frequency (Hertz).decimals",
 "/opt/productivity_link/esrv_b230cee8-59bc-49e9-ab17-bfb36939ef9d/[CHANNEL1] -
Voltage Frequency (Hertz)--Max", "/opt/productivity_link/esrv_b230cee8-59bc-49e9
-ab17-bfb36939ef9d/[CHANNEL1] - Voltage Frequency (Hertz)--Max.decimals", "/opt/
productivity_link/esrv_b230cee8-59bc-49e9-ab17-bfb36939ef9d/[CHANNEL1] - Voltage
 Frequency (Hertz)--Min", "/opt/productivity_link/esrv_b230cee8-59bc-49e9-ab17-b
fb36939ef9d/[CHANNEL1] - Voltage Frequency (Hertz)--Min.decimals", "/opt/product
ivity_link/esrv_b230cee8-59bc-49e9-ab17-bfb36939ef9d/[CHANNEL1] - Voltage Freque
ncy (Hertz seconds)", "/opt/productivity_link/esrv_b230cee8-59bc-49e9-ab17-bfb36
939ef9d/[CHANNEL1] - Voltage Frequency (Hertz seconds).decimals", "/opt/producti
vity_link/esrv_b230cee8-59bc-49e9-ab17-bfb36939ef9d/[CHANNEL1] - Channel(s)", "/
opt/productivity_link/esrv_b230cee8-59bc-49e9-ab17-bfb36939ef9d/[CHANNEL1] - Sta
tus", "/opt/productivity_link/esrv_b230cee8-59bc-49e9-ab17-bfb36939ef9d/[CHANNEL
1] - Version", "Sample #"
To Stop Logging  : [<CTRL>+<C>]
Mon Sep 14 13:49:13 2009, 345897, 2, 9, 4, 0, 1, 15009, 2, 15090, 2, 15000, 2, 5
5000, 0, 4, 59000, 0, 4, 50000, 0, 4, 1304995, 0, 4, 11029, 0, 2, 11090, 0, 2, 5
0000, 0, 2, 254092, 0, 2, 1000, 4, 9000, 4, 0, 4, 93996, 4, 501000, 4, 508999, 4
, 500000, 4, 11618990, 4, 500000, 4, 508999, 4, 500000, 4, 11584992, 4, 1, 1, 20
090303, 1
Mon Sep 14 13:49:14 2009, 360956, 2, 10, 4, 0, 1, 15059, 2, 15090, 2, 15000, 2,
54000, 0, 4, 59000, 0, 4, 50000, 0, 4, 1358995, 0, 4, 11020, 0, 2, 11090, 0, 2,
50000, 0, 2, 265112, 0, 2, 9000, 4, 9000, 4, 0, 4, 102996, 4, 503999, 4, 508999,
 4, 500000, 4, 12122989, 4, 502000, 4, 508999, 4, 500000, 4, 12086992, 4, 1, 1,
20090303, 2
Mon Sep 14 13:49:15 2009, 375975, 2, 10, 4, 0, 1, 15019, 2, 15090, 2, 15000, 2,
57000, 0, 4, 59000, 0, 4, 50000, 0, 4, 1415995, 0, 4, 11029, 0, 2, 11090, 0, 2,
50000, 0, 2, 276141, 0, 2, 4000, 4, 9000, 4, 0, 4, 106996, 4, 500000, 4, 508999,
 4, 500000, 4, 12622989, 4, 507000, 4, 508999, 4, 500000, 4, 12593992, 4, 1, 1,
20090303, 3
```

NOTE

In the listing above, the logged data header is printed in bold and the individual counter readings are printed in italics. Notice on line 169 that a message is interleaved between the header line and the subsequent data lines. This is due to the fact that pl_csv_logger is a multithreaded application and no synchronization is applied to stdout. This does not happen if an output file is specified (see section 3.3.1). If stdout is redirected, such un-wanted lines must be removed with awk or some other text filtering utility.

3.4 Stopping the Logger

The logger may be terminated with one of the following keys or key combinations: Escape, <CTRL>+<C>, or <ALT>+<F4> (Windows only). These keys should be pressed after selecting the console logger window as shown in Figure 8. The logger window may also be terminated by closing the window via a mouse.

NOTE

Be careful when using <CTRL>+<C> as other applications may be interrupted too. The other methods for terminating the logger application are generally preferred.

3.5 Counter Processing

By default, the logger logs the PL counters as-is without applying to them any processing. By adding the `--process` option to the command line, user can force the logger to analyze the counters exported in the monitored PL(s) to find dependencies between counters. Dependency detection is confined to each PL. If such dependencies are found, then the logger will use the suffix counters to compute the base counters' real value. Refer to Appendix section in the *Intel Energy Checker SDK User Guide* for details and recommended usage of suffixes, or use the `--process --help` command line option to display a short explanation of counters and suffix counters. Note that the processing option applies to all monitored PL(s).

If counter processing is active, then on each data line, suffix counter's values are replaced by a star (*) character as shown in the example below.

```
Mon Sep 14 14:06:43 2009, +160810.95, *, +0.0446, *, 0, 1, +150.09, *, +150.90,
*, +150.00, *, +5.5000, *, *, +5.9000, *, *, +5.0000, *, *, +5843.6554, *, *,
+110.70, *, *, +110.90, *, *, +500.00, *, *, +118070.32, *, *, +0.5999, *,
+0.9000, *, +0.0000, *, +491.5683, *, +50.2999, *, +50.8999, *, +50.0000, *,
+53940.1562, *, +50.8999, *, +50.8999, *, +50.0000, *, +53931.5591, *, 1, 1,
20090303, 274
```

NOTE

If there are more than 250 counters in all the PLs accessed by a given logger when counter processing is requested (`--process` option), an error message will be displayed (cannot associate counters). If multiple PLs were selected on a single command line, running separate logger instances for each PL may eliminate the issue.

3.6 Sampling Frequency

By default, the logger samples the counters at 1 Hz frequency. To lower this frequency, use the `--t <second(s)>` option on the command line to specify the pause time to be used by the logger between each sample collection. Pause time is expressed in seconds. The minimal value for the pause time is the default value, which is one (1) second. There is no upper limit for the pause time.

3.7 Data Analysis

The header line in the CSV file is used to name columns in a spreadsheet. The first column (`"Time Stamp"`) provides a time stamp for each entry and the last column (`"Sample #"`) provides an incrementing sequence number for each entry.

Figure 28 shows logger data pulled into Microsoft Excel* and Figure 29 shows logger data graphed in Excel.

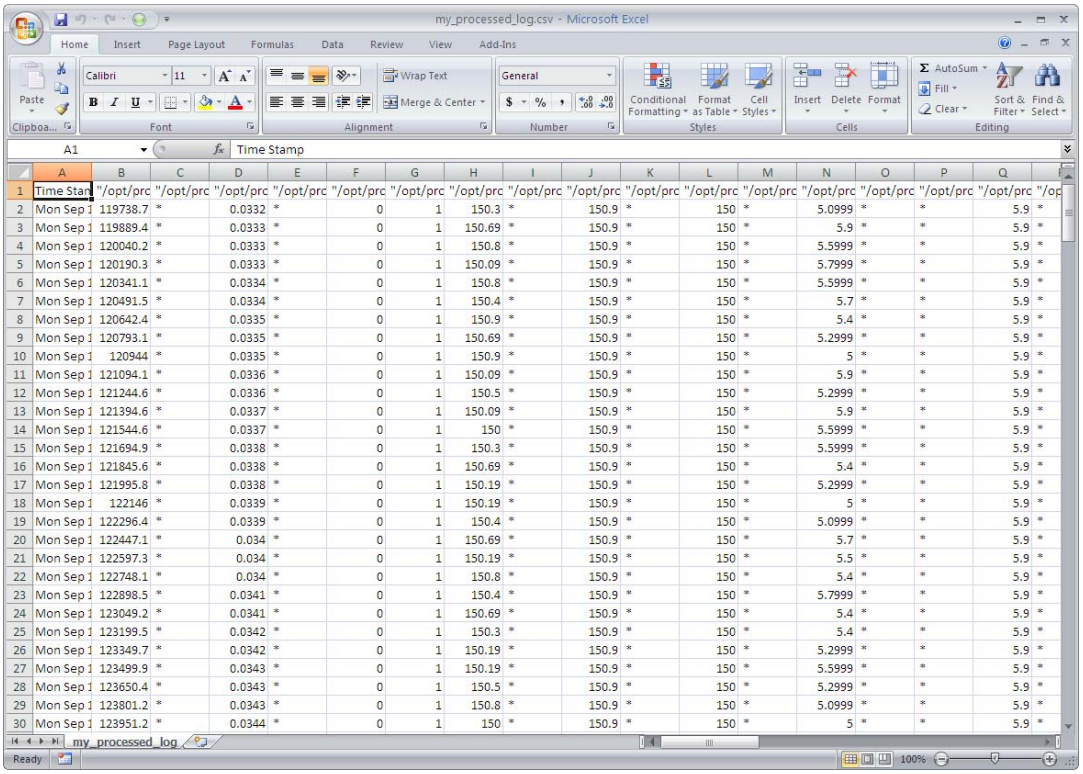

Figure 28: Example processed log data imported into Microsoft Excel

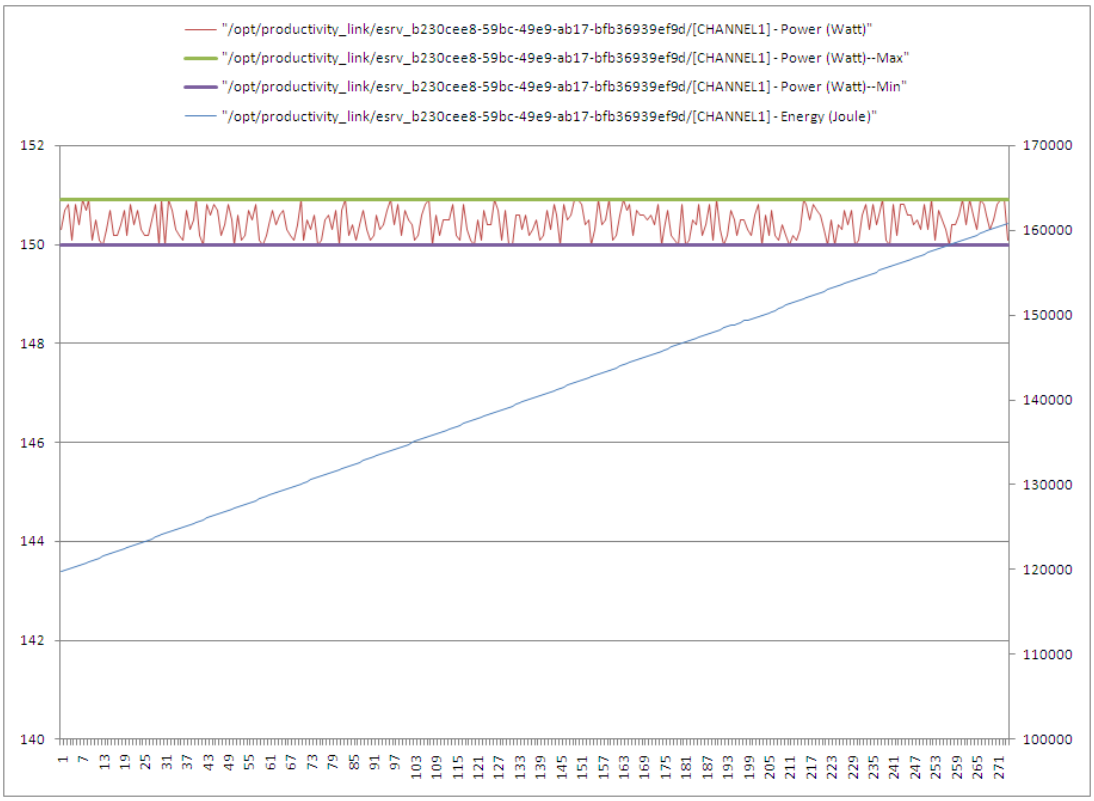

Figure 29: Example partial processed log data charted in Microsoft Excel

3.8 Example Logger Command Lines and Help

3.8.1 Example Logger Command Line Invocations

```
./pl_csv_logger /opt/productivity_link/esrv_b230cee8-59bc-49e9-ab17-
bfb36939ef9d/pl_config.ini --output my_processed_log.csv --process
pl_gui_monitor c:\productivity_link\PovRay_76b5e071-34be-47fc-8f26-
931157d58f5e\pl_config.ini --process
```

3.8.2 Logger Syntax and Help

Use the `--help` option to display basic help on the logger's syntax. The listing below was obtained using this option.

```

Start PL CSV logger and output data in comma-separated variable format that can
be loaded into spreadsheets or other analysis tools.

Usage:  pl_csv_logger [[PL_ini] ... ] [--t <n>] [--output <filename>] [--process]
Or      pl_csv_logger [--version]
Or      pl_csv_logger [--help]
```

```
   [PL_ini] is one or more fully qualified path(s) to PL configuration file(s).
   --t with [n] sets the pause time between each log entry creation.
      n is expressed in seconds.  n is equal to 1 by default.
   --output with [filename] writes output to the named file rather than stdout.
   --process forces additional analysis on counter values with prescribed suffixes.
      Use pl_csv_logger --process --help for more information on the suffixes.
   --version prints version information.
   --help prints this help message.

The logger application continues to append data to the file once per second (or
whatever time interval is selected with the --t option).

Use the --process --help option to display basic help on the logger's counter
processing mechanism.  The listing below was obtained using this option.

Additional help information on the counter suffixes supported by the CSV logger.

   The following suffixes are recognized by csv logger when the --process
   parameter is supplied:

      .decimals (defaults to 0 if suffix counter doesn't exist)
      .scalar (defaults to 1 if suffix counter doesn't exist)
      .scalar.decimals (defaults to 0 if suffix counter doesn't exist)
      .sign (defaults to 0 if suffix counter doesn't exist)
      .offset (defaults to 0 if suffix counter doesn't exist)
      .offset.decimals (defaults to 0 if suffix counter doesn't exist)
      .offset.sign (defaults to 0 if suffix counter doesn't exist)

   The following formula is used to compute a "total" counters' real values:

      real_total = total
                 / (10 ^ total.decimals)
                 * total.scalar
                 / (10 ^ total.scalar.decimals)
                 * (total.sign ? -1 : 1)
                 + total.offset
                 / (10 ^ total.offset.decimals)
                 * (total.offset.sign ? -1 : 1)
      ;

Sign Suffix
-----------
      For maximum portability across different architectures,
   the Intel EC SDK always uses positive values for its counters,
   but negative values can easily be represented with PL counters
   and interpreted by other applications if the following
   convention is adhered to:
      1.  Add a supplemental counter with a ".sign" suffix
```

```
      2.  Write the static sign to the supplemental counter
          using the following convention:
              1 means a negative number.
              0 means a positive number (or zero).
   If the sign suffix is omitted, the number is assumed to be
   positive.

Decimals Suffix
---------------
      The EC SDK always uses unsigned long long values for its counters,
   but floating point values with a fixed number of decimal places can
   easily be represented with PL counters and interpreted by other
   applications if the following convention is adhered to:
      1.  Add a supplemental counter with a ".decimals" suffix
      2.  Write the static number of decimal places to the supplemental
          counter
      3.  Adjust counters by 10 ^ n before writing to PL

   For example, the ESRV utility makes use of this with its
   "Energy (kWh)" counter.  To represent the value to two decimal
   places, the actual number of kiloWatt-hours is multiplied by 100 and
   stored as an unsigned long long.  In addition, at program startup,
   ESRV writes a value of 2 to the "Energy (kWh).decimals" counter to
   indicate that "Energy (kWh)" should be interpreted as having two
   decimal places.

   All applications writing counters representing fixed decimal numbers
   should use the .decimals suffix on a supplemental static counter as
   appropriate.  The number of decimal places should not be changed
   after it is written.

Offset Suffix
-------------
      In some cases, the native source of a counter may represent a
   differential value from some fixed offset value.  Rather than
   constantly adding the differential value to the offset value each
   time the counter is written out, an application developer could
   elect to define a static offset counter and let any consumers of the
   PL data add in the offset where needed.  To do this, the following
   convention has been established:
      1.  Add a supplemental counter with a ".offset" suffix
      2.  Write the static value of the offset to the supplemental
          counter
   For example, if an application is counting the number of visitors
   over 5000 to a theme park, a "Visitor Total" value of 800 might
   represent a total attendance of 5800 visitors.  By adding a
   "Visitor Total.offset" counter and writing a value of 5000 to that
   offset counter, the application developer can indicate what number
   to add to the counter value to get the real value.
```

```
      The Offset suffix may have additional suffixes.  Currently, the
   .sign and .decimal suffixes are supported by this logger.

Scalar Suffix
-------------
      In most cases, the decimals suffix provides sufficient ability
   to scale the way numbers are reported out.  However, there are some
   cases where the counter values written through PL need some sort of
   scaling factor applied to them.  In such cases, the following
   convention should be used:
      1.  Add a supplemental counter with a ".scalar" suffix
      2.  Write the static value of the scaling factor to the
          supplemental counter
   For example, software for an egg producer could report how many
   dozens of eggs were produced with an "Egg Dozens" counter.
   That same value could be written to an "Eggs" counter, provided
   that "Eggs.scalar" had been set up with a value of twelve.
   When units are measured by the thousands or millions, using a
   scalar factor of 1000 or 1,000,000 as appropriate will represent
   the proper scale of the units.

   The scalar suffix also supports the .decimals suffix (i.e.,
   .scalar.decimals) to allow scalar values to represent values
   Other than integers.


NOTE
    The static counters can be used in a compound manner.
    For example, consider the following:
       MyTotal counter has a value of 5
       MyTotal.decimals has a value of 1
       MyTotal.scalar has a value of 72

   In this example, the actual value is 36:
       5 / (10 ^ 1) * 72 = 36
```

3.9 CSV Logger as Sample Code

In addition to its value as a useful utility, the CSV logger also serves as sample code to demonstrate the use of the Intel® EC API. In particular, this sample was created to show how an application should handle suffix counters. Please refer to the Intel® EC SDK user guide for introductory information on suffix counters.

The `pl_csv_logger.c` file is the main file of the sample. If the `--process` option is specified on the command line, then the application starts first by analyzing the PLs to log to see whether they have corresponding suffix counters (see lines 216 to 471). The logger looks for and interprets specific suffix counters as described in the Intel® Energy Checker SDK User Guide.

If a relationship is found between a counter and corresponding suffix counters, then an *ad hoc* structure (`PL_CSV_LOGGER_COUNTER_DATA`, defined in `pl_csv_logger.h`) is updated to store this information for a further use.

```
 99        typedef struct _pl_csv_logger_counter_data {
100
101           int processed;
102           char name[MAX_PATH];
103           unsigned int i_decimals;
104           unsigned int i_offset;
105           unsigned int i_offset_decimals;
106           unsigned int i_offset_sign;
107           unsigned int i_scalar;
108           unsigned int i_scalar_decimals;
109           unsigned int i_sign;
110
111       } PL_CSV_LOGGER_COUNTER_DATA, *PPL_CSV_LOGGER_COUNTER_DATA;
```

Note that for each counter, the corresponding suffix variables are initialized in `pl_csv_logger.c` to values that have no effect on the processed value if no corresponding suffix counter is defined.

```
602           counter = v;
603           counter_scalar = 1;
604           counter_scalar_decimals = 0;
605           counter_decimals = 0;
606           counter_sign = 0;
607           counter_offset = 0;
608           counter_offset_decimals = 0;
609           counter_offset_sign = 0;
```

Once the analysis phase is completed, the logger periodically reads each counter's value with any linked suffix counters. Although the `.sign` suffix is the only suffix counter that should change dynamically, the logger periodically re-reads the suffix counters in case they have changed:

```
620         if(p_counters_data[i][j].processed & DECIMALS_PROCESSING) {
621            ret = pl_read(plds[i], &counter_decimals, p_counters_data[i][j].i_decimals);
622            if(ret == PL_FAILURE) {
623               f_log_error = 1;
624               goto main_compute_processed_counter_value;
625             }
626         }
```

Once a counter's value and its suffix counters' values are known, the real counter value is computed:

```
743            //-------------------------------------------------------------
744            // compute counter real value
745            //-------------------------------------------------------------
746            processed_value =
747               (long double)counter
748               / (long double)pow(10.0, (double)counter_decimals)
749               * (long double)counter_scalar
750               / (long double)pow(10.0, (double)counter_scalar_decimals)
751               * (counter_sign ? -1.0 : 1.0)
752               + (long double)counter_offset
753               / (long double)pow(10.0, (double)counter_offset_decimals)
754               * (counter_offset_sign ? -1.0 : 1.0)
755            ;
```

4 Appendix A: Cross-platform Analysis

Most Productivity Link (PL) operation occurs with PLs on the local machine. However, both the monitor and logger applications allow monitoring or logging PLs on remote machines.

Both applications take fully qualified filenames to PL configuration files as command-line arguments. These fully qualified filenames may be located on the local system or on one or more remote systems (accessible via standard drive mapping or file system mounting mechanisms).

Since all the PL counters are stored as unsigned 64-bit values, there are no issues with cross-platform compatibility. This means that a Windows* system can monitor PLs on another Windows system, a Solaris* system, a Macintosh* system, or a Linux* system—regardless of processor architecture or whether the system is a 32-bit or 64-bit OS.

Refer to the Intel® Energy Checker SDK User Guide for an example network setup supporting cross-platform analysis.