Using the Power Management Features of the Xen Kernel on Lenovo ThinkSystem Servers

Introduces Xen power management features

Provides the basic power management usage in the Xen kernel

Shows how to set the power mode in UEFI on ThinkSystem servers

Demonstrates the simple usage in CPU C/P state

Conie Chang

LENOVO PRESS

Abstract

This paper provides an overview of the power management features of the Xen Kernel as used in SUSE Linux Enterprise Server. provides the basic power management usage, including CPU P-state and CPU C-state.

This paper is intended for IT specialists and IT managers who are familiar with power management and Linux Xen OS.

At Lenovo® Press, we bring together experts to produce technical publications around topics of importance to you, providing information and best practices for using Lenovo products and solutions to solve IT challenges.

See a list of our most recent publications at the Lenovo Press web site:

http://lenovopress.com

Do you have the latest version? We update our papers from time to time, so check whether you have the latest version of this document by clicking the **Check for Updates** button on the front page of the PDF. Pressing this button will take you to a web page that will tell you if you are reading the latest version of the document and give you a link to the latest if needed. While you're there, you can also sign up to get notified via email whenever we make an update.

Contents

Introduction

Power management features are standard in most modern computer systems and commodity operating systems. These features are also important to virtualization kernels such as the Xen kernel OS.

In this paper, we describe the fundamentals of power management as it applies to the Xen hypervisor, and we describe how to enable the use of CPU P-State and C-State controls in the Xen kernel.

Basic components

For the basic components, the simple logic is shown in Figure 1. It includes Xen hypervisor, dom0 and xenpm tool.

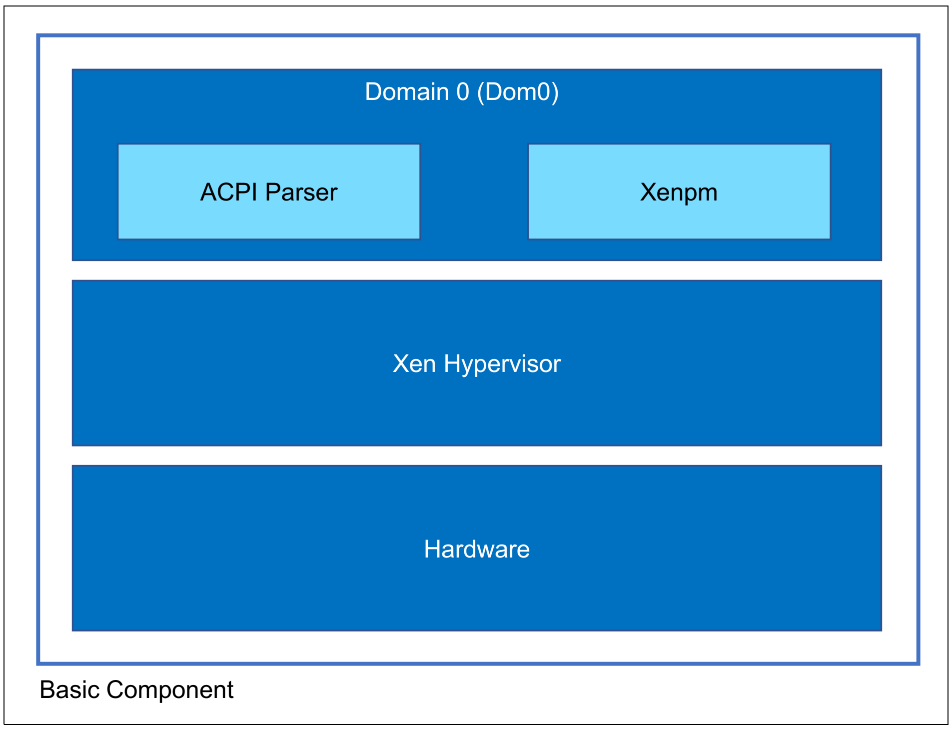

Figure 1 Basic component logic

These components are as follows:

- Xen hypervisor

  The Xen hypervisor, sometimes called a virtual machine monitor, is an open source software program that coordinates the low-level interactions between virtual machines and physical hardware.

- Dom0 (Domain0)

  The virtual machine host environment, also called Dom0 or controlling domain, is composed of several components, such as SUSE Linux Enterprise Server, which provides a graphical and command-line environment to manage the virtual machine host components and their virtual machines.

Dom0 also provides the following useful programs:

– The xl tool stack based on the xenlight library (libxl). Use it to manage Xen guest domains.

– QEMU is an open source software that emulates a full computer system, including a processor and various peripherals. It provides the ability to host operating systems in both full virtualization and paravirtualization mode.

- xenpm - userspace control tool

  Xenpm is a userspace tool that can list the power information of available processors and control the power policy according to users' preferences. Its name stands for Xen Power Management (xenpm for short).

  For the rest of xenpm command, please also refer to the following URL.

  https://wiki.xenproject.org/wiki/Xenpm_command

- ACPI Parser - domain0 kernel ACPI sub-system. It parses ACPI table and pass the information to hypervisor by hypercall

CPU P-States (cpufreq) with Xen

CPU P state (performance state) is one kind of processor power saving state defined in the ACPI specification. CPU P state saves power by changing CPU frequency and voltage. Among the P state (P0, P1... Pn), P0 has the highest frequency and thus the highest power consumption. Pn has the lowest frequency and thus the lowest power consumption. Enhanced Intel SpeedStep Technology (EIST) and AMD PowerNow! Technology are two examples of P-State implementations.

The Xen kernel supports P-State using the cpufreq driver, which is the same as non-Xen Linux OS such as SLES. The logic is similar to the commodity OSes:

1. Periodically measure system status such as CPU utilization
2. Determine the appropriate CPU frequency according to the cpufreq policy and current system status,
3. Issue platform-dependent command to change the CPU frequency.

The logic is shown as Figure 2 on page 5.

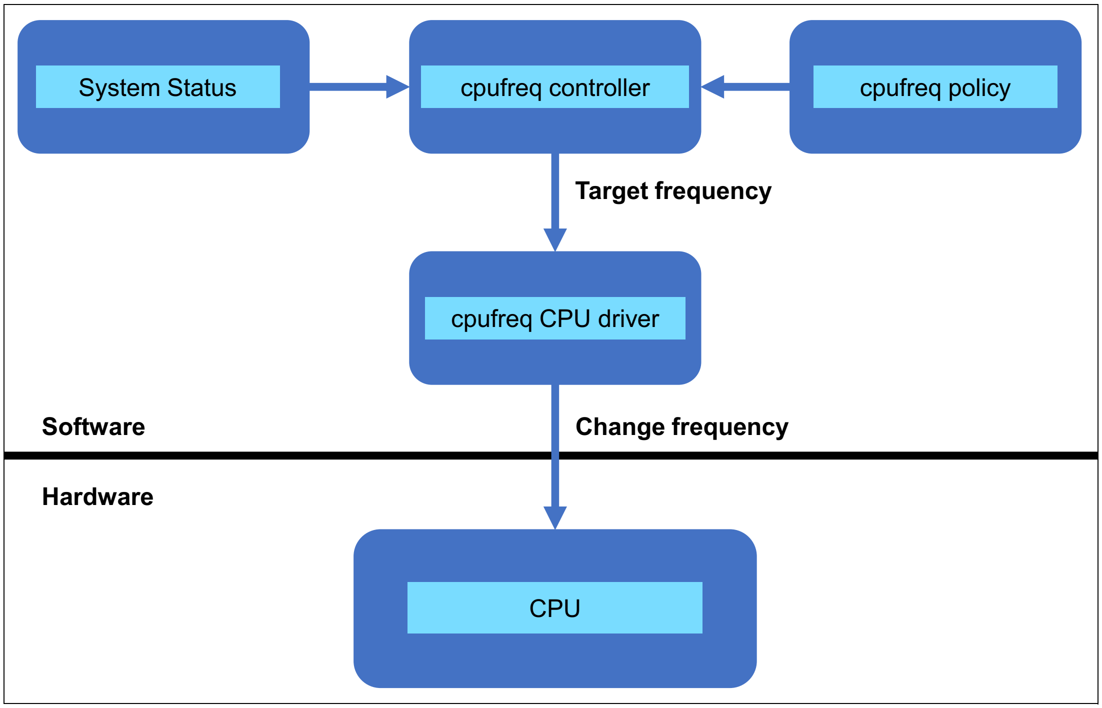

Figure 2 cpufreq

For historical reason, Xen has two implementations:

- Domain0-based cpufreq, which implements the cpufreq logic in domain0
- Hypervisor-based cpufreq, which implements the cpufreq logic in hypervisor.

We describe both of these in the following sections. The default is the hypervisor based cpufreq.

Domain0 based cpufreq

Domain0 based cpufreq reuse the domain0 kernel cpufreq code and let domain0 handle the cpufreq logic.

Xen hypervisor provides two hypercalls to assist the domain0 kernel to get the system status and change the CPU frequency

- XENPF_change_freq, as shown in Figure 3 on page 6
- XENPF_getidletime, as shown in Figure 4 on page 6

These hypercalls are listed in platform.h as show source code on this web page:

https://xenbits.xen.org/people/ianc/docs/platform_8h_source.html

```
#define XENPF_change_freq        52
struct xenpf_change_freq {
    /* IN variables */
    uint32_t flags; /* Must be zero. */
    uint32_t cpu;   /* Physical cpu. */
    uint64_t freq;  /* New frequency (Hz). */
};
```

Figure 3 XENPF_change_freq

```
#define XENPF_getidletime        53
struct xenpf_getidletime {
    /* IN/OUT variables */
    /* IN: CPUs to interrogate; OUT: subset of IN which are present */
    XEN_GUEST_HANDLE(uint8) cpumap_bitmap;
    /* IN variables */
    /* Size of cpumap bitmap. */
    uint32_t cpumap_nr_cpus;
    /* Must be indexable for every cpu in cpumap_bitmap. */
    XEN_GUEST_HANDLE(uint64) idletime;
    /* OUT variables */
    /* System time when the idletime snapshots were taken. */
    uint64_t now;
};
```

Figure 4 XENPF_getidletime

For the Dom0 boot parameter, `cpufreq=dom0-kernel` is deprecated and not supported by all Dom0 kernels. It is not necessary to add `cpufreq=dom0-kernel` when using the dom0 cpufreq feature. For more information, see the following URL:

https://xenbits.xen.org/docs/4.10-testing/misc/xen-command-line.html

Hypervisor based cpufreq

Hypervisor based cpufreq implement most of the cpufreq logic in hypervisor. Figure 5 on page 7 illustrates the logic.

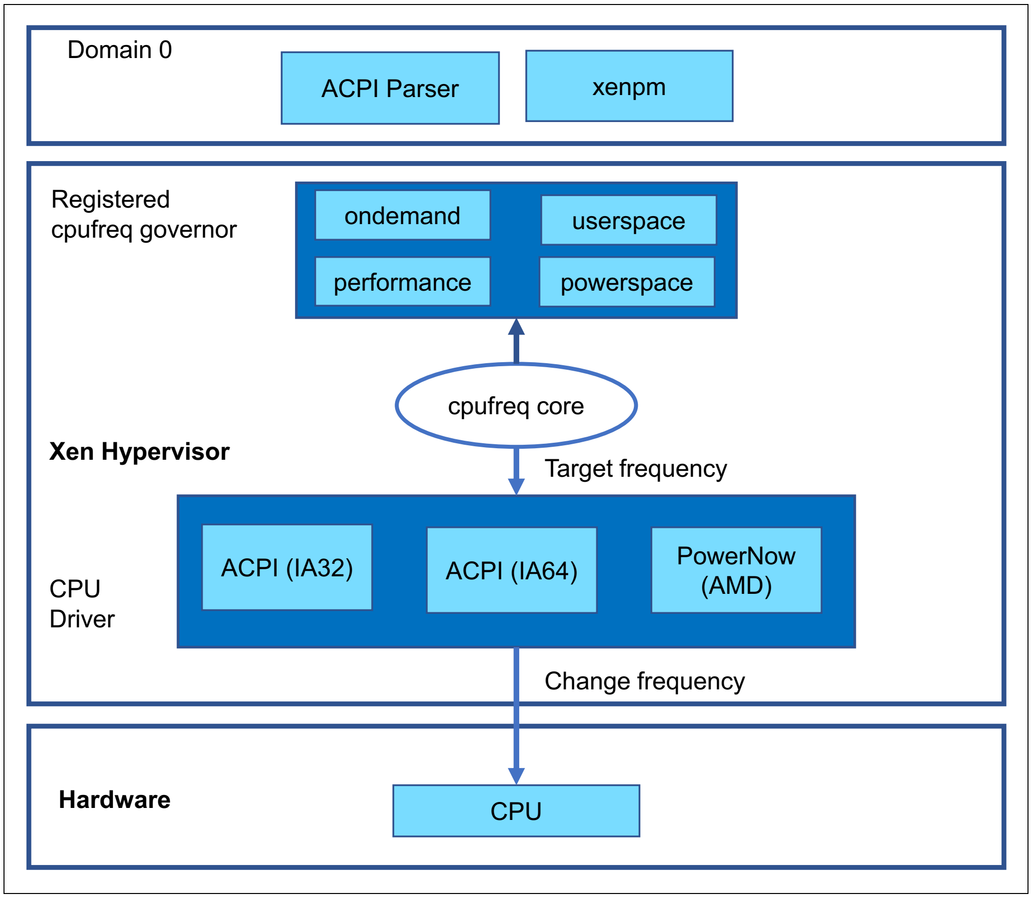

Figure 5 Hypervisor based cpufreq

The Hypervisor has three components as shown in Figure 5:

- cpufreq core - controls the overall logic
- cpufreq governor - controls the cpufreq policy, i.e. which CPU frequency should CPU goes to
- CPU driver - includes two drivers:
  - Cpufreq CPU driver issues command to change physical CPU frequency
  - xen_acpi_processor driver parses the Power Management data and uploads said information to the Xen hypervisor

For xen_acpi_processor driver information, see the following web page:

https://cateee.net/lkddb/web-lkddb/XEN_ACPI_PROCESSOR.html

Xen currently has four governors:

- ondemand: Choose the best frequency which best fit the current system status.
- userspace (default): Choose the frequency that specified by user.
- performance: Select the highest frequency
- powersave: Select the lowest frequency

Xen also supports three CPU drivers:

- ACPI (IA32) for Intel x86 processors
- ACPI (IA64) for Intel Itanium processors
- PowerNow K8 for AMD processors

Domain0 has two components as shown in Figure 5:

- ACPI parser, which parses ACPI table and pass the information to hypervisor cpufreq core
- xenpm tool, a userspace cpufreq control tool, which can select cpufreq governor, specify userspace governor frequency, etc.

We describe the use of xenpm in the next section, “Enabling the use of CPU P-States in Xen” on page 8

Usage: user can use Xen boot option `cpufreq=xen` to enable hypervisor-based cpufreq. From c/s 18950, it is enabled by default with userspace governor, so no Xen boot option is needed after c/s 18950.

Enabling the use of CPU P-States in Xen

In this section, we describe how to enable the use of P-State controls in Xen.

For our testing we used the following system:

- Lenovo ThinkEdge SE450
- SLES15SP4 with Xen (4.16.0_04-150400.2)

The steps to enable the CPU P-state feature are as follows:

1. Confirm that the next boot will boot into Xen kernel, Figure 6.

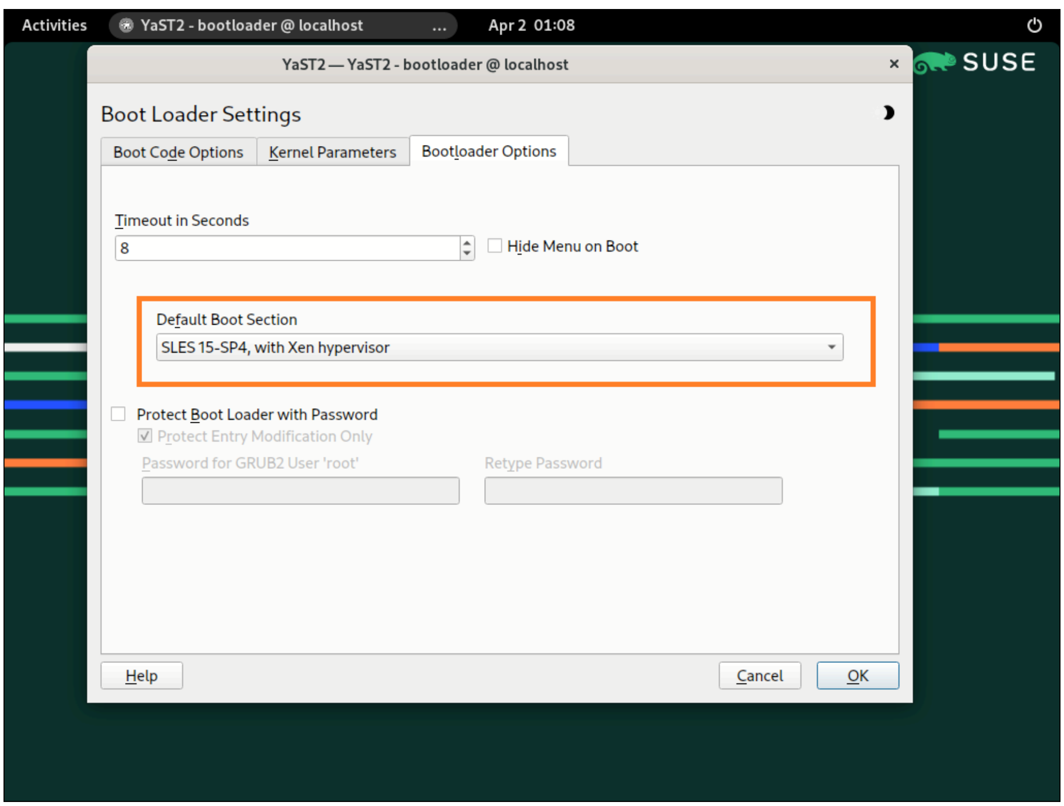

Figure 6 Confirm the default boot section in the SLES 15SP4 bootloader application

2. Reboot system and boot into Xen kernel

With the default UEFI settings, UEFI P-state is set as Autonomous. Autonomous mode means the P-States are fully controlled by system hardware and P-State feature is not presented to the OS. Since in steps 3 to 6, some unexpected output will be observed after loading xen_acpi_processor module with default UEFI setting.

To enable P-state feature, we need to modify UEFI P-State mode to Legacy before booting into Xen kernel. For the P-state enable step, go directly to step 7.

3. Verify that the Xen kernel is running

```
# dmesg | grep Xen
```

4. Check Xen boot message

```
# xl dmesg
```

5. Check whether Xen power management module is loaded

```
# lsmod |grep -i xen_acpi_processor
```

6. If the UEFI setting does not set the P-State legacy mode which P-State feature is presented to the OS, it will show the message in Figure 7 while loading the xen_acpi_processor module.

```
# modprobe xen_acpi_processor
modprobe: ERROR: count not insert 'xen_acpi_processor': No such device
```

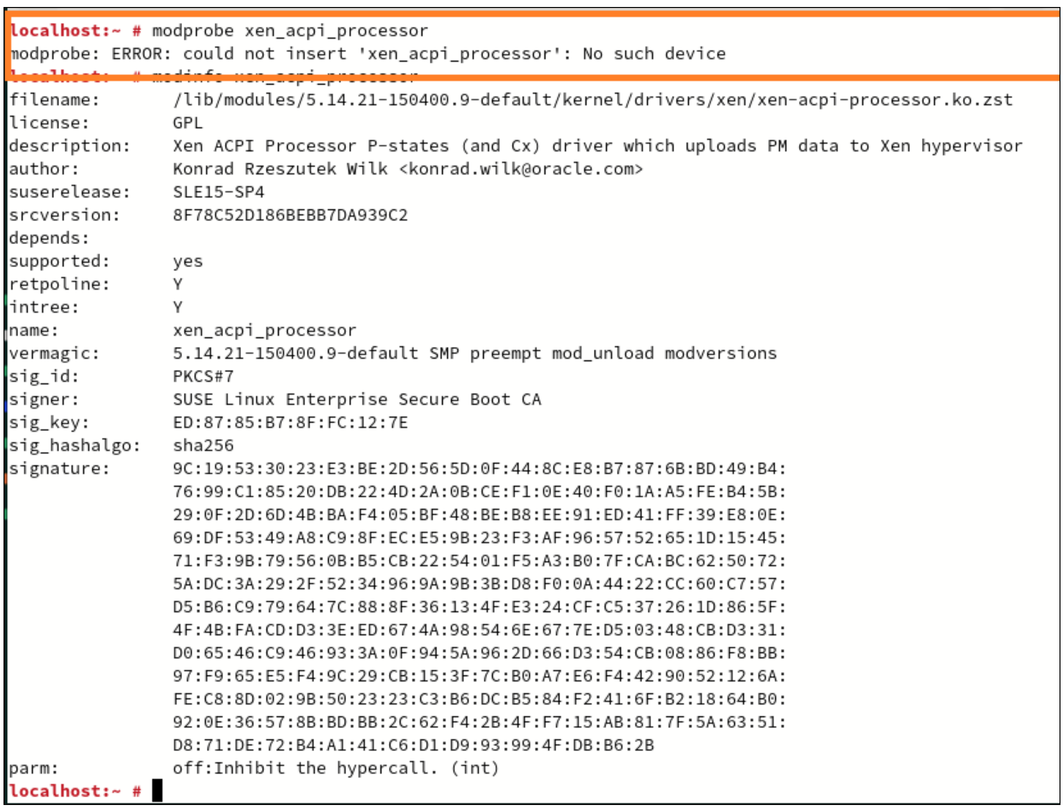

Figure 7 Output of insert xen_acpi_processor with CPU P-State mode [Autonomous]

7. Reboot the system and set the P-State [Legacy] which means that the P-State is presented to operating system as shown in Figure 8 on page 10.

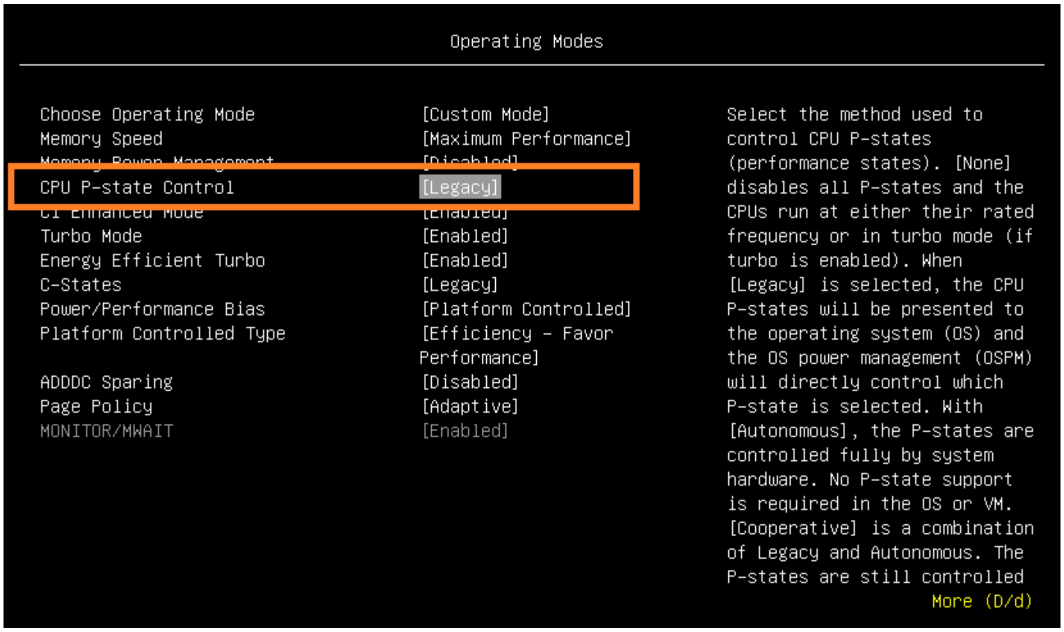

Figure 8 Setting CPU P-State in UEFI

Lenovo UEFI supports multiple P-State control mechanisms. In the UEFI setting description, the CPU P-States Legacy mode will be presented to the OS and the OS power management (OSPM) will directly control which P-State is selected. We are using Legacy mode. Autonomous mode, the P-States are controlled fully by system hardware. No P-State support is required in the OS or VM. This mode is the default setting.

8. After rebooting into Xen, it is necessary to load the xen_acpi_processor module manually.

```
# modprobe xen_acpi_processor
```

9. Check if xen_acpi_processor module is loaded as shown in Figure 9 on page 10.

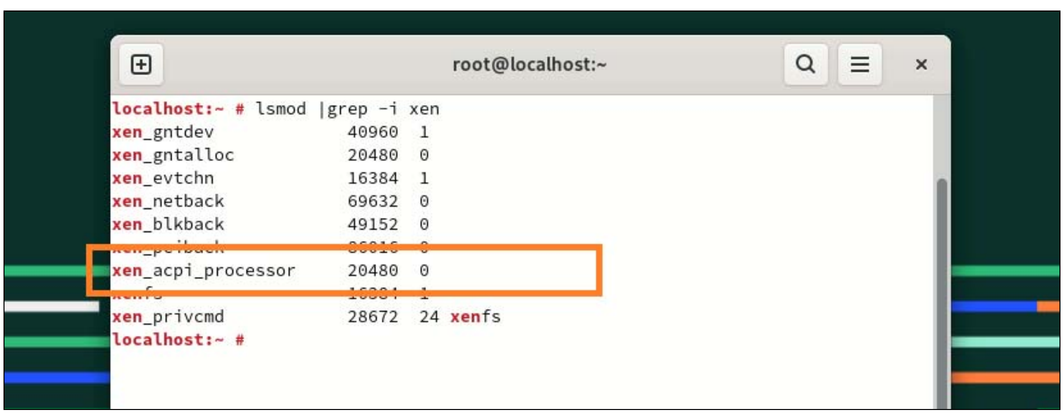

Figure 9 Listing the current Xen-related modules

10. Check the xen_acpi_processor related messages as shown in Figure 10 on page 11.

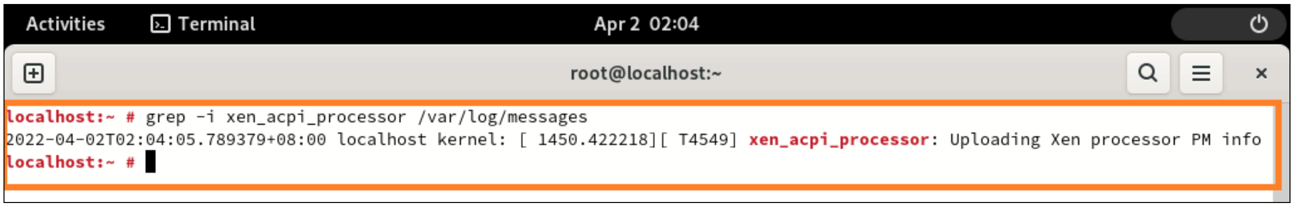

Figure 10   OS message after inserting xen_acpi_xen processor module

11. Get all CPU freq parameters.

    ```
    # xenpm get-cpufreq-para [cpuid]
    # xenpm get-cpufreq-para 0
    ```

    Output is shown in Figure 11.

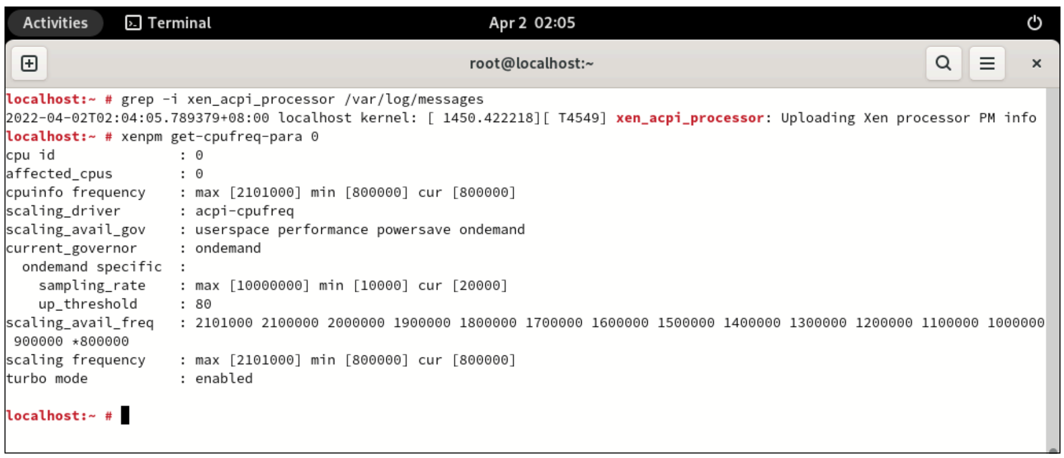

Figure 11   cpu 0 cpufreq parameter output.

12. Set cpufreq governor.

    ```
    # xenpm set-scaling-governor ondemand|performance|powersave
    ```

13. Get cpufreq P-State status. It shows P-State current status as shown in Figure 12 on page 12.

    ```
    # xenpm get-cpufreq-states 0
    ```

```
localhost:~ # xenpm get-cpufreq-state 0
cpu id               : 0
total P-states       : 15
usable P-states      : 15
current frequency    : 800 MHz
P0         [2101 MHz]: transition [                 10]
                       residency  [                235 ms]
P1         [2100 MHz]: transition [                  1]
                       residency  [                 18 ms]
P2         [2000 MHz]: transition [                  0]
                       residency  [                  0 ms]
P3         [1900 MHz]: transition [                  0]
                       residency  [                  0 ms]
P4         [1800 MHz]: transition [                  0]
                       residency  [                  0 ms]
P5         [1700 MHz]: transition [                  0]
                       residency  [                  0 ms]
P6         [1600 MHz]: transition [                  0]
                       residency  [                  0 ms]
P7         [1500 MHz]: transition [                  0]
                       residency  [                  0 ms]
P8         [1400 MHz]: transition [                  0]
                       residency  [                  0 ms]
P9         [1300 MHz]: transition [                  1]
                       residency  [                  0 ms]
P10        [1200 MHz]: transition [                  0]
                       residency  [                  0 ms]
P11        [1100 MHz]: transition [                  1]
                       residency  [                  0 ms]
P12        [1000 MHz]: transition [                  2]
                       residency  [                  1 ms]
P13        [ 900 MHz]: transition [                  0]
                       residency  [                  0 ms]
*P14       [ 800 MHz]: transition [                 10]
                       residency  [                978 ms]

localhost:~ #
```

Figure 12 CPU 0 current cpufreq P state status

CPU C-States (cpuidle)

ACPI defines processor power-saving states as C-States, which include C0, C1, C2, C3, and so on. C0 is the normal working state whereby the CPU executes instructions. C1 ~ Cn are progressively higher power-saving states where CPU stops executing instructions and powers down some internal components. The higher the C state, the greater the power savings, and also the greater the wakeup latency. A processor in a C1~Cn state will be woken up by an event (e.g. interrupt) and transition back to C0.

There are several ways to make the processor enter a C state:

- HLT instruction will make processor enter C1 state.
- Reading some I/O port can make the processor enter a different C state. The I/O port number is platform specific and can be retrieved from ACPI table.
- The platform can also provide specific instruction to enter a C state. For example, in Intel processors, the monitor/mwait instruction pair can be used to enter deeper C state.

cpuidle in Xen

Xen also supports CPU C-States. The logic can be explained as follows:

- When to enter C state: When one physical CPU has no task (VCPU) assigned, it will run idle vcpu, which in turn will put CPU into C-State. When there is breaking event (e.g interrupt) happen, the CPU will be brought out from C state and back to work.
- Which C state to enter: This logic is more complicated: Deeper C state has more power saving, but also more latency. A good algorithm should balance both power saving and performance. Xen uses menu governor to select the deepest C state which also satisfies the latency requirement in the meantime.

Enabling the use of CPU C-States in Xen

In this section, we describe how to enable the use of C-State controls in Xen.

For our testing we used the following system:

- Lenovo ThinkEdge SE450
- SLES15SP4 with Xen (4.16.0_04-150400.2)

The steps to enable C-State in Xen are as follows:

1. Confirm whether the system hardware supports C-State.

   If the C-State is used in dom0/Xen kernel, it is necessary to set C-State as [legacy] to have this feature presented in Xen kernel. This UEFI setting is shown in Figure 13.

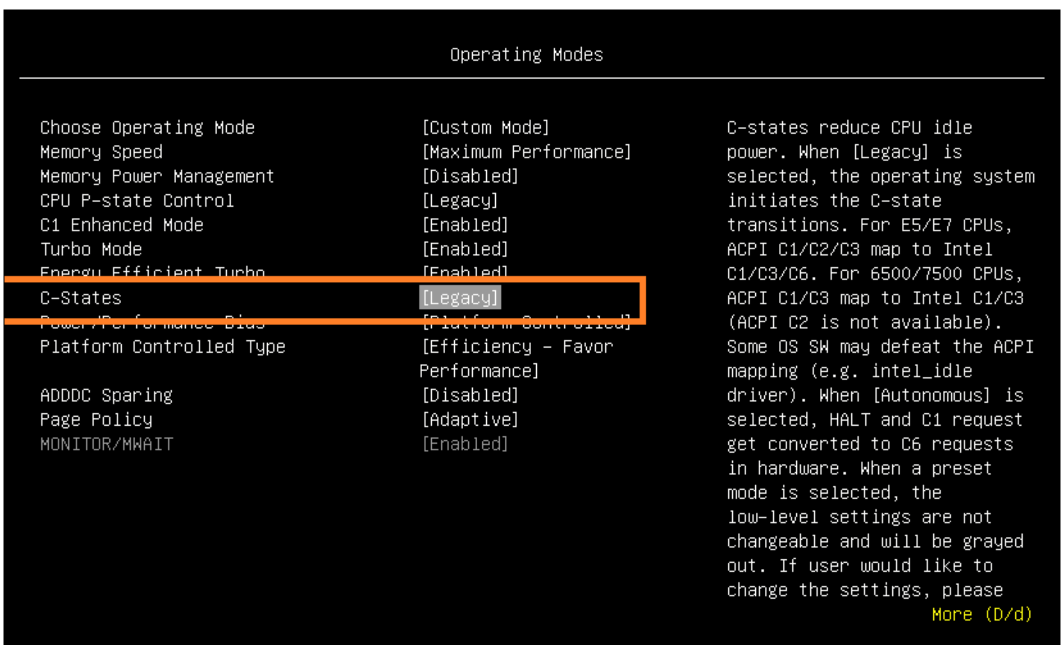

Figure 13 Enable CPU-state as legacy mode

Lenovo UEFI supports multiple C-State control mechanisms. In the UEFI setting description, when the C-States legacy mode is selected, the OS initiates the C-State transitions. ACPI C1/C2/C3 map to Intel C1/C3/C6. Since legacy mode is suitable for the current purpose.

2. Load the xen_acpi_processor in Domain0, as described in “Hypervisor based cpufreq” on page 6.

```
# modprobe xen_acpi_processor
```

3. Determine the Xen version. If the Xen version is earlier than Xen Version 4.4, you will need to add Xen boot parameter, `cpuidle`. For Xen versions 4.4 or later, the parameter is not needed.

   In our lab testing, the Xen kernel version in SLES 15 SP4 is later than V4.4, so the parameter is not needed in the SUSE yast2 bootloader, as shown in Figure 14.

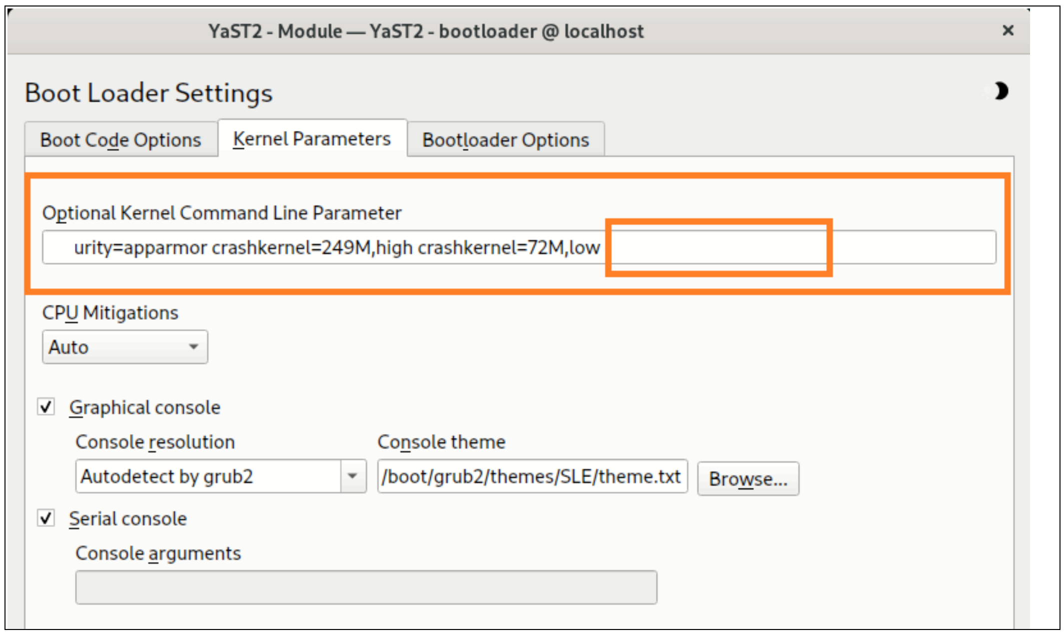

Figure 14 cpuidle parameter not needed for SLES 15 SP4

For older operating systems where the Xen version is older than 4.4, then you will need to add the cpuidle parameter as shown in Figure 15 on page 14.

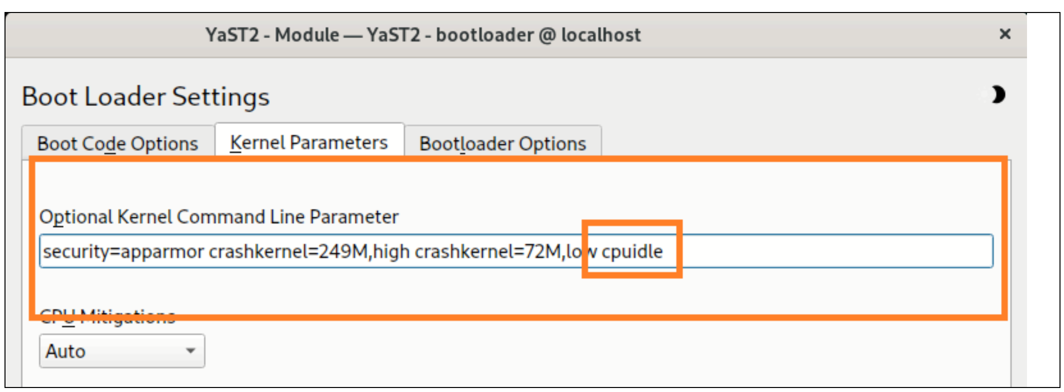

Figure 15 Add the cpuidle parameter for old Xen versions

4. In the SLES 15 SP4 OS, open terminal and issue the following command to show the current C-State status. As shown in the red line, C0 is not enabled.

```
localhost:~ # xenpm start &  sleep 5; killall -s SIGINT xenpm
[1] 3143
Start sampling, waiting for CTRL-C or SIGINT or SIGALARM signal ...
Elapsed time (ms): 5053

CPU0:   Residency(ms)           Avg Res(ms)
  C0    1        ( 0.03%)       0.02
  C1    0        ( 0.00%)       0.00
  C2    489      ( 9.69%)       6.80
  C3    4562     (90.28%)       304.17

  P0    0        ( 0.00%)
  P1    0        ( 0.00%)
  P2    0        ( 0.00%)
  P3    0        ( 0.00%)
  P4    0        ( 0.00%)
  P5    0        ( 0.00%)
  P6    0        ( 0.00%)
  P7    0        ( 0.00%)
  P8    0        ( 0.00%)
  P9    0        ( 0.00%)
  P10   0        ( 0.00%)
  P11   0        ( 0.00%)
  P12   0        ( 0.00%)
  P13   0        ( 0.00%)
  P14   0        (100.00%)
  Avg freq       798380   KHz
```

Figure 16   Checking C-State

5. Set C-State as C0.

```
# localhost:~ # xenpm set-max-cstate 0
```

6. Check the C-State status again using the following commands.

```
localhost:~ # xenpm start &  sleep 5; killall -s SIGINT xenpm
Start sampling, waiting for CTRL-C or SIGINT or SIGALARM signal ...
Elapsed time (ms): 5049

CPU0:   Residency(ms)           Avg Res(ms)
  C0    5049    (100.00%)       5049.06
  C1    0       ( 0.00%)        0.00
  C2    0       ( 0.00%)        0.00
  C3    0       ( 0.00%)        0.00

  P0    0       ( 0.00%)
  P1    0       ( 0.00%)
  P2    0       ( 0.00%)
  P3    0       ( 0.00%)
  P4    0       ( 0.00%)
  P5    0       ( 0.00%)
  P6    0       ( 0.00%)
  P7    0       ( 0.00%)
  P8    0       ( 0.00%)
  P9    0       ( 0.00%)
  P10   0       ( 0.00%)
  P11   0       ( 0.00%)
  P12   0       ( 0.00%)
  P13   0       ( 0.00%)
  P14   0       (100.00%)
  Avg freq      798380  KHz
```

Figure 17

```
localhost:~ # xenpm get-cpuidle-states 0
Max possible C-State: C0

cpu id                  : 0
total C-States          : 4
idle time(ms)           : 70157197
C0                      : transition [           256584]
                          residency  [         69174343 ms]
C1                      : transition [            81488]
                          residency  [             5153 ms]
C2                      : transition [           169547]
                          residency  [            66842 ms]
C3                      : transition [             5548]
                          residency  [           933516 ms]
pc2                     : [          19740 ms]
pc6                     : [         527049 ms]
cc6                     : [         756835 ms]

localhost:~ #
```

Figure 18   xenpm command

According to the above output, all activities can be found in CPU C0 state.

When setting a maximum C-State, note that C-State numbers do not necessarily correspond directly with ACPI C-State names. The C-States mentioned in the output shown in Figure 18 on page 16 are simply a numerical list of all the C-States that Xen knows about, listed as C0, C1, C2, etc. This means that ACPI C0 will appear as C0, ACPI C1 will appear as C1, ACPI C1E will appear as C2, and so on. Furthermore, some states might not be individually addressable with `xenpm set-max-cstate <X>` and setting the maximum C-State to X does not necessarily mean that only C0..CX will be used.

Resources

- Enhanced Intel SpeedStep Technology

  https://edc.intel.com/content/www/us/en/design/ipla/software-development-platforms/client/platforms/alder-lake-desktop/12th-generation-intel-core-processors-datasheet-volume-1-of-2/001/enhanced-intel-speedstep-technology_1/

- AMD PowerNow!

  https://en.wikipedia.org/wiki/PowerNow%21

- Xen Project Source Repositories

  https://xenbits.xen.org/

- Xen power management

  https://wiki.xenproject.org/wiki/Xen_power_management

- Linux Kernel documentation - Working-State Power Management

  https://www.kernel.org/doc/html/latest/admin-guide/pm/working-state.html

- Intel VTune Profiler User Guide - P-State

  https://www.intel.com/content/www/us/en/develop/documentation/vtune-help/top/reference/energy-analysis-metrics-reference/P-State.html?wapkw=p%20state

- Power Management - 001 - ID:655258 | 12th Generation Intel Core Processors Datasheet, Volume 1 of 2

  https://edc.intel.com/content/www/us/en/design/ipla/software-development-platforms/client/platforms/alder-lake-desktop/12th-generation-intel-core-processors-datasheet-volume-1-of-2/001/power-management/

- SLES 15 SP3 - Introduction to Xen virtualization

  https://documentation.suse.com/sles/15-SP3/single-html/SLES-virtualization/#cha-xen-basics

- Power Settings in Citrix Hypervisor: C-States, Turbo and CPU Frequency Scaling

  https://support.citrix.com/article/CTX200390

- Linux Kernel Driver DataBase: CONFIG_XEN_ACPI_PROCESSOR: Xen ACPI processor (cateee.net)

  https://cateee.net/lkddb/web-lkddb/XEN_ACPI_PROCESSOR.html

Authors

Conie Chang is a Linux Engineer in the Lenovo Infrastructure Solutions Group, based in Taipei, Taiwan. She has experience in Red Hat and SUSE Linux OS.

Thanks to the following people for their contributions to this project:

- Michelle CF Tsai, Lenovo Linux Engineer
- Alpus Chen, Lenovo VMWare Engineer
- Gary Cudak, Lenovo OS Architect
- David Watts, Lenovo Press

Notices

Lenovo may not offer the products, services, or features discussed in this document in all countries. Consult your local Lenovo representative for information on the products and services currently available in your area. Any reference to a Lenovo product, program, or service is not intended to state or imply that only that Lenovo product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any Lenovo intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any other product, program, or service.

Lenovo may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not give you any license to these patents. You can send license inquiries, in writing, to:

Lenovo (United States), Inc.
1009 Think Place - Building One
Morrisville, NC 27560
U.S.A.
Attention: Lenovo Director of Licensing

LENOVO PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some jurisdictions do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. Lenovo may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

The products described in this document are not intended for use in implantation or other life support applications where malfunction may result in injury or death to persons. The information contained in this document does not affect or change Lenovo product specifications or warranties. Nothing in this document shall operate as an express or implied license or indemnity under the intellectual property rights of Lenovo or third parties. All information contained in this document was obtained in specific environments and is presented as an illustration. The result obtained in other operating environments may vary.

Lenovo may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation to you.

Any references in this publication to non-Lenovo Web sites are provided for convenience only and do not in any manner serve as an endorsement of those Web sites. The materials at those Web sites are not part of the materials for this Lenovo product, and use of those Web sites is at your own risk.

Any performance data contained herein was determined in a controlled environment. Therefore, the result obtained in other operating environments may vary significantly. Some measurements may have been made on development-level systems and there is no guarantee that these measurements will be the same on generally available systems. Furthermore, some measurements may have been estimated through extrapolation. Actual results may vary. Users of this document should verify the applicable data for their specific environment.

This document was created or updated on May 19, 2022.

Send us your comments via the **Rate & Provide Feedback** form found at
http://lenovopress.com/lp1596

Trademarks

Lenovo and the Lenovo logo are trademarks or registered trademarks of Lenovo in the United States, other countries, or both. These and other Lenovo trademarked terms are marked on their first occurrence in this information with the appropriate symbol (® or ™), indicating US registered or common law trademarks owned by Lenovo at the time this information was published. Such trademarks may also be registered or common law trademarks in other countries. A current list of Lenovo trademarks is available from https://www.lenovo.com/us/en/legal/copytrade/.

The following terms are trademarks of Lenovo in the United States, other countries, or both:

Lenovo®	Lenovo(logo)®

The following terms are trademarks of other companies:

Intel, Intel Core, Intel SpeedStep, Itanium, VTune, and the Intel logo are trademarks or registered trademarks of Intel Corporation or its subsidiaries in the United States and other countries.

Linux is a trademark of Linus Torvalds in the United States, other countries, or both.

Other company, product, or service names may be trademarks or service marks of others.