Tuning UEFI Settings for Performance and Energy Efficiency on 4th Gen Intel Xeon Scalable Processor-Based ThinkSystem Servers

Planning / Implementation

The Lenovo ThinkSystem UEFI provides an interface to the server firmware that controls boot and runtime services. The server firmware contains numerous tuning parameters that can be set through the UEFI interface. These tuning parameters can affect all aspects of how the server functions and also how well the server performs.

Lenovo ThinkSystem UEFI offers four convenient operating modes that set pre-defined tuning parameters:

- Minimal Power
- Efficiency - Favor Power
- Efficiency - Favor Performance
- Maximum Performance

This paper describes the tuning parameter settings for each operating mode and other tuning parameters to consider for performance and efficiency. This paper covers ThinkSystem servers with 4th Gen Intel Xeon Scalable (Sapphire Rapids) processors.

Tuning UEFI series

This paper is one in a series on the tuning of UEFI settings on ThinkSystem servers

- Tuning UEFI on servers with AMD processors
  - 2nd, 3rd Gen AMD EPYC processors
  - 4th Gen AMD EPYC processors
  - 5th Gen AMD EPYC processors
- Tuning UEFI on servers with Intel processors:
  - 1st, 2nd, 3rd Intel Xeon Scalable processors
  - 4th Gen Intel Xeon Scalable processors (this paper)

Summary of operating modes

The ThinkSystem servers with Intel processors offer four preset operating modes, Minimal Power, Efficiency – Favor Power, Efficiency – Favor Performance and Maximum Performance. These modes are a collection of predefined low-level UEFI settings that simplify the task of tuning the server for either minimal power, maximum performance, or efficiency.

The four pre-defined modes are as follows:

- **Minimal Power**: Minimize the absolute power consumption of the system.
- **Efficiency – Favor Power**: Maximize the performance/watt efficiency with a bias towards power savings. It is expected that will be the favored mode for SPECpower benchmark testing.
- **Efficiency – Favor Performance** (default): Maximize the performance/watt efficiency with a bias towards performance. It is the favored mode for Energy Star certification, EU Lot 9, and is comparable to Intel Optimized Power Mode (OPM).
- **Maximum Performance**: Achieves maximum performance but with higher power consumption and lower energy efficiency.

Table 1 summarizes the settings that are made for each mode selected for the ThinkSystem server Intel platforms. Table 2 summarizes additional settings not in the preset operating modes that can be tuned for performance and efficiency. The values in the Category column (column 3) in each table are as follows:

- **Recommended**: Settings follow Lenovo's best practices and should not be changed without sufficient justification.
- **Suggested**: Settings follow Lenovo's general recommendation for a majority of workloads, but these settings can be changed if justified by workload specific testing.
- **Test**: The non-default values for the Test settings can optionally be evaluated because they are workload dependent.

Table 1. UEFI Settings for operating modes on ThinkSystem servers with Intel processors

Menu Item	Category	Minimal Power	Efficiency – Favor Power	Efficiency – Favor Performance	Maximum Performance
Operating Mode	Suggested	Minimal power	Efficiency – Favor Power	Efficiency – Favor Performance	Maximum Performance
CPU P-State Control	Recommended	Autonomous	Autonomous	Autonomous	None
C-States	Recommended	Legacy	Legacy	Legacy	Disabled
MONITOR/ MWAIT	Suggested	Enabled	Enabled	Enabled	Disabled
C1E Enhanced Mode	Recommended	Enabled	Enabled	Enabled	Disabled
UPI Link Frequency	Recommended	Minimal power	Minimal Power	Max Performance	Max performance
UPI Link Disable	Recommended	Minimum Number of Links Enabled	Minimum Number of Links Enabled	Enabled All Links	Enabled all links
UPI Power Management	Recommended	L1: Enabled	L1: Enabled	L1: Enabled	L1: Disabled
Turbo Mode	Suggested	Disabled	Disabled	Enabled	Enabled
Energy Efficient Turbo	Recommended	Enabled	Enabled	Enabled	Disabled
Power/ Performance Bias	Suggested	Platform Controlled	Platform Controlled	Platform Controlled	Platform Controlled
Platform Controlled Type	Suggested	Minimal Power	Efficiency – Favor Power	Efficiency – Favor Performance	Maximum Performance
Memory Speed	Recommended	Minimal power	Balanced	Max Performance	Max Performance
Page Policy	Recommended	Closed	Closed	Closed	Closed
ADDDC Sparing	Recommended	Disabled	Disabled	Disabled	Disabled

Table 2 lists additional UEFI settings that you should consider for tuning for performance or energy efficiency. These settings are not part of the preset operating modes.

Table 2. Other UEFI settings to consider for performance and efficiency

Menu Item	Category	Comments
Hyper-Threading	Suggested	This setting allows two separate instruction streams to run simultaneously on each processor core. Most applications benefit from hyperthreading so this parameter should be left Enabled unless it is known that performance degrades with a specific application. Applications that can fully utilize a core with a single instruction stream should set hyperthreading to Disabled.
Cores in CPU Package	Recommended	This setting logically powers off a set number of cores for each processor in a system.
CPU Frequency Limits	Suggested / Test	Enable one of the CPU Frequency Limit options to restrict the turbo frequency increase. This will minimize performance variability among a population of servers caused by core frequency jitter at the expense of achieving peak performance and maximum turbo frequency upside. This may be a better option than disabling turbo mode.
Processors X to Y Cores Active	Test	Related to CPU Frequency Limits and used for restricting the maximum core frequency.
Hardware Prefetcher	Recommended	Also known as the MLC Streamer Prefetcher. When enabled, this parameter fetches the next cache line from memory into the processor's L2 cache if two consecutive cache lines were read. This parameter should be Enabled unless experiments have been run selectively disabling the prefetchers one at a time and performance is improved when Disabled.
Adjacent Cache Prefetch	Recommended	Also known as the MLC Spatial Prefetcher. When enabled, this parameter fetches both cache lines that make up a 128-byte cache line pair even if the requested data is only in the first cache line. This parameter should be Enabled unless experiments have been run selectively disabling the prefetchers one at a time and performance is improved when Disabled.
DCU Streamer Prefetcher	Recommended	When enabled, this parameter fetches the next cache line into the L1 data cache when multiple loads from the same cache line are executed in a certain time limit. This parameter should be Enabled unless experiments have been run selectively disabling the prefetchers one at a time and performance is improved when Disabled.
DCU IP Prefetcher	Recommended	When enabled, this parameter fetches the next cache line into the L1 data cache if there is a sequential load history of cache line accesses. This parameter should be Enabled unless experiments have been run selectively disabling the prefetchers one at a time and performance is improved when Disabled.
XPT Prefetcher	Recommended	Recommend this parameter remain Enabled unless experiments have been run showing performance is improved when Disabled.
UPI Prefetcher	Suggested	Recommend this parameter remain Enabled unless experiments have been run showing performance is improved when Disabled.
L2 RFO Prefetcher	Recommended	Primarily used in conjunction with several other settings to optimize for large application environments on 4-socket and 8-socket configurations.
LLC Prefetch	Suggested / Test	When enabled, this parameter gives the core prefetcher the ability to prefetch data directly into the LLC without necessarily filling into the MLC
Direct Cache Access (DCA)	Suggested	Experiments can be run with this parameter Enabled or Disabled depending on whether an IO device supports Direct Cache Access.
P-State Hysteresis	Suggested / Test	Test for environments that do not disable CPU P-state Control. A lower value can lead to better performance.
Rocket Mode	Suggested / Test	CPU P-state Control must be set to Autonomous. When enabled, this parameter allows the cores to jump to maximum turbo frequency instantly as opposed to a smooth ramp up.
Workload Configuration	Suggested	I/O sensitive should be used with expansion cards that require high I/O bandwidth when the CPU cores are idle to allow enough frequency for the workload.

How to use OneCLI and Redfish

In addition to using UEFI Setup, Lenovo also provides OneCLI/ASU variables and Redfish UEFI Setting Attribute names for managing system settings.

- **OneCLI/ASU variable usage**
  Show current setting:

  ```
  Onecli config show "<OneCLI/ASU Var>" --override --log 5 --imm <userid>:<password>
  @<IP Address>
  ```

  Example:

  ```
  onecli config show "OperatingModes.ChooseOperatingMode" --override --log 5 --imm U
  SERID:PASSW0RD@10.240.218.89
  ```

  Set a setting:

  ```
  Onecli config set "<OneCLI/ASU Var>" "<choice>" -override -log 5 -imm <userid>:<pa
  ssword>@<IP Address>
  ```

  Example:

  ```
  onecli config set "OperatingModes.ChooseOperatingMode" "Maximum Efficiency" --over
  ride --log 5 --imm USERID:PASSW0RD@10.240.218.89
  ```

- **Redfish Attributes configure URL**
  Setting get URL: `https://<BMC IP>/redfish/v1/Systems/Self/Bios`
  Setting set URL: `https://<BMC IP>/redfish/v1/Systems/Self/Bios/SD`

  Example:

  Get URL: `https://10.240.55.226/redfish/v1/Systems/Self/Bios`
  Set URL: `https://10.240.55.226/redfish/v1/Systems/Self/Bios/SD`

- **Redfish Value Names of Attributes**
  If no special description, choice name is same as possible values. If there is a space character (‘ ‘), dash character (‘-‘) or forward slash character (‘/’) in the possible values, replace them with underline (“_”). This is because the Redfish standard doesn’t support those special characters.

  If you use OneCLI to configure the setting, OneCLI will automatically replace those characters with an underline character. However, if you use other Redfish tools, then you may need to replace them manually.

  For example, “Operating Mode” has three choices: Maximum Efficiency, Maximum Performance and Custom Mode, their Redfish value names are `MaximumEfficiency`, `MaximumPerformance` and `CustomMode`.

For more detailed information on the BIOS schema, please refer to the DMTF website:
https://redfish.dmtf.org/redfish/schema_index

Usually, postman can be used for get/set BIOS schema:
https://www.getpostman.com/

The remaining sections in this paper provide details about each of these settings. We describe how to access the settings via System Setup (Press F1 during system boot).

UEFI menu items

The following items are provided to server administrators in UEFI menus that are accessible by pressing F1 when a server is booted, through the XClarity Controller (XCC) service processor, or through command line utilities such as Lenovo's Advanced Settings Utility (ASU) or OneCLI.

These parameters are made available because they are regularly changed from their default values to fine tune server performance for a wide variety of customer use cases.

Menu items described in this paper for ThinkSystem servers with 4th Gen Intel Xeon Scalable processors are as follows:

- Settings for Operating modes
- Settings for Processors
- Settings for Memory
- Settings for Power
- Settings for Devices and I/O Ports
- Hidden UEFI Items

Settings for Operating modes

Settings for Operating modes:

- Choose Operating Mode
- CPU P-state Control
- C-States
- MONITOR/MWAIT
- C1 Enhanced Mode
- UPI Link Frequency
- UPI Link Disable
- UPI Power Management
- Turbo Mode
- Energy Efficient Turbo
- Power/Performance Bias
- Platform Controlled Type
- Memory Speed
- Page Policy

Choose Operating Mode

This setting is used to set multiple processor and memory variables at a macro level.

Choosing one of the predefined Operating Modes is a way to quickly set a multitude of processor, memory, and miscellaneous variables. It is less fine grained than individually tuning parameters but does allow for a simple "one-step" tuning method for two primary scenarios.

Tip: Prior to optimizing a workload for maximum performance, it is recommended to set the Operating Mode to "Maximum Performance" and then reboot rather than simply starting from the "Efficiency – Favor Performance" default mode and then modifying individual UEFI parameters. If you do not do this, some settings may be unavailable for configuration.

This setting is accessed as follows:

- System setup: **System Settings** -> **Operating Modes** > **Choose Operating Mode**
- OneCLI/ASU variable: `OperatingModes.ChooseOperatingMode`
- Redfish attribute: `OperatingModes_ChooseOperatingMode`

Possible values:

- **Minimal Power**
  Minimal Power mode strives to minimize the absolute power consumption of the system while it is operating. In addition, if a customer wants additional power savings, they can set a power cap or fan speed restriction. The tradeoff with the minimal power mode and/or power cap or fan speed restriction is that performance may be reduced depending on the application that is running.

- **Efficiency – Favor Power**
  Efficiency - Favor Power mode maximizes the performance/watt efficiency with a bias towards power savings. It provides the best features for reducing power and increasing performance in applications where maximum bus speeds are not critical. It is expected that this will be the favored mode for SPECpower testing. This mode maintains backwards compatibility with systems that included the preset operating modes before Energy Star for servers was released.
- **Efficiency – Favor Performance**
  Efficiency - Favor Performance mode optimizes the performance/watt efficiency with a bias towards performance. It is the favored mode for Energy Star, EU Lot 9, and is comparable to Intel Optimized Power Mode (OPM). Note that this mode is slightly different than the Favor Power mode. In Favor Performance mode, no bus speeds are derated as they are in Favor Power mode.
- **Maximum Performance**
  Maximum Performance mode will maximize the absolute performance of the system without regard for power. In this mode, power consumption is a “don't care”. Things like fan speed and heat output of the system may increase in addition to power consumption. Efficiency of the system may go down in this mode, but the absolute performance can go up depending on the benchmark that is run. Note, for maximum performance on applications that do not utilize all CPU cores simultaneously, it is best to select ‘Maximum Performance’ first, then select ‘Custom’ and enable C-states. Doing so will allow the active cores to achieve maximum turbo uplift.
- **Custom Mode**
  Custom Mode allows the user to customize the performance & power settings. Custom Mode will inherit the UEFI settings from the previous preset operating mode. For example, if the previous operating mode was the Maximum Performance operating mode and then Custom Mode was selected, all the settings from the Maximum Performance operating mode will be inherited. Note that there are certain settings that may be mutually exclusive or interdependent. For these settings, an error will be surfaced if one of the pre-requisite or interrelated settings is set in such a way as to make configuration of the setting in question non-valid.

CPU P-state Control

P-states (Intel Enhanced SpeedStep) dynamically adjusts core frequency and voltage dependent on processor utilization. This setting allows for a processor to have a variable core voltage which allows for either a low power or base rated clock speed frequency. P-states work by adjusting processor core voltage and frequency.

All modern operating systems have drivers that support P-states. The amount of adjustment to the core voltage and frequency (either up or down) is dependent on the type and stepping of a CPU, the P-states mode, and requests from operating system drivers.

Autonomous mode is used for normal power savings and serves well for most typical business applications. Those applications which are clock frequency sensitive it is recommended to test with Cooperative or Legacy mode, while for those workloads which are latency sensitive Lenovo recommends setting the value to None.

This setting is accessed as follows:

- System setup: System **Settings** → **Processors** → **CPU P-state Control**
- OneCLI/ASU variable: `Processors.CPUPstateControl`
- Redfish attribute: `Processors_CPUPstateControl`

Possible values:

- **Autonomous** (Default)
  All CPU P-state management is handled automatically in the background without any OS intervention.
- **Cooperative without Legacy**
  UEFI does not provide legacy P-States. OS provides hints to the processor’s PCU on the desired P-state min / max levels. Requires Windows Server 2016 and Linux kernel v4.2 and higher. PCU runs in Autonomous mode until the OS sets the desired frequency.
- **Cooperative with Legacy**
  UEFI leaves the legacy P-states interface initially enabled until/if later an OS that is aware of Intel Hardware P-states (HWP) native mode sets the bit. Legacy P-sates will be used until OS sets the HWP native mode. After that, P-states will switch to same behavior as “Cooperative without Legacy”.

- **Legacy**
  Legacy control mechanisms currently implemented for systems with processors prior to the Intel Xeon Scalable Processor codenamed Skylake. Uses standard ACPI interface. Use for applications which benefit from OS level power controls.
- None
  No ACPI table entries for P-states are created. P-states are disabled. Use this setting to minimize latency caused by P-state transitions.

C-States

This setting specifies the highest level of C-state allowed on a system. The higher a C-state is, as denoted by its C-state modifier being numerically larger (i.e., C1 has a higher power and activity state then C3) the less power is drawn from that core / CPU. Deeper (i.e., "higher") C-states achieve their power savings by progressively shutting down parts of a processor package, from idling inactive cores all the way to shutting down idle cores.

C-states are found at the package and core level. Core C-states resolve to the highest power threads running on a core, whereas package C-states resolve to the highest power C-state of all cores in a processor package. Package C-states affect the entire processor and entail a deeper low power state and even higher exit latencies then core C-states.

Package C-states are engaged when all cores on a processor package are at a C-state level >= package C-state level requested (i.e. - all cores must be at >= C1e for the package C-state to go to C1e).

It is important to understand that as C-states go deeper, a processor's instruction execution latency increases and the probability of cache coherency decreases.

The C-States are beneficial if either power savings are important or if used in conjunction with Turbo Mode and *bursty* applications (i.e., applications that have cyclical periods of inactivity followed by periods of large processing demands) to provide additional power savings for Turbo Mode to draw from. Cache coherent applications (Database & HPC-type applications) will suffer performance hits if they are highly threaded due to the low power states flushing processor and package caches. If an application typically utilizes less than all the CPU cores and reaching the maximum CPU frequency is important, c-states should be enabled so that the active cores can reach maximum turbo frequency.

This setting is accessed as follows:

- System setup: **System Settings → Processors → C-States**
- OneCLI/ASU variable: `Processors.CStates`
- Redfish attribute: `Processors_CStates`

Possible values:

- **Legacy** (Default)
  C-states are presented to OS via ACPI table entries. OSPM controls core C-state transitions. PCU controls package C-state transitions.
- **Disabled**
  C-states are disabled.

MONITOR/MWAIT

Some operating systems (for example, some Linux distributions) engage C-states by using MONITOR/MWAIT instructions and not the ACPI table. These operating systems will still enter higher C-states even if the C-States UEFI parameter is Disabled. To prevent this, you need to disable MONITOR/MWAIT.

This setting is accessed as follows:

- System setup: **System Settings → Processors → MONITOR/MWAIT**
- OneCLI/ASU variable: `Processors.MONITORMWAIT`
- Redfish attribute: `Processors_MONITORMWAIT`

Possible values:

- **Enabled** (Default)
  Enable MONITOR/MWAIT instructions support.
- **Disabled**
  Disable MONITOR/MWAIT instructions support.

C1 Enhanced Mode

C1 Enhanced mode (C1E) is a processor power saving feature that halts cores not in use and maintains cache coherency. C1E maintains all of the C1 halt state functionality, but the core voltage is reduced for enhanced power savings. If all cores in a package are in C1 state, the package itself will enter C1E unless C1E is disabled.

C1E can help to provide power savings in those circumstances where cache coherency is paramount. Those applications which thread well and can maintain utilization of processor cores (virtualization, HPC and database workloads) do not benefit and under certain circumstances may be hindered by C1E. If a user is attempting to achieve maximum opportunity for Turbo Mode to engage, C1E is recommended. C1E is not recommended for latency sensitive workloads.

Note: This item is only valid if C-states are set to legacy. If C-states are set to Autonomous the C1E item is statically set to Enabled.

This setting is accessed as follows:

- System setup: **System Settings → Processors → C1 Enhanced Mode**
- OneCLI/ASU variable: `Processors.C1EnhancedMode`
- Redfish attribute: `Processors_C1EnhancedMode`

Possible values:

- **Disabled**
- **Enabled** (Default)

UPI Link Frequency

UPI Link frequency determines the rate at which the UPI processor interconnect link will operate.

If a workload is highly NUMA aware, sometimes lowering the UPI link frequency can free up more power for the cores and result in better overall performance. But there is also interplay between this menu item and UPI link disable and Uncore Frequency Scaling.

This setting is accessed as follows:

- System setup: **System Settings → Processors → UPI Link Frequency**
- OneCLI/ASU variable: `Processors.UPILinkFrequency`
- Redfish attribute: `Processors_UPILinkFrequency`

Possible values:

- **Maximum Performance** (Default)
  Set UPI interface to rated speed for CPU SKU
- **Balanced**
  Set UPI interface to N-1 of rated speed for CPU SKU
- **Minimal Power**
  Set UPI interface to minimum rated speed for CPU family

UPI Link Disable

Disable one of the available UPI processor interconnect links on the CPU. Use UPI Link disable when attempting to conserve power on a platform. Disabling a UPI Link will impact performance for any workload which does cross socket communication. Disabling a UPI Link is not recommended for high frequency, low jitter, low latency, virtualization, or database workloads.

This setting is accessed as follows:

- System setup: **System Settings → Processors → UPI Link Disable**
- OneCLI/ASU variable: `Processors.UPILinkDisable`
- Redfish attribute: `Processors_UPILinkDisable`

Possible values:

- **Enabled All Links** (Default)
  Enable all available UPI links between CPUs.
- **Minimum Number of Links Enabled**
  Enables the minimum number of UPI links required by the processor architecture. The actual number of UPI links may vary depending on the server's design, such as 2-socket compared to 4-socket mesh.

UPI Power Management

UPI power management is used when attempting to conserve power on a platform. Low power mode reduces UPI frequency and bandwidth. It is recommended for power saving, however, UPI power management is not recommended for high frequency, low jitter, low latency, virtualization, or database workloads. For these workloads Lenovo recommends setting L1 to Disabled. Note, L0p is now set statically to Disabled as per Intel recommendation and cannot be changed.

L1 saves the most power but has the highest impact on latency and bandwidth. L1 takes a UPI link and allows it to transition from Full Width>x8 width>Link down. L1 is the deepest power savings state.

This setting is accessed as follows:

- System setup: **System Settings → Processors → UPI Power Management**
- OneCLI/ASU variable: `Processors.L1`
- Redfish attribute: `Processors_L1`

Possible values:

- **Enabled** (Default)
- **Disabled**

Turbo Mode

Turbo Mode implements Intel Turbo Boost Technology and allows the processor to opportunistically increase a set of CPU cores higher than the CPU's rated base clock speed based on the number of active cores, power, and thermal headroom in a system. It is important to understand that this is not a guarantee of a CPU frequency increase, rather it is enabling the opportunity to run at a higher clock frequency. The performance of Turbo Mode increases when fewer cores are active, dynamic power management is enabled, and the system is running below the thermal design limits for the platform.

Turbo Mode is denoted as a series of numbers in the form `a/b/c/d/x`, where each number is a MHz increment in CPU frequency. This corresponds to the number of cores active in the package. It can generally be interpreted as:

```
all cores / a-1 cores / a-2 cores / a-x cores
```

Use Turbo Mode when you have applications which can benefit from clock frequency enhancements. Avoid using this feature with latency sensitive or clock frequency sensitive (low jitter) applications, or if power consumption is a concern.

This setting is accessed as follows:

- System setup: **System Settings → Processors → Turbo Mode**
- OneCLI/ASU variable: `Processors.TurboMode`
- Redfish attribute: `Processors_TurboMode`

Possible values:

- **Enabled** (Default)
- **Disabled**

Energy Efficient Turbo

When Energy Efficient Turbo is enabled, the CPU's optimal turbo frequency will be tuned dynamically based on CPU utilization. The actual turbo frequency the CPU is set to is proportionally adjusted based on the duration of the turbo request.

Memory usage of the OS is also monitored. If the OS is using memory heavily and the CPU core performance is limited by the available memory resources, the turbo frequency will be reduced until more memory load dissipates, and more memory resources become available. The power/performance bias setting also influences energy efficient turbo.

Energy Efficient Turbo is best used when attempting to maximize power consumption over performance.

This setting is accessed as follows:

- System setup: **System Settings → Processors → Energy Efficient Turbo**
- OneCLI/ASU variable: `Processors.EnergyEfficientTurbo`
- Redfish attribute: `Processors_EnergyEfficientTurbo`

Possible values:

- **Enabled** (Default)
  Enable Energy Efficient Turbo to optimize power consumption over performance.
- **Disabled**
  Disable Energy Efficient Turbo.

Power/Performance Bias

Power/Performance Bias controls how aggressively the CPU will be power managed and placed into turbo mode. Power/Performance bias has no effect on CPU frequencies when P-states are disabled, but it can still influence CPU power management features.

Lenovo recommends enabling Platform Controlled for workloads requiring low jitter and low latency. For High Frequency workloads, it is suggested to set the parameter to Platform Controlled, however some multi-threaded applications may benefit from setting it to OS Controlled, allowing the OS to control power management due to thread scheduling allowing for non-used cores to be placed in a halt / lower power mode.

This setting is accessed as follows:

- System setup: **System Settings → Power → Power/Performance Bias**
- OneCLI/ASU variable: `Power.PowerPerformanceBias`
- Redfish attribute: `Power_PowerPerformanceBias`

Possible values:

- **Platform Controlled** (Default)
  The system UEFI configuration determines if and how aggressively the platform will enable power management and turbo mode.
- **OS Controlled**
  The operating system controls how aggressively power management and Turbo Mode will be used via legacy ACPI calls requesting processor performance governors. Not all operating systems support OS control of power management and performance governors. In the event that an OS does not support power management instrumentation, the processor's power control unit (PCU) defaults to Maximum Performance until an OS makes power management calls.

Platform Controlled Type

Controls how aggressively the processor's Power Control Unit (PCU) will engage power management and how the CPU cores are placed into turbo mode. When set to Maximum Performance, turbo mode can be engaged opportunistically before it is requested. In addition, the lowest engagement of uncore power management features occurs.

Lenovo recommends enabling Maximum Performance for all systems when workload performance is more important than power savings.

This setting is accessed as follows:

- System setup: **System Settings → Power → Platform Controlled Type**
- OneCLI/ASU variable: `Power.PlatformControlledType`
- Redfish attribute: `Power_PlatformControlledType`

Possible values:

- **Maximum Performance**
  Turbo mode can be engaged opportunistically before it is requested and uncore power management features (Memory, UPI, C-state demotion, I/O bandwidth limit and UFS) are aggressively disabled.

- **Efficiency – Favor Performance** (Default)
  Turbo is quickly engaged but not opportunistically. Some light uncore and UPI power management features are engaged to provide low-cost power savings with an overall weighting towards per core and per package performance.
- **Efficiency - Favor Power**
  Turbo is engaged more slowly but uncore and UPI power management policies are quickly engaged to balance the systems towards power savings.
- **Minimal Power**
  Turbo is not engaged and aggressive power management policies on the uncore and UPI are engaged to drive lowest performance / watt.

Memory Speed

The Memory Speed setting determines the frequency at which the installed memory will run. Memory speed setting could be changed if the objective is to conserve power, since lowering the clock frequency to the installed memory will reduce overall power consumption of the DIMMs.

This setting is accessed as follows:

- System setup: **System Settings → Memory → Memory Speed**
- OneCLI/ASU variable: `Memory.MemorySpeed`
- Redfish attribute: `Memory_MemorySpeed`

Possible values:

- **Maximum Performance** (Default)
  Maximum Performance sets the memory to the maximum allowed frequency as dictated by the type of CPU installed and the memory DIMMs installed.
- **Balanced**
  Balanced attempts to balance power and performance for the memory subsystem. The DIMMs will run at a frequency of N-1 of the rated frequency for the CPU and memory DIMMs installed. Memory voltage will be set to the lowest supported value to drive N-1 frequency.
- **Minimal Power**
  Minimal power optimizes memory power savings over performance. Memory is set to run at the minimum support memory frequency supported by the platform. For the 4th Gen Intel Xeon Scalable processors, this is 4000 MHz.

Page Policy

Page Policy setting determines whether the memory controller keeps the last accessed page open. With the Open Page policy, memory access is faster in case of a Page hit but slower in case of a Page Miss.

Lenovo recommends using the Adaptive mode page policy for low latency and high-performance workloads due to lower loaded latencies and better overall performance. Closed Page mode works well for workloads such as databases that may do many streaming reads / writes to the memory subsystem.

This setting is accessed as follows:

- System setup: **System Settings → Memory → Page Policy**
- OneCLI/ASU variable: `Memory.PagePolicy`
- Redfish attribute: `Memory_PagePolicy`

Possible values:

- **Closed (Default)**
  Closed page policy maps memory pages out to avoid “hot spots” on DIMMs, it is useful for workloads such as databases that may do many memory transactions. Closed page mode is recommended as the default setting for the DDR5 page policy for Sapphire Rapids due to several architectural differences between server DDR5 and DDR4 technologies, and CPU considerations that change over time.
- **Adaptive**
  Adaptive page policy balances mapping out memory pages across the DIMM infrastructure to provide both performance and latency. Recommended for DDR4-based servers.

Settings for Processors

Settings for Processors:

- P-State Hysteresis
- Rocket Mode
- Hyper-Threading
- Cores in CPU Package
- CPU Frequency Limits
- Processors X to Y Cores Active
- Processor Prefetchers
- XPT Prefetcher
- UPI Prefetcher
- L2 RFO Prefetcher
- LLC Prefetch
- DCA (Direct Cache Access)
- Uncore Frequency Scaling
- SNC
- UMA-Based Clustering
- Snoop Preference
- Intel Speed Select Technology
- C0 Nap Time
- C-state Interrupt Response Time
- PCH PCIe Relaxed Ordering
- CPU PCIe Relaxed Ordering
- Intel Virtualization Technology

P-State Hysteresis

This setting controls the minimum dwell time before a P-state change occurs. A higher value can lead to more efficient operation. A lower value can lead to better performance.

This setting is accessed as follows:

- System setup: **System Settings → Processors → P-State Hysteresis**
- OneCLI/ASU variable: `Processors.P-stateHysteresis`
- Redfish attribute: `Processors_P-stateHysteresis`

Possible Values:

- **500 us** (Default)
- **50 us**
- **0 us**

Rocket Mode

This setting enables cores to jump to max Turbo instantly as opposed to a smooth ramp up. When Rocket Mode is enabled, it is only engaged when Processor P-states are set to Autonomous.

This setting is accessed as follows:

- System setup: **System Settings → Processors → Rocket Mode**
- OneCLI/ASU variable: `Processors.RocketMode`
- Redfish attribute: `Processors_ RocketMode`

Possible Values:

- **Disabled** (Default)
- **Enabled**

Hyper-Threading

Hyper-threading (HT) is the capability of a single core to execute two threads simultaneously. An OS will register a hyper-thread as a logical CPU and attempt to schedule instruction threads accordingly. All processor cache is shared between the physical core and its corresponding hyper-thread. Many operating systems and hypervisors are able to schedule instructions such that both threads execute on the same core.

Hyper-threading takes advantage of out-of-order execution, deeper execution pipelines and improved memory bandwidth in today's processors to be an effective way of garnering many of the benefits of additional logical CPUs without having to supply the power necessary to drive a physical core.

Most workloads can improve performance with hyper-threading enabled or at least not degrade performance. Virtualization and database workloads generally benefit from HT as they can spawn threads and intelligently place them on the same physical core.

Disable hyper-threading if one instruction stream (thread) can fully utilize a physical core. HPC and scientific workloads can be adversely impacted due to sharing of processor cache resources with the hyper-thread (reducing the logical CPU to cache ratio by 50%) and the fact that a hyper-thread is still dependent on sharing the same instruction execution engines in a single core. Applications that are thread bound may also see a performance impact as they tend to have larger cache to core ratios. Disable hyper-threading for latency sensitive applications, and for some high frequency trading and high-performance computing environments.

The benefit of hyper-threading is workload dependent so evaluate on a case-by-case basis.

This setting is accessed as follows:

- System setup: **System Settings → Processors → Hyper-Threading**
- OneCLI/ASU variable: `Processors.HyperThreading`
- Redfish attribute: `Processors_HyperThreading`

Possible values:

- **Enabled** (Default)
- **Disabled**

Cores in CPU Package

This setting logically powers off a set number of cores for each processor in a system. As opposed to restricting the number of logical processors an OS will run on, this setting directly affects the number of cores physically powered on by turning off the core level power gates on each processor.

Manipulating the number of physically powered cores is primarily used in three scenarios:

- Where users have a licensing model that supports a certain number of active cores in a system.
- Where users have poorly threaded applications but require the additional LLC (last level cache) available to additional processors, but not the core overhead.
- Where users are looking to limit the number of active cores in an attempt to reclaim power and thermal overhead to increase the probability of Turbo Mode being engaged.

This setting is accessed as follows:

- System setup: **System Settings → Processors → Cores in CPU Package**
- OneCLI/ASU variable: `<value> Processors.CoresinCPUPackage`
- Redfish attribute: `Processors_CoresinCPUPackage`

Possible values:

- **All** (Default)
  All cores are enabled.
- **1**
  Only 1 core is enabled in each CPU package.
- **2**
  Only 2 cores are enabled in each CPU package.

- **3**
  Only 3 cores are enabled in each CPU package.
- **N-1**
  N is the maximum cores available in CPU package.

CPU Frequency Limits

Turbo ratio limits can be set between the rated frequency and the default turbo frequencies programmed for the number of cores active. Frequency is set from (Max - 1 bin) to (Max - 4 bins) for number of active cores, in approximately 100MHz increments. A bin is an ~100 MHz increment of core frequency.

It is useful when one desires control over turbo mode frequency uplift to minimize potential frequency fluctuations and reduce jitter.

This setting is accessed as follows:

- System setup: **System Settings → Processors → CPU Frequency Limits**
- OneCLI/ASU variable: `Processors.CPUFrequencyLimits`
- Redfish attribute: `Processors_CPUFrequencyLimits`

Possible values:

- **Full turbo uplift** (Default)
  Turbo could run to maximum frequency supported by CPU when Turbo Mode is enabled.
- **Restrict maximum frequency**
  Turbo is only able to run to limited frequency number when Turbo Mode is enabled.

Processors X to Y Cores Active

This setup menu is available when the parameter CPU Frequency Limits is set to **Restrict maximum frequency** as described in the previous section.

The maximum frequency (turbo, AVX, and non turbo) can be restricted to a frequency that is between the maximum turbo frequency for the CPU installed and 1.2GHz. This can be useful for synchronizing CPU tasks.

Note: The max frequency for N+1 cores cannot be higher than N cores. If an unsupported frequency is entered, it will automatically be limited to a supported value. If the CPU frequency limits are being controlled through application software, leave this menu item at the default, full turbo uplift.

This setting is accessed as follows:

- System setup: **System Settings → Processors → CPU Frequency Limits**
- OneCLI/ASU variable: `Processors.ProcessorsXtoYcoresactive`
  Example: `Processors.Processors17to18coresactive`
- Redfish attribute: `Processors_ProcessorsXtoYcoresactive`

X and Y are used more as engineering numbers.

- X means the start core number of turbo limitation, has a different value SKU to SKU, and can be changed if some CPU cores are disabled.
- Y means the end core number of turbo limitation, equal to actual enabled CPU cores.

User does not need to care the meaning of X and Y, just need to pay attention on the specific variable name when using OneCLI or Redfish to configure this setting.

Possible values:

- **Maximum Turbo Frequency - 1 bin** (Default)
  In the Setup menu, it shows the actual available highest frequency value that can be restricted. For example, 3.40GHz
- **Maximum Turbo Frequency - 2 bins**
- **Maximum Turbo Frequency - 3 bins**
- …

- **Maximum Turbo Frequency - N bins**

In setup menu, it shows the actual available lowest frequency value that can be restricted. For example, 3.2GHz.

Processor Prefetchers

A processor prefetcher attempts to predictively retrieve information from main memory or a higher level processor cache for storage in a processor's L1 or L2 cache. Prefetchers allow a CPU to do more work per clock cycle by maintaining a CPU core fed with information.

When a prefetcher successfully pre-stages information in a CPU's cache, it can serve to lower latency and improve performance. When a prefetcher pre-stages invalid information, there is an extra latency and performance impact as the cache must be flushed of information and new information must be loaded in.

Prefetchers come in two types and two varieties. Prefetchers either pre-stage information to a core's L1 or L2 cache, and either retrieve instructions or data from main memory.

There are up to four tunable prefetchers in ThinkSystem server Intel platforms:

- Data Cache Unit (DCU) IP Prefetcher – a processor L1 instruction cache prefetcher that uses a predictive instruction prefetching scheme. This setting fetches the next cache line into the L1 data cache if there is a sequential load history of cache line accesses. The next cache line is fetched from either L2 or main memory.
- Data Cache Unit (DCU) Streamer Prefetcher – a processor L1 data cache prefetcher that fetches the next cache line into the L1 data cache when multiple loads from the same cache line are executed in a certain time limit. The next cache line is fetched from the L2 cache or main memory.
- Hardware Prefetcher (MLC Streamer Prefetcher) – a processor L2 cache instruction prefetcher that prefetches extra cache lines for every memory request issued. The hardware prefetcher retrieves additional cache lines into the L2 cache of a core based on current requests.
- Adjacent Cache Line Prefetcher (MLC Spatial Prefetcher) – a processor L2 data cache prefetcher that fetches both cache lines that make up a 128- byte cache line pair even if the requested data is only in the first cache line. When disabled, this setting only fetches the required data from a single 64-byte cache line.

Application information access patterns, which tend to be relatively predictable, benefit greatly from prefetching. Most typical Line of Business (LOB) applications, virtualization and scientific applications benefit from having pre-fetching enabled. However, due to the non-inclusive LLC structure of Intel's Intel Xeon Scalable processors, it is recommended to carefully evaluate application performance to determine the benefit of disabling prefetching. Keep these settings enabled unless low-level cache analysis has been done with Intel tools or experiments have been run selectively disabling the prefetchers one at a time.

The Processor Prefetchers setting is accessed as follows:

- System setup:
  - **System Settings→ Processors → Hardware Prefetcher**
  - **System Settings→ Processors → Adjacent Cache Prefetch**
  - **System Settings→ Processors → DCU Streamer Prefetcher**
  - **System Settings→ Processors → DCU IP Prefetcher**
- OneCLI/ASU variable:
  - `Processors.HardwarePrefetcher`
  - `Processors.AdjacentCachePrefetch`
  - `Processors.DCUStreamerPrefetcher`
  - `Processors.DCUIPPrefetcher`
- Redfish attribute:
  - `Processors_HardwarePrefetcher`
  - `Processors_AdjacentCachePrefetch`
  - `Processors_DCUStreamerPrefetcher`
  - `Processors_DCUIPPrefetcher`

Possible values:

- **Enabled** (Default)
- **Disabled**

XPT Prefetcher

Extended Prediction Table (XPT) prefetcher (memory prefetch from the core) is a mechanism that enables a read request that is being sent to the last level cache to speculatively issue a copy of that read to the memory controller prefetching. It is designed to reduce local memory access latency.

XPT prefetcher is an “LLC miss predictor” in each core that will issue a speculative DRAM read request in parallel to an LLC lookup, but only when XPT predicts a “miss” from the LLC lookup. Data from an XPT prefetcher will wait for a short time in the memory and will be returned to the core only if there is an actual LLC miss. Local memory access latency is reduced because the LLC lookup has been bypassed by the speculative DRAM read.

Note: If **XPT Prefetcher** is set to Enabled and an unbalanced memory population is installed and **SNC** is set to Enabled (see SNC (Processor Sub-NUMA Clustering), the prefetcher will remain disabled. Balanced memory means the same capacity/speed/number of DIMMs are installed in the memory channels among all of the memory controllers in the CPU sockets.

XPT prefetcher is best used for those workloads which have good NUMA locality. This setting is recommended for low jitter / low latency workloads.

This setting is accessed as follows:

- System setup: **System Settings → Processors → XPT Prefetcher**
- OneCLI/ASU variable: `Processors.XPTPrefetcher`
- Redfish attribute: `Processors_XPTPrefetcher`

Possible values:

- **Enabled** (Default)
- **Disabled**

UPI Prefetcher

Ultra Path Interconnect (UPI) prefetch enables an early memory read on the memory bus. The UPI receive path spawns a memory read to the memory controller prefetcher.

Note: If **UPI Prefetcher** is set to Enabled and an unbalanced memory population is installed AND SNC is Disabled (see the SNC section), the prefetcher will remain disabled. Balanced memory means all populated memory channels should have the same total memory capacity and the same total number of ranks. In addition, all memory controllers on a processor socket should have the same configuration of memory DIMMs.

UPI prefetcher is best used for those workloads which have good NUMA locality. This setting is recommended for low jitter / low latency workloads. For high frequency workloads it is recommended to leave enabled unless a workload analysis has been performed.

This setting is accessed as follows:

- System setup: **System Settings → Processors → UPI Prefetcher**
- OneCLI/ASU variable: Processors.UPIPrefetcher
- Redfish attribute: Processors_UPIPrefetcher

Possible values:

- **Enabled** (Default)
- **Disabled**

L2 RFO Prefetcher

One of 4 variables (IRQThreshold, StaleAtoS, CRQoSConfiguration/CRFastGoConfiguration, L2RFOPrefetcher) used to optimize performance for SAP HANA on servers with 2-hop memory configurations such as 4-socket ring, 6-socket, and 8-socket configurations.

The Enabled option makes L2 prefetcher less aggressive and lowers NT (Non-Temporal) write bandwidth. Disable limits burstiness and reducing snooping.

This setting is accessed as follows:

- OneCLI/ASU variable: `Processors.L2RFOPrefetcher`

Possible values:

- **Disabled** (Default)
  Enable L2 RFO Prefetch.
- **Enabled**
  Disable L2 RFO Prefetch.

LLC Prefetch

The LLC (Last Level Cache) prefetcher is a new prefetcher added to the Intel Xeon Scalable family as a result of the non-inclusive cache architecture. The LLC prefetcher is an additional prefetch mechanism on top of the existing prefetchers that prefetch data into the core DCU and the MLC.

Enabling LLC prefetch gives the core prefetcher the ability to prefetch data directly into the LLC without necessarily filling into the MLC. In some cases, setting this option to disabled can improve performance.

This setting is accessed as follows:

- OneCLI/ASU variable: `Processors.LLCPrefetch`

Possible values:

- **Disabled** (Default)
- **Enabled**

DCA (Direct Cache Access)

DCA capable I/O devices such as network controllers can place data directly into the CPU cache, which improves response times.

If a network controller supports DCA, this might provide lower packet transfer latency. However, if applications are managing the last-level cache (LLC) directly, this functionality may pollute the LLC state. It is recommended to enable this setting for High Frequency workloads and to evaluate its functionality for low latency and low jitter workloads.

This setting is accessed as follows:

- System setup: **System Settings → Processors → DCA**
- OneCLI/ASU variable: Processors.DCA
- Redfish attribute: Processors_DCA

Possible values:

- **Enabled** (Default)
- **Disabled**

Uncore Frequency Scaling

When Uncore Frequency Scaling is enabled, the CPU uncore will dynamically change speed based on the workload. This involves a separate voltage and frequency scaling for Core /LLC complex. For the Intel Xeon Scalable generation of processors, enabling this feature turns on uncore power savings features.

After enabling this setting, you can use the Uncore Frequency Limit setting to set the maximum frequency when scaling. See the Uncore Frequency Limit section. Lenovo recommends you disable this setting for maximum performance for servers with Intel Xeon Scalable processors.

This setting is accessed as follows:

- System setup: **System Settings → Processors → Uncore Frequency Scaling**
- OneCLI/ASU variable: `Processors.UncoreFrequencyScaling`
- Redfish attribute: `Processors_UncoreFrequencyScaling`

Possible values:

- **Enabled** (Default)

- **Disabled**

SNC

SNC (Processor Sub-NUMA Clustering) partitions Intel Xeon Scalable processor cores and last-level cache (LLC) into disjoint clusters with each cluster bound to a set of memory controllers in the system. SNC improves average latency to the LLC and memory. SNC is a replacement for the cluster on die (COD) feature found in previous processor families such as the Intel Xeon E5 v4 and E7 v4 processors. For a multi-socketed system, all SNC clusters are mapped to unique NUMA domains.

When workloads are highly NUMA affinitized, SNC can enable better memory performance and utilization of the LLC. Recommend testing an application workload prior to enabling.

Note: OS support that recognizes each cluster and a separate NUMA node is necessary to take advantage of SNC. Consult your OS documentation to determine if SNC is supported.

This setting is accessed as follows:

- System setup: **System Settings** → **Processors** → **SNC**
- OneCLI/ASU variable: `Processors.SNC`
- Redfish attribute: `Processors_SNC`

Possible values:

- **Disabled** (Default)
  The LLC is treated as one cluster when this option is disabled.
- **SNC2**
  2-way Sub-NUMA Clustering partitions the last level cache (LLC) into two clusters called NUMA nodes that contain an equal number of cores, equal number of LLC slices in close proximity to the cores, an equal amount of socket address space, and with each cluster bound to a subset of the memory controllers in the socket.

  This requires left-to-right DIMM symmetry. If there is not left-to-right DIMM symmetry, SNC will be forced to Disabled.

  When SNC2 is selected, the OS sees two NUMA nodes per physical CPU package.
- **SNC4**
  4-way Sub-NUMA Clustering is recommended for best latencies for core to cache / memory. The cores are affinitized with the agents and the memory channels within each cluster.

  This requires full DIMM symmetry (left-to-right and top-to-bottom). If there is not full DIMM symmetry, SNC will be forced to Disabled.

  When SNC4 is selected with XCC processors, the OS sees four NUMA nodes per physical CPU package.

UMA-Based Clustering

UMA-Based Clustering options are only available when SNC is Disabled.

This setting is accessed as follows:

- System setup: System Settings → Processors → UMA-Based Clustering
- OneCLI/ASU variable: `Processors.UMA-BasedClustering`
- Redfish attribute: `Processors_UMA-BasedClustering`

Possible values are:

- **Disabled**
  No clustering and no LLC to memory controller affinitization.
- **Hemisphere**
  Two clusters per socket that reduces latency due to proximity of LLC and memory controllers. Requires symmetry in memory configuration between the left side and the right side of the processor.

- **Quadrant (Default)**
  Works the same as Hemisphere but divides the die up into four clusters and only works on XCC processors. This mode affinitizes LLC slices to memory controllers based on proximity. Requires full symmetry in memory configuration between the left side and the right side, and between the top half and the bottom half of the processor.

Snoop Preference

Snoop mode determines the behavior in which a processor will validate an instruction or piece of data before performing an operation on it. The processor snoop function keeps caches coherent across the Intel processor interconnect fabric thus guaranteeing that data reads from cache or memory obtain the current copy of the data.

Snoop modes should be configured based on the specific instruction and memory access patterns of a particular workload. In general, low latency workloads tend to benefit from the Home Snoop w. Directory + OSB + HitME cache snoop mode, known as Home Snoop Plus, though the NUMA optimization of one's application should be evaluated.

Note: Setting the snoop mode preference does not always guarantee that it will be selected. The mode will be changed if the current hardware configuration does not support the desired mode

This setting is accessed as follows:

- System setup: **System Settings → Processors → Snoop Preference**
- OneCLI/ASU variable: `Processors.SnoopPreference`
- Redfish attribute: `Processors_SnoopPreference`

Possible values:

- **Home Snoop Plus** (Default)
  Home Snoop Plus = Home Snoop with Directory lookup + Opportunistic Snoop Broadcast (OSB) + HitME cache

  Best overall for most workloads. Speculative home snoop broadcasts are done under light UPI load conditions to avoid directory lookups from memory and reduce memory bandwidth. OSB is used only for local InvItoE (invalidate state to exclusive state) requests, which are generated due to full-line writes from the core or IO. Local InvItoE requests do not require a data read for this operation; hence the opportunistic broadcast feature is only used with writes to local memory to avoid memory access due to directory lookup. HitME cache is in the Caching and Home Agent (CHA) and caches directory information to speed up cache-to-cache transfers. HitME cache resources scale with number of CHAs in the system.
- **Home Snoop**
  Best for bandwidth sensitive workloads. Home snoop may increase performance on those workloads requiring maximum local and remote bandwidth for cache / memory lookups.

Intel Speed Select Technology

With Intel Speed Select Technology (SST), the rated frequency of the CPU can increase as the number of CPU cores that are enabled in UEFI goes down. Essentially, with SST, the CPU can achieve a guaranteed turbo frequency.

If a CPU is installed that does not support SST, the Base option will be used regardless of the setting selected. Config1/Config2 -force the SST cores limits based on the 'Config' option selected. Note Config1/Config2 may override the option that enables the number of CPU cores in UEFI.

For more information about how to configure Intel SST, see:
https://networkbuilders.intel.com/solutionslibrary/intel-speed-select-technology-performance-profile-intel-sst-pp-overview-user-guide

This setting is accessed as follows:

- System setup: **System Settings → Processor → Intel Speed Select Technology**
- OneCLI/ASU variable: Processors.IntelSpeedSelect
- Redfish attribute: Processors_IntelSpeedSelect

Possible values:

- **Auto** (Default)
  The level of SST enablement is controlled automatically based on the number of CPU cores enabled in UEFI.

- Base
  Effectively disables SST.
- **Config 1**
  Forces the number of enabled cores to the maximum number supported with Config1 for the CPU SKU installed.
- **Config 2**
  Forces the number of enabled cores to the maximum number supported with Config2 for the CPU SKU installed.
- **SST-PP V2**
  SST-PP V2 is a dynamic option and enables operating system level software control. It is not supported if CPU P-state Control is set to autonomous, legacy, or none for fourth generation Intel Xeon processors. CPU P-State Control must be set to Cooperative with Legacy or Cooperative without Legacy to expose this option.

C0 Nap Time

Controls maximum allowed time to nap in C0 sub-state, and to control whether C0.2 is supported. Default value indicates OS posed no limit on nap time.

This setting is accessed as follows:

- System setup: **System Settings → Processors → C0 Nap Time**
- OneCLI/ASU variable: `Processors.CoNapTime`
- Redfish attribute: `Processors_CoNapTime`

Possible values:

- <value>
  Where value is 0x0000_0000 to 0xFFFF_FFFF (default = 0x0000_0000)

C-state Interrupt Response Time

Controls the relative interrupt response time in C-states. Default value means the setting is not used.

This setting is accessed as follows:

- System setup: **System Settings → Processors → C-State Interrupt Response Time**
- OneCLI/ASU variable: `Processors.C-StateInterruptResponseTime`
- Redfish attribute: `Processors_ C-StateInterruptResponseTime`

Possible values:

- <value>
  Where value is 0x0000 to 0xFFFF (default = 0x0000)

PCH PCIe Relaxed Ordering

Enabling PCH PCIe Relaxed Ordering will always allow downstream completions to pass posted writes. Relaxed Ordering (RO) allows the system to relax some of the standard PCIe ordering rules.

Do not use Relaxed Ordering in the following cases:

- A shared memory architecture where more than one thread accesses the same locations in memory.
- A race condition exists in which a read to a location can occur before a previous write to that location completes.

This setting is accessed as follows:

- System setup: **System Settings → Processors → PCH PCIe Relaxed Ordering**
- OneCLI/ASU variable: `Processors.PCHPCIeRelaxedOrdering`
- Redfish attribute: `Processors_PCHPCIeRelaxedOrdering`

Possible values:

- **Enabled** (Default)
- **Disabled**

CPU PCIe Relaxed Ordering

Enabling CPU PCIe Relaxed Order will always allow downstream completions to pass posted writes.

This setting is accessed as follows:

- System setup: **System Settings → Processors → PCH PCIe Relaxed Ordering**
- OneCLI/ASU variable: `Processors.CPUPCIeRelaxedOrdering`
- Redfish attribute: `Processors_CPUPCIeRelaxedOrdering`

Possible values:

- **Enabled**
- **Disabled** (Default)

Intel Virtualization Technology

Intel Virtualization Technology (VTx) is a set of technologies within a system that accelerate virtualization instructions by carrying them out directly in the silicon rather than as code to be binarily translated from a Guest OS, through a Hypervisor to a physical server platform.

VTx must be enabled when a system will be used for virtualization. For general business applications, VTx has little to no performance impact. In those scenarios when high frequency low latency code execution is paramount, VTx should be disabled. Financial, scientific and HPC applications often benefit from disabling VTx.

This setting is accessed as follows:

- System setup: **System Settings → Processors → Intel Virtualization Technology**
- OneCLI/ASU variable: `Processors.IntelVirtualizationTechnology`
- Redfish attribute: `Processors_IntelVirtualizationTechnology`

Possible values:

- **Enabled** (Default)
- **Disabled**

Settings for Memory

Settings for Memory:

- ADDDC Sparing
- Socket Interleave
- Patrol Scrub
- Memory Data Scrambling

ADDDC Sparing

ADDDC (Adaptive Double DRAM Device Correction) Sparing is a RAS function that provides more reliability of memory error correction in virtual lockstep mode. When the memory correctable error count reaches a pre-defined threshold, ADDDC will trigger virtual lockstep mode, which can significantly reduce performance.

For high frequency, low latency, or low jitter workloads, Lenovo generally recommends disabling ADDDC sparing for high-performance.

ADDDC supported with x4 DRAM only: This setting is not available with x8 DIMMs.

This setting is accessed as follows:

- System setup: **System Settings → Memory → ADDDC Sparing**
- OneCLI/ASU variable: `Memory.ADDDCSparing`
- Redfish attribute: `Memory_ADDDCSparing`

Possible values:

- **Disabled** (Default)
  It is static Disable when “Page Policy” is Adaptive.
- **Enabled**
  Enable ADDDC Sparing to increase memory reliability.

Socket Interleave

Socket interleave determines how the memory map will be laid out within the system. Memory is either laid out such that each CPU has a map of local attached memory (NUMA) or in a flat memory model with no NUMA nodes (Non-NUMA).

Under most circumstances, customers should leave this setting at the default. Only in certain legacy applications which do not handle NUMA memory should this be changed.

This setting is accessed as follows:

- System setup: **System Settings → Memory → Socket Interleave**
- OneCLI/ASU variable: Memory.SocketInterleave
- Redfish attribute: Memory_SocketInterleave

Possible values:

- **NUMA** (Default)
  Each CPU socket is interleaved into separate NUMA nodes, improves average latency to LLC and memory.
- **Non-NUMA**
  Memory mapping is a flat model. BIOS will attempt to interleave all sockets into one NUMA node. If an OS or system application is not NUMA-aware, user may need to choose this option.

Patrol Scrub

Memory scrubbing technology runs at the hardware level to prevent the accumulation of correctable errors within a block of memory that could result in uncorrectable errors. Patrol scrub works in the background to proactively check DIMMs for memory errors. Enabling patrol scrub will negatively impact performance per watt.

Memory scrubbing is strongly encouraged to be implemented on production systems, as it will aid in maintaining the resiliency of the memory subsystem. Patrol scrub will provide the most consistent verification, though at the cost of a slight loss of memory bandwidth and increase in latency. In those workloads where memory latency is vital to application performance customers may elect to disable all memory scrubbing features to gain additional performance.

This setting is accessed as follows:

- System setup: **System Settings → Memory → Patrol Scrub**
- OneCLI/ASU variable: `Memory.PatrolScrub`
- Redfish attribute: `Memory_PatrolScrub`

Possible values:

- **Enabled** (Default)
  Memory patrol scrubbing is enabled with 24 hours interval. If the correctable error count reaches the error threshold, then the issuing memory page would be retired with an error record in XCC.
- **Disabled**
  Memory patrol scrubbing is disabled.

Memory Data Scrambling

Memory traffic on the data bus is not random and can cause current “hot spots” on the DIMM. Memory data scrambling uses a data scrambling feature in the memory controller to create pseudo-random patterns on the DDR4 data bus to reduce possibility of data-bit errors due to the impact of excessive current fluctuations.

Lenovo recommends leaving this parameter enabled to avoid data-bit errors. Recommended for all workloads.

This setting is accessed as follows:

- System setup: **System Settings → Memory → Memory Data Scrambling**
- OneCLI/ASU variable: `Memory.MemoryDataScrambling`

- Redfish attribute: `Memory_MemoryDataScrambling`

Possible values:

- **Enabled** (Default)
- **Disabled**

Settings for Power

Settings for Power:

- Workload Configuration
- PCIe Power Brake
- ASPM (PCIe Active State Power Management)

Workload Configuration

Workload configuration bias is used the tune the system's I/O bandwidth profile. This setting tunes the how aggressively the system will allocate processor core and uncore frequency to handle I/O requests.

I/O sensitive workloads will benefit from this setting as it enables higher I/O bandwidth. Lenovo recommends using I/O sensitive for low latency workloads and to evaluate the impact for high frequency and low jitter workloads.

This setting is accessed as follows:

- System setup: **System Settings → Power → Workload Configuration**
- OneCLI/ASU variable: `Power.WorkloadConfiguration`
- Redfish attribute: `Power_WorkloadConfiguration`

Possible values:

- **Balanced** (Default)
  CPU core and uncore frequency is balanced to provide equal performance weighting between I/O tasks and application workload threads.
- **I/O sensitive**
  CPU core and uncore frequency is weighted to allocate enough resources to provide high I/O bandwidth when CPU cores are at low utilization.

PCIe Power Brake

PCIe Power Brake quickly reduces the power consumption and performance of high powered PCIe devices (e.g.: GPU, FPGA). Performance of PCIe devices that are low power are not impacted by this setting. A high powered PCIe device is one that is rated at 60W TDP or greater.

This setting is accessed as follows:

- System setup: **System Settings → Power → PCIe Power Brake**
- OneCLI/ASU variable: `Power.PCIePowerBrake`
- Redfish attribute: `Power_PCIePowerBrake`

Possible values:

- **Reactive**
  The system performs PCIe throttling when a PSU indicates a momentary overconsumption or high temperature warning is occurring. The system also proactively performs PCIe power throttling to thermally protect high powered PCIe devices in unsupported high ambient temperatures and during loss of cooling redundancy.

  The high temperature warning setting is different among different platforms (typical setting is 35°C degree), and it would be adjusted to a lower value to avoid the loss of cooling redundancy.

- **Proactive** (Default)
  The system performs PCIe throttling based on the maximum power rating of the installed high power PCIe adapters. High powered PCIe device performance is reduced when the total power from high powered PCIe devices is greater than 1/3 of the sum of active power supply (PSU) wattages. Proactive mode also includes the PCIe throttling features of Reactive mode.

  To de-assert PCIe throttling for this case, user needs to remove some GPUs to down the total TDP of GPU, or replace a proper higher watt PSU.

- **Disabled**
  The system will not perform PCIe throttling. Proactive thermal protection provided under Proactive/Reactive modes are not operational. PCIe throttling is limited to the self-throttling capabilities of the high powered PCIe devices.

ASPM (PCIe Active State Power Management)

ASPM is a PCIe power saving feature. It puts the PCIe link into a low power mode when the link is idle. Two low power "standby" Link states are defined for ASPM – L0 low power Link state and L1 Link state:

- The L0s low power Link state is optimized for short entry and exit latencies, while providing substantial power savings. If there is a retimer on the link path, L0s should be disabled.
- The L1 Link state is optimized for maximum power savings at a cost of longer entry and exit latencies. L1 reduces Link power beyond the L0s state for cases where very low power is required and longer transition times are acceptable.

There are two pitfalls to bear in mind when configuring ASPM:

- Some older PCIe devices did not support L0s for both transmit and receive cycles. There is a newer PCIe engineering change notice (ECN) that now requires software to only enable L0s, if both ends of the link support L0s. L1 already required both ends to be in L1 and software has to check it. The ECN now also allows hardware to support L1 state with support for L0s.
  Members of the PCI SIG can view more details on the change at:
  http://www.pcisig.com/specifications/pciexpress/specifications/ECN_ASPM_Optionality_2009-08-20.pdf

- Under Linux, the ASPM policy can be set either by writing to a sysfs file/object (/sys/modules/pcie_aspm/parameters/policy) or from the Linux kernel command line (boot with pcie_aspm.policy=X). In either case, the valid values for "policy" are powersave, performance, and default. When default is chosen, it means Linux will use the policy specified by the platform firmware.
  The pcie_aspm option simply provides runtime control for disabling/enabling ASPM. There are 2 valid options for pcie_aspm: off and force. The default setting for ASPM in the Linux kernel is enabled.

  The 2 values for pcie_aspm are:

  - off – disables the ASPM in OS
  - force – acts an override to platform firmware settings. In other words, this forces ASPM to be enabled, regardless of how the platform firmware settings are configured.

Lenovo recommends disabling ASPM for all systems where workload performance is more important than power savings. Meanwhile, after enabling ASPM, some PCIe devices could encounter uncorrectable errors.

This setup option is available starting with Intel platform based ThinkSystem V2 products.

This setting is accessed as follows:

- System setup: **System Settings → Power → ASPM**
- OneCLI/ASU variable: `Power.ASPM`
- Redfish attribute: `Power_ASPM`

Possible values:

- **Auto**
  Enable ASPM on PCIe endpoint adapters that support it.

- **Disabled** (Default)
  Disable ASPM for all PCIe endpoints. User needs to disable ASPM if problems occur.

Settings for Devices and I/O Ports

Settings for Devices and I/O Ports:

- Intel Virtualization Technology for I/O (VT-d)

Intel Virtualization Technology for I/O (VT-d)

Intel Virtualization Technology for I/O (VT-d) includes four key capabilities:

- I/O device assignment. This feature allows an administrator to assign I/O devices to VMs in any desired configuration.
- DMA remapping. Supports address translations for device DMA data transfers.
- Interrupt remapping. Provides VM routing and isolation of device interrupts.
- Reliability features. Reports and records system software DMA and interrupt errors that may otherwise corrupt memory and impact VM isolation.

VT-d is used in Virtualization environments. For low latency and low jitter environments, we recommend disabling VT-d.

This setting is accessed as follows:

- System setup: **System Settings → Devices and I/O Ports → Intel VT for DirectedI/O (VT-d)**
- OneCLI/ASU variable: `DevicesandIOPorts.IntelVTforDirectedIOVTd`
- Redfish attribute: `DevicesandIOPorts_IntelVTforDirectedIOVTd`

Possible values:

- **Enabled** (Default)
- **Disabled**

Hidden UEFI Items

The following UEFI items are more limited in their applicability to customer use cases and are not exposed in UEFI menus but can be accessed using the command line utilities such as Lenovo's Advanced Settings Utility (ASU) or OneCLI.

No Redfish support: These UEFI items also have no Redfish attribute and cannot be accessed via the Redfish interface.

The UEFI menu items described here are as follows:

- PackageCstate var
- UPI Gate Disable
- Isochronous Mode
- Maximum P-state
- CR QoS Configuration and CR FastGo Configuration
- Local/Remote Threshold
- Stale AtoS
- Snoop Response Hold Off
- LLC dead line allocation
- Uncore Frequency Limit

PackageCstate var

Enables / Disables package C-states. Not supported on 8-socket platform starting with 4th Gen Intel Xeon Scalable processor.

This setting is accessed as follows:

- OneCLI/ASU option: `Processors.PackageCstate`

Possible Values:

- **Disable**
  Disable package C-states.

- **Enable** (Default)
  Enable package C-states and set limit to PC6 with retention
- **PC6NR**
  Enable package C-states and set limit to PC6 with no retention
- **PC2**
  Enable package C-states and set limit to PC2

UPI Gate Disable

Prohibits / allows clock gating on the UPI links. The setting is negative logic so when the ASU option is enabled, it is actually disabling UPI clock gating. When disabled, it allows clock gating on the UPI links.

This setting is accessed as follows:

- OneCLI / ASU option: `Processors.UPIGateDisable`

Possible Values:

- **Enable** (Default)
- **Disable**

Isochronous Mode

Isochronous can be enabled to support QoS and latency requirement. It should be left disabled for maximum bandwidth.

This setting is accessed as follows:

- OneCLI / ASU option: `Processors.Isochronous`

Possible Values:

- **Enabled**
- **Disabled** (Default)

Maximum P-state

Maximum P-state limits the P-state to the maximum number specified (P0=fastest, Pn=slowest frequency) supported by the installed CPU SKU. If turbo is enabled and the maximum P-state is set to a number higher than P0, turbo will never be used. When a P-state limit is set, some OS schedulers may further restrict the final P-state value that is set. This is mainly dependent on the power plan selected under the OS.

This setting is accessed as follows:

- OneCLI / ASU option: `Processors.MaxPstate`

Possible Values:

- **P0** (Default)
- **P1**
- …
- **Pn**

CR QoS Configuration and CR FastGo Configuration

One of 4 variables (IRQThreshold, StaleAtoS, CRQoSConfiguration/CRFastGoConfiguration, L2RFOPrefetchDisable) used to optimize performance for SAP HANA on servers with 2-hop memory configurations such as 4-socket ring, 6-socket, and 8-socket configurations.

The default value of **Auto** corresponds to a setting of **Option 5** when a 2-hop memory configuration is detected. Option 5 makes the L2 prefetcher less aggressive and lowers NT (Non-Temporal) write bandwidth. This has the effect of limiting bursty local and remote traffic.

This setting is accessed as follows:

- OneCLI/ASU variable: `Memory.CRQoSConfiguration` or `Memory.CRFastGoConfiguration` with current UEFI builds. Some older UEFI builds may still use the variable name `Memory.CRQoSConfiguration`.

Possible values:

- **Auto** (Default)
  The **Auto** value corresponds to **Option 5** for 2-hop memory configurations and **None** for all other configurations.
- **None**
  No optimization
- **Option 1**
- **Option 2**
- **Option 3**
- **Option 4**
- **Option 5**
  Higher option number makes L2 prefetcher less aggressive and lowers NT write bandwidth.

Local/Remote Threshold

In multi-socket environments that generate remote traffic, the Table Of Requests (TOR) services both remote and local requests across the Intel UPI links. Remote reads allocate the TOR through the RRQ (Remote Request Queue) ingress and typically send out a read request to the memory controller. When the data returns from the memory controller, the TOR will send a data response back to the requesting socket via the Intel UPI links.

The rate at which these requests and responses can be sent out is governed by Intel UPI bandwidth via credits. The rate is adjustable to limit the number of incoming transactions to avoid overloading a remote socket. This bandwidth is substantially less than local memory bandwidth or TOR service bandwidth.

When the Intel UPI link is the bottleneck, these remote reads stay in the TOR for a long time. When one socket's TOR is saturated with remote reads, it cannot service its own local requests well enough. The other socket is relatively free to continue sending remote requests to the saturated socket, which may cause an imbalance. If the imbalance shifts around then the performance averages out over time and can converge to a balance, but if a socket is saturated, and continues to get requests from the better performing socket, it may not build enough momentum to overturn the imbalance in its favor. Application workloads with UMA and remote traffic mixes are likely candidates for this imbalance and could affect overall performance.

The UEFI item called *local/remote threshold* is available to control this rate and help avoid the imbalance. This option controls two features simultaneously:

- RRQ (Remote Request Queue) throttling limits remote reads in the TOR by specifying a threshold.
- IRQ (Incoming Request Queue) throttling limits the number of local-to-remote reads by specifying a threshold (the requests are from IRQ, but the home node of the requested address is a remote socket).

These features are used to control bandwidth of a multi-socket system, to maximize the throughput of each single socket while trying to make the performance across sockets as balanced as possible. The default threshold values set by BIOS are defined based on internal experiments and are believed to be optimal for typical workloads, but other options are provided to customers if their workload is showing noticeably different behavior from the internal workloads.

The threshold values set by the UEFI are dependent on the system configuration. The UEFI will first detect the platform configuration in terms of the number of sockets and the number of Intel UPI links and then automatically set the default mode accordingly.

To simplify the UEFI setting, the two features RRQ throttling and IRQ throttling are a unified option and the value of each option, based on the number of processors installed and the value of IrqThreshold selected, is listed in the following table.

Table 3. Setting RRQ throttling and IRQ throttling using the IrqThreshold parameter

Value of IrqThreshold	1 socket (1S)	2S/4S-Mesh	4S Ring	8S
Auto (default)	Disabled mode	Medium mode	Medium mode	Medium mode
Disabled	RRQ=Disabled, IRQ=Disabled	RRQ=Disabled, IRQ=Disabled	RRQ=Disabled, IRQ=Disabled	RRQ=Disabled, IRQ=Disabled
Low Threshold	Not applicable	RRQ=6, IRQ=Disabled	RRQ=7, IRQ=2	RRQ=7, IRQ=2
Medium Threshold	Not applicable	RRQ=7, IRQ=Disabled	RRQ=7, IRQ=7	RRQ=7, IRQ=7
High Threshold	Not applicable	RRQ=7, IRQ=Disabled	RRQ=9, IRQ=10	RRQ=8, IRQ=10

This setting is accessed as follows:

- OneCLI/ASU variable: Processors.IrqThreshold

Possible values (see the above table):

- **Auto** (Default)
- **Disabled**
- **Low**
- **Medium**
- **High**

Stale AtoS

Stale AtoS is one of four variables (IRQThreshold, StaleAtoS, CRQoSConfiguration/ CRFastGoConfiguration, L2RFOPrefetchDisable) used to optimize performance for SAP HANA on servers with 2-hop memory configurations such as 4-socket ring, 6-socket and 8-socket configurations.

The in-memory directory within the Intel memory controller that tracks the usage state of all cache lines has three states: I, A and S:

- I (Invalid) state means the data is clean and does not exist in any other socket's cache.
- A (snoop All) state means the data may exist in another socket in exclusive or modified state.
- S (Shared) state means the data is clean and may be shared across one or more socket's caches.

When doing a read to memory, if the directory line is in the A state we must snoop all the other sockets because another socket may have the line in the modified state. If this is the case, the snoop will return the modified data. However, it may be the case that a line is read in A state and all the snoops come back a miss. This can happen if another socket read the line earlier and then silently dropped it from its cache without modifying it.

If Stale AtoS feature is enabled, in the situation where a line is in A state returns only snoop misses, the line will transition to S state. That way, subsequent reads to the line will encounter it in S state and not have to snoop, saving latency and snoop bandwidth.

Stale AtoS may be beneficial in a workload where there are many cross-socket reads.

This setting is accessed as follows:

- OneCLI/ASU variable: `Processors.StaleAtoS`

Possible values:

- **Auto** (Default), which corresponds to:
  - **Disabled** when 2 CPUs are installed
  - **Enabled** if more than 2 CPUs are installed
- **Disabled**
- **Enabled**

Snoop Response Hold Off

Set the I/O delay in response to snoop request. This parameter is used to fix a low InfiniBand performance problem. This hidden parameter is accessible on ThinkSystem servers that use Intel Xeon Scalable processors.

For some workloads in which throughput and latency are critical, it is better to constantly poll the status of an I/O device rather than use an interrupt. Network adapter device drivers commonly use a thread to continuously poll in a fast loop so that incoming requests can be handled as fast as possible.

Continuous polling can create contention between a processor core running the polling thread and the processor's Integrated I/O feature (IIO) for an I/O-owned line in cache. This contention can cause an I/O operation to lose ownership of the cache line it has just acquired. It must then spend more time reacquiring the cache line to write it back.

When there are a large number of network ports each servicing small packets, the system may not be able to achieve the full throughput required due to excessive I/O and core contentions of cache lines. For this situation, the I/O operation should delay its response to core snoops and hold onto its cache lines until it successfully completes its write.

The Snoop Response Hold Off parameter allows the I/O operation to delay its snoop response by a selected amount to achieve this delay.

This setting is accessed as follows:

- OneCLI/ASU variable: `Processors.SnoopResponseHoldOff`

Possible values:

- 0, 1, 2 ... F
  Set the Hold Off IO delay value. The value entered is a hex digit and the valid range is 0 to F. The default (initial value) is 9 for ThinkSystem servers. The value corresponds to a number cycles as shown in Table 4. The cycles in the table are IIO clock cycles which are 2 ns per cycle. For Intel Omni-Path network adapter use, setting this parameter to 9 is recommended as a starting point. Network performance tests should be performed to determine the most optimal value for each workload.

Table 4. Possible values of the parameter in the Processors.SnoopResponseHoldOff command

Value	Number of cycles (1 cycle = 2 nanoseconds)
0	Disabled
1	8
2	16
3	32
4	64
5	128
6	256
7	512
8	1K
9	2K
A	4K
B	8K
C	16K
D	32K
E	64K
F	128K

LLC dead line allocation

When enabled, opportunistically fill dead lines in LLC. In the Intel Xeon Scalable family non-inclusive cache scheme, MLC evictions are filled into the LLC. When lines are evicted from the MLC, the core can flag them as "dead" (in other words, not likely to be read again). The LLC has the option to drop dead lines and not fill them in the LLC. If the Dead Line LLC Allocation feature is disabled, dead lines will always be dropped and will never fill into the LLC.

Not filling the LLC with dead lines can help save space in the LLC and prevent the LLC from evicting useful data. However, if the Dead Line LLC Allocation feature is enabled, the LLC can opportunistically fill dead lines into LLC if there is free space available.

This setting is accessed as follows:

- OneCLI/ASU variable: `Processors.LLCdeadlinealloc`

Possible values:

- **Enabled** (Default)
- **Disabled**
- **Auto**
  Map to Enabled.

Uncore Frequency Limit

When uncore frequency scaling is enabled (see the Uncore Frequency Scaling section), the Uncore Frequency Limit setting is used to set the upper frequency limit of the uncore in the CPU.

Note that this setting is not available if Uncore Frequency Scaling is disabled. When Uncore Frequency Scaling is disabled, the uncore runs at the maximum frequency all the time.

This setting is accessed as follows:

- OneCLI/ASU variable: `Processors.UncoreFrequencyLimit`

Possible values:

- Hex value 0, 1, 2 ... 3F
  Specifies the CPU uncore frequency limit value. The value is entered as a hex value in the range of 0 to 3F. The default value is 3F (maximum). The value, when converted to decimal, corresponds to 1/10th of a GHz. For example, 3F is 63 in decimal and corresponds to a limit of 6.3 GHz, and a value of 14 hex (20 in decimal) corresponds to a limit of 2.0 GHz.

  Do not include any prefix or suffix when entering the hex value.

  The default value of 3F (63 in decimal, resulting in a limit of 6.3 GHz) is the maximum possible value that can be set. However, if a value is set that is higher than what the installed CPU SKU supports, the CPU will clip the value to the highest supported value. For example, if uncore frequency limit is set to 3F (6.3 GHz) but the CPU SKU has a maximum uncore frequency of 2.0 GHz, then the limit will be clipped to 2.0 GHz.

Red Hat Enterprise Linux (RHEL) and derivatives

The following are general considerations for optimizing systems for low jitter / low latency under Red Hat Enterprise Linux / CentOS Linux:

- Disable unnecessary system services
- Avoid non-data disk access by:
  - Consider disabling filesystem journaling
  - Consider running w/o swap partition / swap file
- For a quick set of OS level tunings for low latency / high frequency workloads consider running tuned-adm profile latency-performance
- Disable power management in both kernel and UEFI
  - Use powertop or turbostat to see if there are P-state and C-state transitions
  - Set both intel_idle.max_cstate=0 and processor.max_cstate=n (n = 0 to 3)
    CAUTION: `intel_idle.max_cstate=0` does not mean maximum state=0 (zero). This option disables `intel_idle`.
  - If you set `intel_idle.max_cstate=n` (n= 1 to 6), then `intel_idle` is enabled and maximum cstate is set to n.
- Consider using the deadline elevator for I/O tuning
  `ELEVATOR = "deadline"`
- Enforce NUMA locality for applications / threads:
  `numactl -C1 -m1 ./command`

  numad user-level daemon (new since RHEL 6.3 and disabled by default) attempts automatically co-locate a process and its memory
- Consider using control groups (Cgroups) for CPU/Memory/Network/Disk to manage multi-application and / or large NUMA machines
- Make sure adapters are using MSI-X for IRQ deployment For example, use the following and then look for PCI-MSI-X in the output
  `cat /proc/interupts | grep eth`
- Consider pinning adapter IRQs on large NUMA systems (requires deep knowledge of systems and applications – not for the faint of heart). This requires disabling irqbalance daemon
- Disable usb0 to avoid USB device polling and unplug and USB devices (keyboards, mice, USB keys, etc)

Additional References

The following resources provide additional information:

- Optimizing RHEL 9 for Real Time for low latency operation
  https://access.redhat.com/documentation/en-us/red_hat_enterprise_linux_for_real_time/9/html/optimizing_rhel_9_for_real_time_for_low_latency_operation/index
- 4th Generation Intel Xeon Scalable Processors
  https://ark.intel.com/content/www/us/en/ark/products/series/228622/4th-generation-intel-xeon-scalable-processors.html

About the author

Charles Stephan is a Senior Engineer and Technical Lead for the System Performance Verification team in the Lenovo Performance Laboratory at the Lenovo Infrastructure Solutions Group (ISG) campus in Morrisville, NC. His team is responsible for analyzing the performance of storage adapters, network adapters, various flash technologies, and complete x86 platforms. Before transitioning to Lenovo, Charles spent 16 years at IBM as a Performance Engineer analyzing storage subsystem performance of RAID adapters, Fibre Channel HBAs, and storage servers for all x86 platforms. He also analyzed performance of x86 rack systems, blades, and compute nodes. Charles holds a Master of Science degree in Computer Information Systems from the Florida Institute of Technology.

Related product families

Product families related to this document are the following:

- Processors
- ThinkSystem SD650 V3 server
- ThinkSystem SD650-I V3 server
- ThinkSystem SD650-N V3 Server
- ThinkSystem SR630 V3 Server
- ThinkSystem SR650 V3 Server
- ThinkSystem SR850 V3 Server
- ThinkSystem SR860 V3 Server
- ThinkSystem SR950 V3 Server
- ThinkSystem ST650 V3 Server

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.
8001 Development Drive
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, LP1836, was created or updated on November 16, 2023.

Send us your comments in one of the following ways:

- Use the online Contact us review form found at:
  https://lenovopress.lenovo.com/LP1836
- Send your comments in an e-mail to:
  comments@lenovopress.com

This document is available online at https://lenovopress.lenovo.com/LP1836.

Trademarks

Lenovo and the Lenovo logo are trademarks or registered trademarks of Lenovo in the United States, other countries, or both. A current list of Lenovo trademarks is available on the Web at https://www.lenovo.com/us/en/legal/copytrade/.

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

The following terms are trademarks of other companies:

AMD and AMD EPYC™ are trademarks of Advanced Micro Devices, Inc.

Intel® and Xeon® are trademarks of Intel Corporation or its subsidiaries.

Linux® is the trademark of Linus Torvalds in the U.S. and other countries.

Windows Server® and Windows® are trademarks of Microsoft Corporation in the United States, other countries, or both.

SPECpower® is a trademark of the Standard Performance Evaluation Corporation (SPEC).

C3® is a trademark of IBM in the United States, other countries, or both.

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