Implementing CXL Memory on Linux on ThinkSystem V4 Servers

Planning / Implementation

In this section, we introduce the support for CXL technology in Lenovo ThinkSystem V4 servers, which includes three types of devices and three types of protocols. This white paper focuses on the support for CXL memory devices in Lenovo V4 servers, with a detailed discussion on CXL memory devices provided in the next section.

CXL is an alternate set of protocols that run across the standard PCIe physical layer that can auto-negotiate to either the standard PCIe transaction protocol or the alternate CXL transaction protocols, as shown in Figure 1. It is a high-performance I/O bus architecture used to interconnect peripheral devices, either traditional non-coherent IO devices or accelerators with additional capabilities.

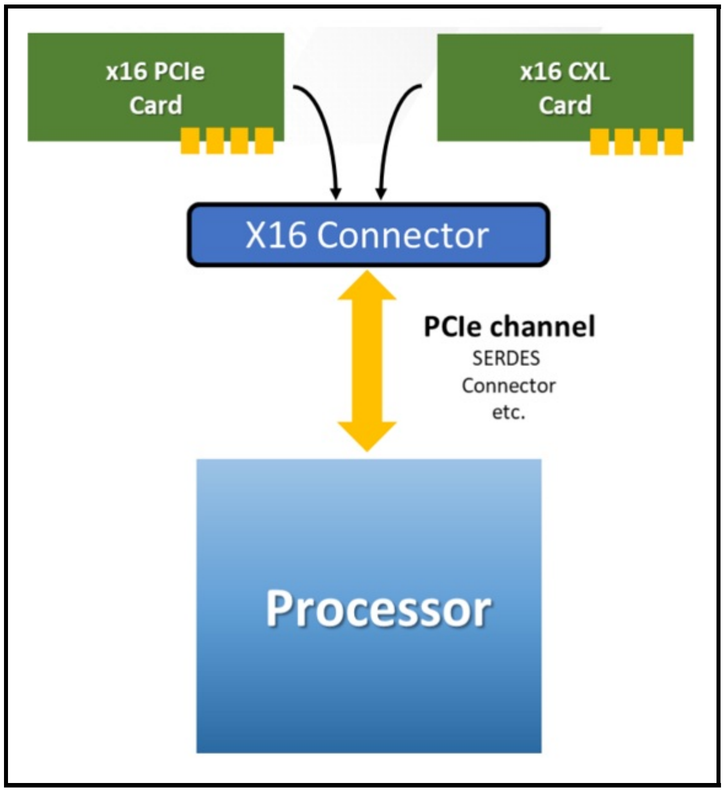


Figure 1. CXL runs on the standard PCIe physical layer but with CXL transaction protocols

CXL currently supports three types of devices, including type 1 caching devices, type2 accelerators with memory, and type 3 memory buffers (devices and modules), as showed in Figure 2. This white paper focuses on the type 3 memory devices.

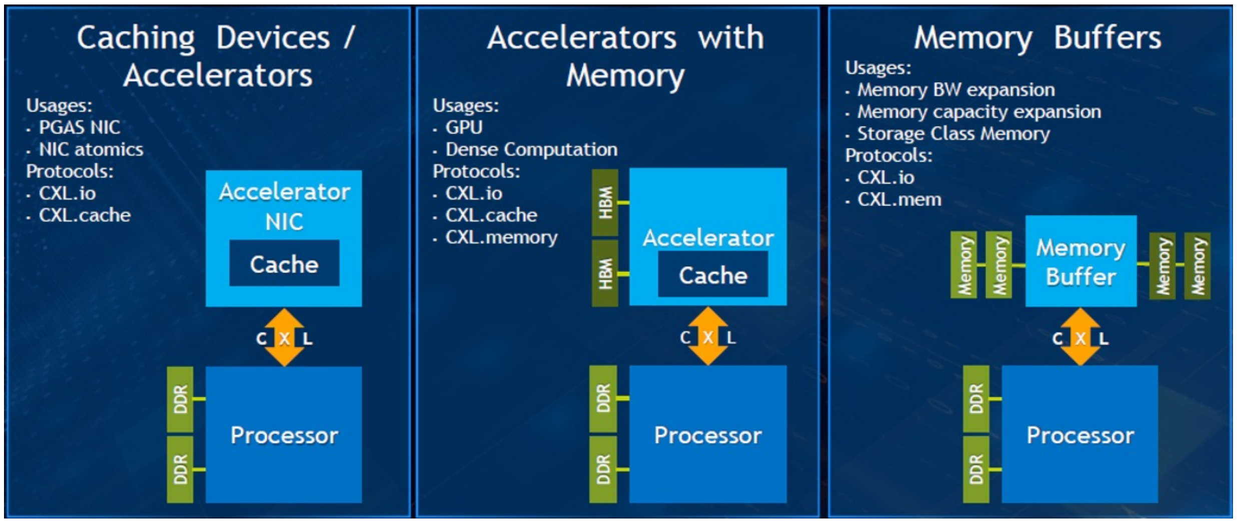


Figure 2. CXL supports three types of devices and three types of protocols

The three types of devices depend on the use cases and the transaction layers needed. These transaction layers are CXL.io, CXL.cache, and CXL.mem.

- The CXL.io is mandatory in all the CXL device types. It provides a non-coherent load/store interface for I/O devices. The CXL.io is not used for CXL.mem or CXL.cache protocol. For example, CXL.io is used for credit-based flow control, which is not used for CXL.cache or CXL.mem.
  CXL.io is always required for discovery and enumeration, error reporting, and Host-Managed Device Memory (HDM). CXL.io can also be used for transaction types, transaction packet formatting, credit-based flow control, virtual channel management and transaction ordering rules that follow the PCIe definition.
- CXL.cache protocol defines the interactions between the device and hosts as some requests. This is applied when use cases have special needs that make a fully coherent cache in the device valuable. Essential coherency allows an accelerator to implement any ordering mode and enables it to implement unlimited atomic operations.
- CXL.mem is the CXL Memory protocol, a transactional interface between the CPU and memory. It applies to different memory types (volatile, persistent, among others) and configurations (flat and hierarchical). This protocol supports various attach options, highlighting when the Memory Controller (MC) is in the Host CPU, when the MC is within an accelerator device, or when the MC is in the memory buffer chip.

CXL memory device support

In this section, we first discuss the architecture of the Lenovo ThinkSystem V4 servers from CXL memory device support perspective and then discuss this support from both hardware and software views from the Linux OS perspective.

- CXL device support introduction
- CXL device support: Hardware perspective
- CXL device support: Software perspective

CXL device support introduction

From the platform perspective, when a CXL device is attached to a CXL-capable PCIe bus, this PCIe bus will be switched to the Flex bus protocol by system firmware and runs the CXL.io, CXL.cache and/or CXL.mem protocols on top of Flex bus, as shown in Figure 3.

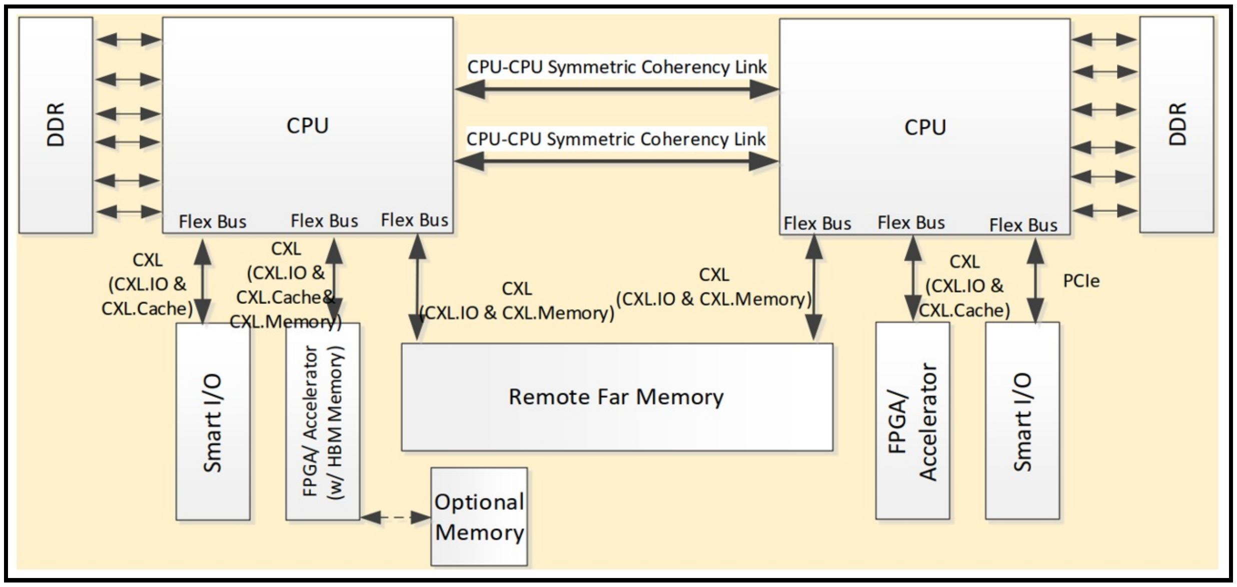

DDR
CPU
CPU-CPU Symmetric Coherency Link
CPU-CPU Symmetric Coherency Link
CPU
DDR
Flex Bus
Flex Bus
Flex Bus
Flex Bus
Flex Bus
Flex Bus
CXL (CXL.IO & CXL.Cache)
CXL (CXL.IO & CXL.Cache& CXL.Memory)
CXL (CXL.IO & CXL.Memory)
CXL (CXL.IO & CXL.Memory)
CXL (CXL.IO & CXL.Cache)
PCIe
Smart I/O
FPGA/ Accelerator (w/ HBM Memory)
Remote Far Memory
Optional Memory
FPGA/ Accelerator
Smart I/O

Figure 3. PCIe bus switches to Flex bus and runs the CXL.io, CXL.cache, and CXL.mem protocols

CXL device support: Hardware perspective

The right side of Figure 4 is the internal block diagram of a specific CXL memory device (memory expander). As can be seen in the block diagram of a CXL memory device, there is a CXL controller that is in charge of handling CXL protocols and data transactions.

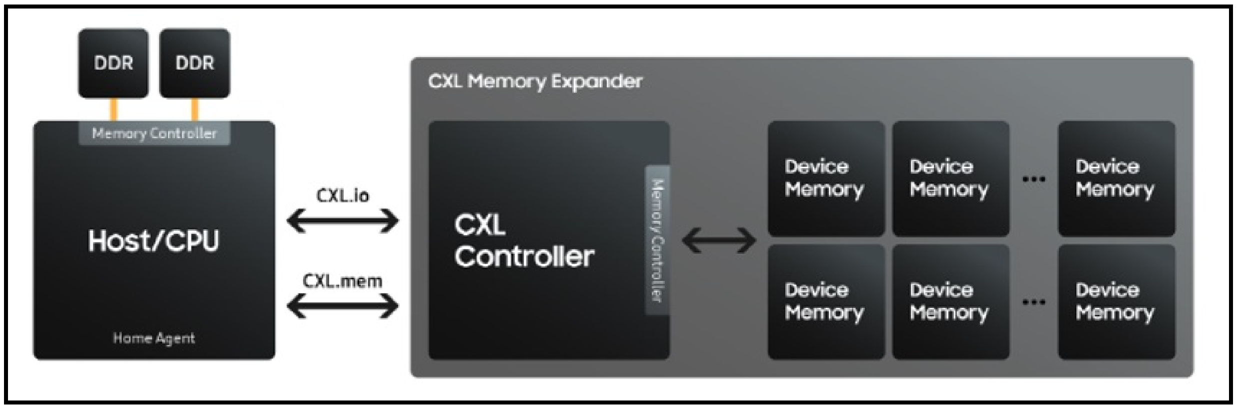


Figure 4. CXL memory device block diagram

There are several form factor implementations of CXL memory devices, as shown in Figure 5.

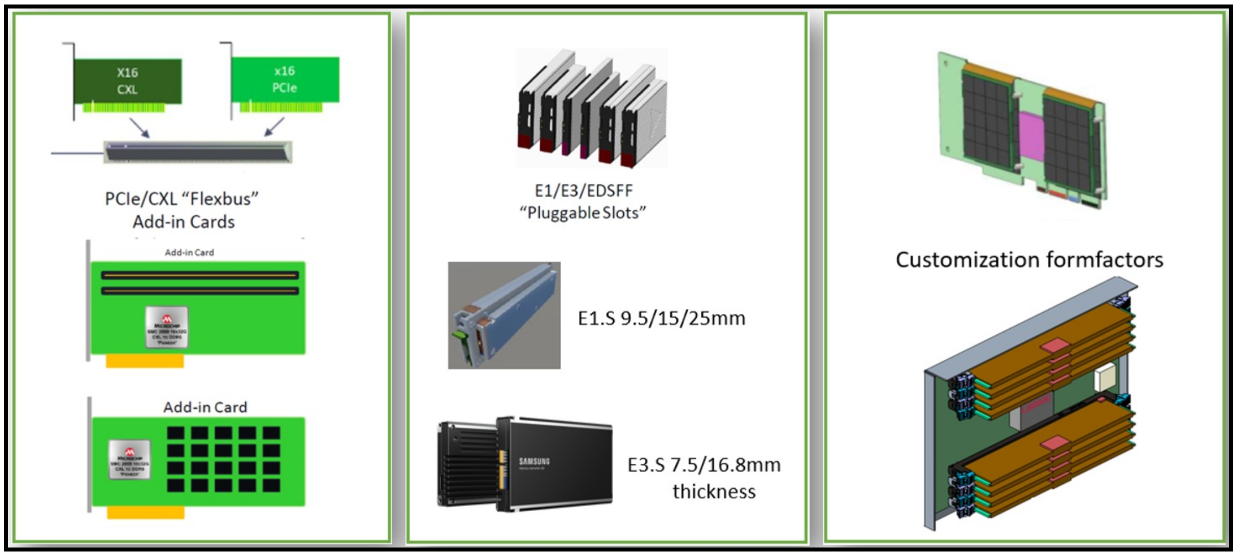

X16 CXL
x16 PCIe
PCIe/CXL "Flexbus" Add-in Cards
Add-in Card
Add-in Card
E1/E3/EDSFF "Pluggable Slots"
E1.S 9.5/15/25mm
E3.S 7.5/16.8mm thickness
Customization formfactors

Figure 5. Several CXL memory device form factor implementations

In this paper, we will install and validate E3.S CXL memory modules in the Lenovo ThinkSystem SR630 V4 server (as shown in Figure 6) with Ubuntu 24.04 LTS server version OS installed. For the details of setup and validation, see the "Enabling CXL on the ThinkSystem servers" and "CXL Memory Module Validation" sections in this white paper.

Figure 6. Lenovo ThinkSystem SR630 V4

The benefits with the CXL memory devices installed in the server system include:

- Increased Memory Capacity — Improve processor performance, faster execution and/or run more VMs/processes.
- Increased Memory Bandwidth — For machine learning or other workloads needing higher memory bandwidths.
- Memory Pooling — Allows for optimal provision of local DRAM on server motherboard.

CXL device support: Software perspective

In this section, we will focus on the CXL memory device support from the server system firmware (UEFI) and the Linux OS perspective.

In its most basic form, the system software component responsibilities are:

- The system firmware is responsible for enumerating and configuring volatile CXL memory capacity that is present at boot time.
- The Linux OS components are responsible for enumerating and configuring all topologies not covered by the system firmware.

For the CXL memory devices to be supported on the servers, the server system firmware will need to implement the Advanced Configuration and Power Interface (ACPI) standard that allows the CXL devices to be discovered and interact with the Linux OS. Major CXL-related ACPI tables and data structures and their functions are summarized in Table 1.

Table 1. CXL-related ACPI tables

ACPI Tables	**Descriptions**
SRAT	System Resource Affinity Table. Describes various NUMA domains in a system including host processors, accelerators, and memory.
SLIT	System Locality Information Table. Describes the relative distances between different NUMA nodes in the system. This information helps the operating system schedule tasks and allocate memory closer to the processors that need them, reducing access latency and improving performance.
HMAT	Heterogeneous Memory Attribute Table. Describes bandwidth and latency from any initiator (a processor or an accelerator) to any memory target.
CDAT	Coherent Device Attribute Table. This table is a data structure exposed by a coherent component and describes its performance characteristics.
CEDT	CXL Early Discovery Table. UEFI and OS drivers utilize to retrieve pointers to all of the CXL CHBS, a Set of Fixed Memory Windows (CFMWS) for each CXL host bridge present at platform boot time.
CHBS	CXL Host Bridge Structure. Each CXL Host Bridge instance will have a corresponding CHBS that identifies the version of the CXL host bridge supports and a pointer to the CXL root complex register block needed for programming the CXL root complex's HDM decoder.
CFMWS	CXL Fixed Memory Window Structure. A new structure type in CEDT describes all the platform allocated and programmed HPA based windows where the system firmware, UEFI, and OS drivers can map CXL memory.
ACPI0016	CXL Host Bridge Object. Virtual SW entity implemented by the system firmware under the _SB (System Bus) of the ACPI tree and consumed by the OS. Each ACPI0016 object represents a single CXL root complex.

For the Linux OS CXL support, CXL for memory devices utilizes the existing Linux NVDIMM architecture, replacing the NVDIMM Firmware Interface Table (NFIT) based interfaces with ACPI0017 CEDT based CXL interfaces, and adding a CXL subsystem to handle the new CXL specific functionality.

OS CXL components are introduced that utilize the information provided through the ACPI0017 CEDT in the OS layer and the ACPI0016 in the firmware layer. The Linux OS CXL components and their functions are summarized in Table 2 and Figure 7.

Table 2. Linux OS CXL components

Linux OS CXL components	Descriptions
CXL subsystem	It provides the CXL specific services for the NVDIMM BUS component, the CXL specific IOCTL and the SYSFS interfaces for management of CXL devices.
ACPI0017	It provides CXL information for the CXL kernel subsystem.
LIBNVDIMM BUS	Existing LIBNVDIMM BUS component. It provides generic NVDIMM bus related functionality including namespace management and enumerates memory device endpoints for the LIBNVDIMM REGION component.
LIBNVDIMM REGION	Existing LIBNVDIMM REGION component. It consumes endpoint memory devices produced by the LIBNVDIMM BUS and the region labels on each device (from the LSA). It also organizes the region labels into interleave sets, validates and publishes the regions to PMEM, DEVICE DAX and Block Translation Table (BTT) components.
PMEM	A kernel component that represents a single instance of a region.
DEVICE DAX	A simplified direct pipeline between the application and the persistent memory namespace that bypasses the filesystem and memory mapped file usage.
FS DAX	Direct memory access for memory mapped file systems.
BTT	A component that consumes the device block translation table and implements a block storage protocol on top of persistent memory.
PMEM NAMSPACE	Each region can be subdivided into volumes referred to as Namespaces. The configuration of each Namespace is described in the namespace labels stored in the LSA, which is exposed through the Command Interface.
CXL-CLI	Linux CXL memory management CLI.
LIBCXL	Linux CXL memory management library.
NDCTL	Linux NVDIMM memory management CLI utilized with CXL.
LIBNDCTL	Linux NVDIMM memory management library.

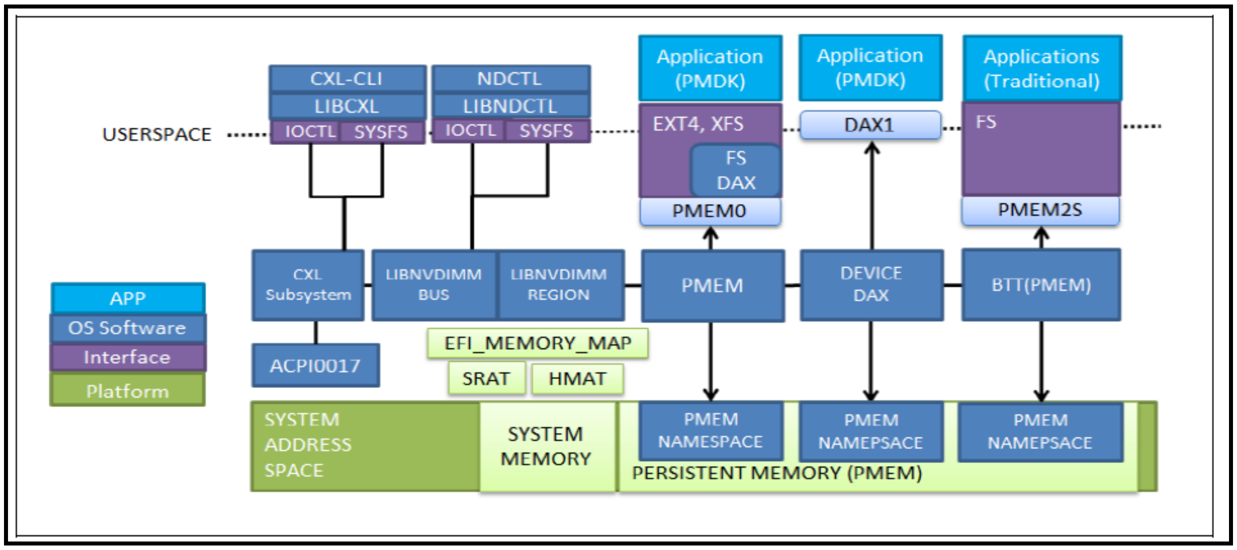


Figure 7. Linux CXL Architecture Diagram

For the CXL memory devices operation modes, an Intel processor attached with CXL memory expansion devices can support three types of CXL-attached memory operation modes: 1LM+Vol mode (One-Level Memory + Volatile mode, also referred to as memory tiering), Heterogeneous interleave mode, and Intel Flat Memory mode.

- 1LM+Vol mode

  1LM+Vol or memory tiering is the default memory mode and allows capacity expansion. This 2-tier memory mode implies the native DDR5 and CXL attached memory are different address spaces and separate NUMA nodes.

  This mode is also referred to as Software Managed tiering considering a software entity must manage the placement of data in separate tiers and must manage any desired movement of data between tiers. This management of placement and movement may be performed by the OS, or by a higher-level middleware or directly by an application.

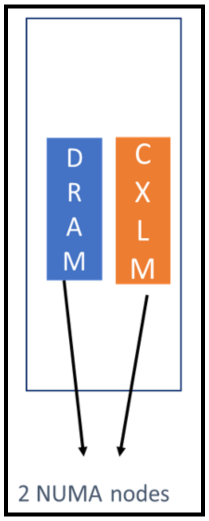
  

  Figure 8. CXL memory device configured to run in the 1LM+Vol mode

- Heterogeneous Interleave mode

  This mode supports memory interleave between CXL memory attached device and native DDR memory. Interleaving of memory requests across a combination of native attach DDR5 channels and CXL-connected memory to increase aggregate bandwidth.

  The entire combined capacity of native DDR memory and CXL is visible to the software as a single NUMA domain. As a result, no software changes are needed in the system to use the Heterogeneous Interleave mode.

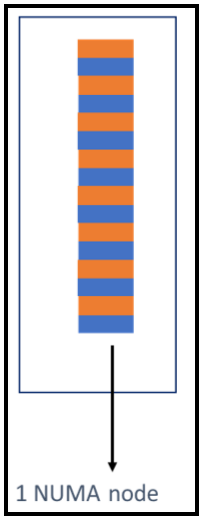


Figure 9. CXL memory device configured to run in the Heterogeneous Interleave mode

- Intel Flat Memory mode

  Intel Flat Memory mode uses logic in the processor to keep frequently used data in DRAM where performance is best and swapping out less frequently used data to CXL memory. The movement of data between two memory tiers is completely managed at the hardware level without any OS intervention.

  The CPU itself creates a single NUMA node and memory tier exposed to the OS, combining DRAM and CXL-attached memory in a 1:1 capacity ratio. Data is moved between DRAM and CXL memory at a cache line granularity. The memory controller will keep those cache lines accessed frequently in DRAM, and those cache lines that are not accessed frequently will be residing in CXL-attached memory. This process is completely transparent to the software.

  This feature eliminates the need for any OS or application-level software changes to augment server memory resources via CXL.

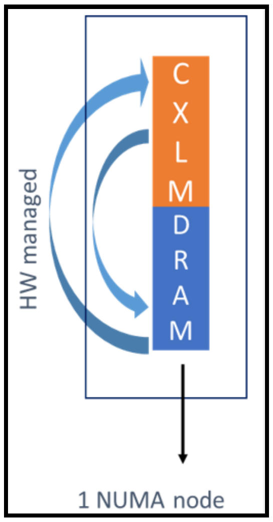


Figure 10. CXL memory device configured to run in the Intel Flat Memory mode

Note that not every memory population of the DRAM and CXL-attached memory combination supports all three CXL memory operation modes. For the specific CXL mode that is planned to run on a system, please check the server system specification for the required population rules and the required DRAM and CXL memory module specifications (such as memory frequency and capacity) for that system before the deployment.

In case you set the CXL memory modules to either Heterogenous Interleave mode or Intel Flat Memory mode but the installed DRAM and CXL memory modules do not meet the required specifications, the system firmware will disable these two modes and the CXL memory modules will fall back to 1LM+Vol mode.

Note: ThinkSystem SR630 V4 currently does not support Intel Flat Memory mode, as shown in Figure 12.

Enabling CXL memory

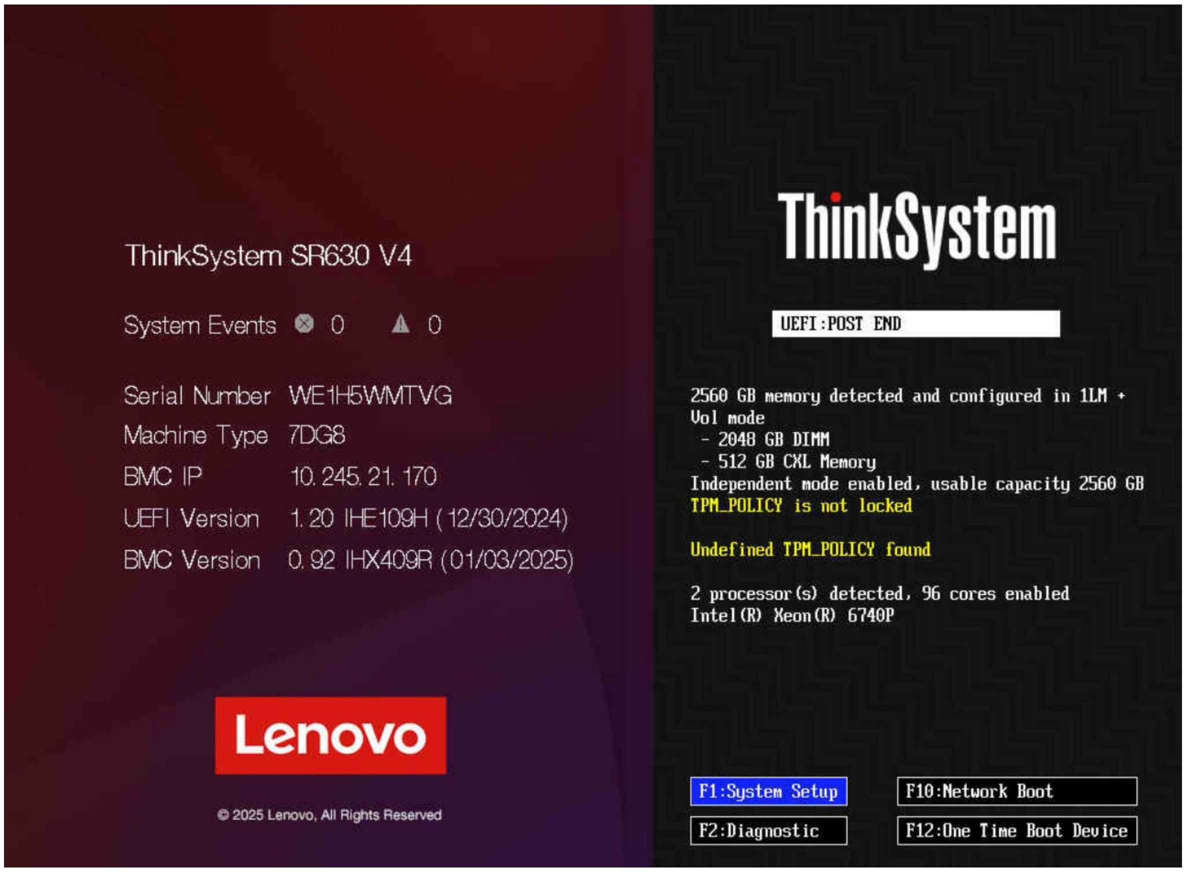
In this section, we will demonstrate how the CXL memory devices can be used with Lenovo ThinkSystem SR630 V4 server. As there will be only one NUMA node per CPU when the CXL memory devices are configured to run in either Heterogeneous Interleave mode or Intel Flat Memory mode (this mode is not supported by Lenovo V4 servers yet), we will focus on the setup of 1LM+Vol CXL memory mode which will have two NUMA nodes per CPU seen in the OS when correctly configured.

The configuration for this setup is:

- Platform: ThinkSystem SR630 V4
- CPU: Intel Xeon 6740P x 2
- Memory: DDR5 DIMM 128GB x 16
- CXL memory device: Samsung E3.S CXL memory module 128GB x 4
- OS: Ubuntu 24.04 LTS server version OS

The steps to configure the server UEFI to enable the CXL memory devices in 1LM+Vol CXL memory mode operation are as follows.

1. Press F1 when prompted during server boot to enter System Setup.


```
ThinkSystem SR630 V4

System Events  0   0

Serial Number  WE1H5WMTVG
Machine Type   7DG8
BMC IP         10.245.21.170
UEFI Version   1.20 IHE109H (12/30/2024)
BMC Version    0.92 IHX409R (01/03/2025)
```

```
UEFI:POST END

2560 GB memory detected and configured in 1LM +
Vol mode
 - 2048 GB DIMM
 - 512 GB CXL Memory
Independent mode enabled, usable capacity 2560 GB
TPM_POLICY is not locked

Undefined TPM_POLICY found

2 processor(s) detected, 96 cores enabled
Intel(R) Xeon(R) 6740P
```

Figure 11. ThinkSystem SR630 V4 server boot to enter System Setup

2. Select System Settings > Memory > CXL Memory Module > Memory Mode, and press Enter to select 1LM+Vol mode.

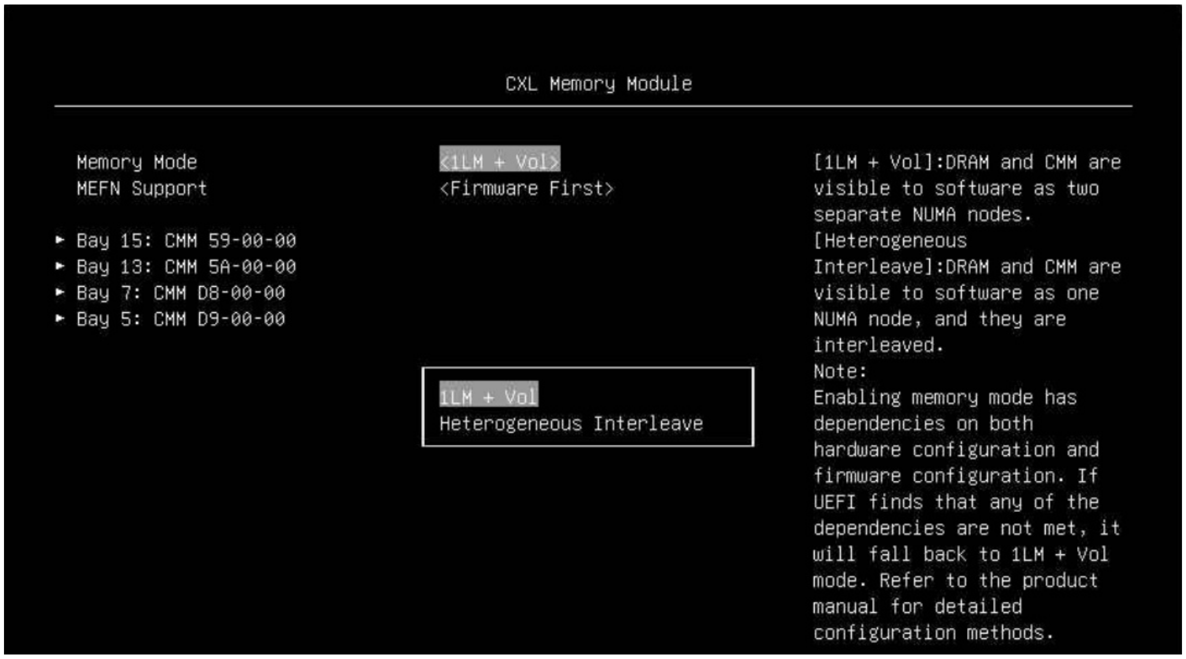
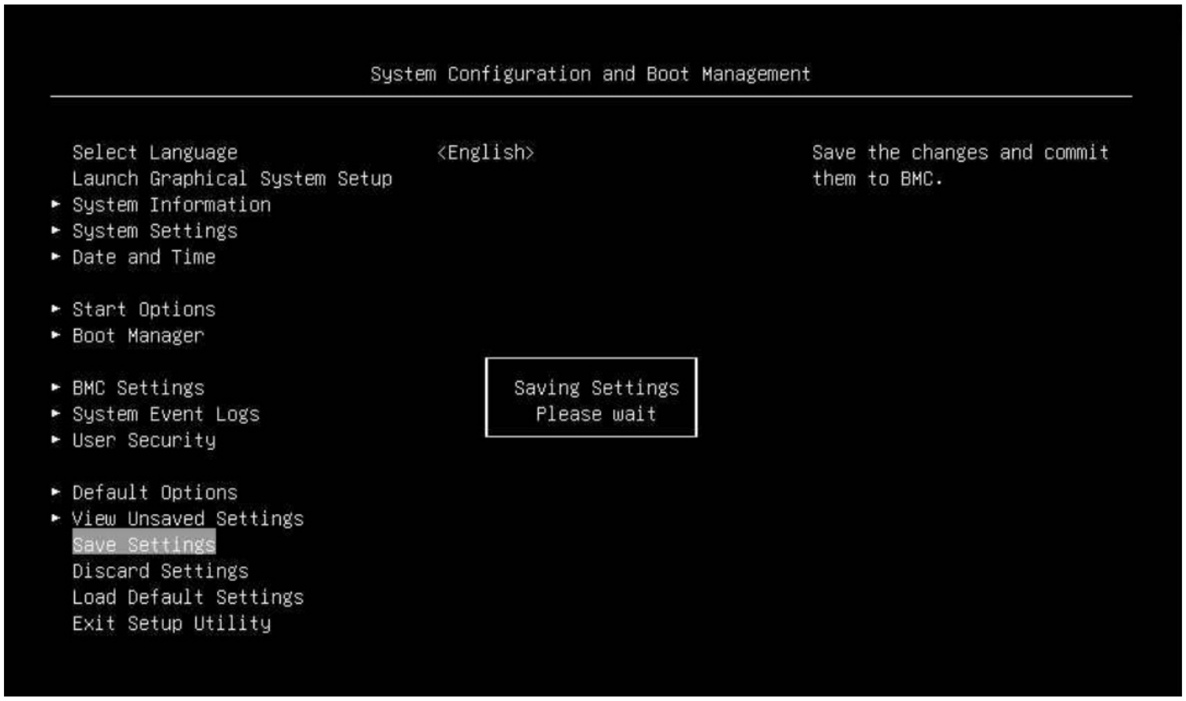

```
                         CXL Memory Module

  Memory Mode                <1LM + Vol>            [1LM + Vol]:DRAM and CMM are
  MEFN Support               <Firmware First>       visible to software as two
                                                    separate NUMA nodes.
► Bay 15: CMM 59-00-00                              [Heterogeneous
► Bay 13: CMM 5A-00-00                              Interleave]:DRAM and CMM are
► Bay 7: CMM D8-00-00                               visible to software as one
► Bay 5: CMM D9-00-00                               NUMA node, and they are
                                                    interleaved.
                                                    Note:
                             1LM + Vol              Enabling memory mode has
                             Heterogeneous Interleave  dependencies on both
                                                    hardware configuration and
                                                    firmware configuration. If
                                                    UEFI finds that any of the
                                                    dependencies are not met, it
                                                    will fall back to 1LM + Vol
                                                    mode. Refer to the product
                                                    manual for detailed
                                                    configuration methods.
```

Figure 12. Select the CXL Memory Module’s Memory Mode to 1LM+Vol

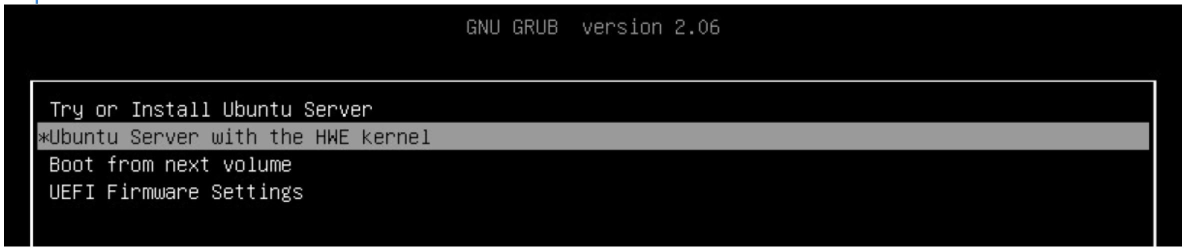
3. Save the settings and reboot the system.



```
                System Configuration and Boot Management

  Select Language                   <English>          Save the changes and commit
  Launch Graphical System Setup                        them to BMC.
► System Information
► System Settings
► Date and Time

► Start Options
► Boot Manager

► BMC Settings                       Saving Settings
► System Event Logs                    Please wait
► User Security

► Default Options
► View Unsaved Settings
  Save Settings
  Discard Settings
  Load Default Settings
  Exit Setup Utility
```

Figure 13. Save the settings before system reboot

4. After the system reboots to UEFI screen, the CXL operation mode should be in 1LM+Vol mode and the CXL memory capacity should be correctly detected, as shown in the right part of Figure 11.
5. Install the Ubuntu 24.04 LTS server version OS with HWE kernel. It should be straight forward, and we will not describe the OS installation details here. For more details on the Ubuntu server OS, see https://ubuntu.com/server



```
                    GNU GRUB  version 2.06

 Try or Install Ubuntu Server
*Ubuntu Server with the HWE kernel
 Boot from next volume
 UEFI Firmware Settings
```

Figure 14. Install the Ubuntu 24.04 LTS server version OS with HWE kernel

6. Now, boot to OS for CXL memory module validation.

CXL memory validation

In this section, we will verify the proper CXL memory module functionality in the Linux OS for this system, by checking the command output correctness with Linux commands like lspci, dmesg, free, lsmod, numactl, … etc., then we will use CXL related command line interface (CLI) tools to confirm CXL memory modules are running in the expected mode: 1LM+Vol mode.

- Linux commands
- CXL related commands

Linux commands

Below is a list of Linux OS built-in commands required to check the CXL components in the server from the OS perspective. Use these to get an overall picture about how the Linux OS CXL subsystem sees the CXL components and their behaviors.

- `lspci -vvv`

  One CXL memory device is detected at 0000:59:00.0 and similar devices are also detected at 0000:5a:00.0, 0000:d8:00.0, and 0000:d9:00.0 (not listed here). Note that NUMA node information from this command is not accurate; it should be NUMA node 2 and 3, as in the ACPI SRAT table, and the `numactl` command output (see below).

```
lspci -vvv
0000:59:00.0 CXL: Montage Technology Co., Ltd. Device c002 (rev 03) (pr
og-if 10 [CXL Memory Device (CXL 2.x)])
 Subsystem: Samsung Electronics Co Ltd Device 0112
 Physical Slot: 79
 Control: I/O- Mem+ BusMaster+ SpecCycle- MemWINV- VGASnoop- ParErr+ St
epping- SERR+ FastB2B- DisINTx+
 Status: Cap+ 66MHz- UDF- FastB2B- ParErr- DEVSEL=fast >TAbort- SERR-
 Kernel driver in use: cxl_pci
 Kernel modules: cxl_pci
```

- `dmesg | grep ACPI | grep SRAT`

  With this command, we confirm that this system's UEFI correctly implements the SRAT (System Resource Affinity Table) necessary for the CXL devices to be seen and interacted with the OS. Now the hotplug memory devices (CXL memory modules) are arranged as NUMA node 2 and 3. Note that the memory ranges are also compatible with the output from the command: `cxl list -Hu` (see the `cxl list -Hu` command in the CXL related commands section)

```
dmesg | grep ACPI | grep SRAT
[    0.010743] ACPI: SRAT: Node 0 PXM 0 [mem 0x00000000-0x7fffffff]
[    0.010745] ACPI: SRAT: Node 0 PXM 0 [mem 0x100000000-0x1007fffffff]
[    0.010746] ACPI: SRAT: Node 1 PXM 1 [mem 0x67e80000000-0x77e7ffffff
f]
[    0.010749] ACPI: SRAT: Node 2 PXM 10 [mem 0x10080000000-0x1407fffff
ff] hotplug
[    0.010750] ACPI: SRAT: Node 3 PXM 11 [mem 0x77e80000000-0x7be7fffff
ff] hotplug
```

- `free`

  The command show that the total memory space available to the OS is 2.5 TB, which is expected when all CXL memory module memory space is activated and available to the OS.

```
free
               total        used        free      shared  buff/cache
available
Mem:      2650416280    21271888  2636001748       13164     1013908   2
629144392
Swap:        8388604           0     8388604
```

- `lsmod | grep -i CXL`

This command lists all the CXL memory device related kernel modules that are loaded.

```
lsmod | grep -i CXL
dax_cxl                12288  0
cxl_mem                12288  0
cxl_pmu                32768  0
cxl_pmem               24576  0
cxl_port               16384  0
cxl_pci                28672  0
cxl_acpi               24576  0
cxl_core              299008  11 cxl_pmem,cxl_port,cxl_mem,cxl_pci,dax_
cxl,cxl_acpi,cxl_pmu
```

- `numactl -H`

  This command shows that there are 4x NUMA nodes. The DRAM memory spaces are arranged as NUMA node 0 and 1, each with size 1.0 TB, while CXL memory module memory spaces are arranged as NUMA node 2 and 3, each with size 256 GB.

```
numactl -H
available: 4 nodes (0-3)
node 0 cpus: 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22
23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46
47 96 97 98 99 100 101 102 103 104 105 106 107 108 109 110 111 112 113 1
14 115 116 117 118 119 120 121 122 123 124 125 126 127 128 129 130 131
132 133 134 135 136 137 138 139 140 141 142 143
node 0 size: 1031904 MB
node 0 free: 1025554 MB
node 1 cpus: 48 49 50 51 52 53 54 55 56 57 58 59 60 61 62 63 64 65 66 67
68 69 70 71 72 73 74 75 76 77 78 79 80 81 82 83 84 85 86 87 88 89 90 91
92 93 94 95 144 145 146 147 148 149 150 151 152 153 154 155 156 157 158
159 160 161 162 163 164 165 166 167 168 169 170 171 172 173 174 175 176
177 178 179 180 181 182 183 184 185 186 187 188 189 190 191
node 1 size: 1032105 MB
node 1 free: 1024244 MB
node 2 cpus:
node 2 size: 262144 MB
node 2 free: 262143 MB
node 3 cpus:
node 3 size: 262144 MB
node 3 free: 262142 MB
node distances:
node   0   1   2   3
  0:  10  21  14  24
  1:  21  10  24  14
  2:  14  24  10  26
  3:  24  14  26  10
```

CXL related commands

As mentioned in Table 2 and Figure 7 in the CXL device support: Software perspective section, all the NDCTL, DAXCTL, and CXL-CLI packages are required for validation of CXL memory module operation modes. Some of the OS either did not have these packages, or they shipped with older versions of these packages, in such case, you may need to install or upgrade to the latest versions of these software packages to make sure they work properly on your OS.

To install the NDCTL and DAXCTL packages on Linux, please follow the steps listed in: https://docs.pmem.io/ndctl-user-guide/installing-ndctl/installing-ndctl-packages-on-linux

To install the CXL-CLI package on Linux, please follow the steps listed in: https://docs.pmem.io/ndctl-user-guide/installing-ndctl/installing-the-cxl-cli-package-on-linux

After installing or upgrading the necessary software packages, we can perform the commands with results shown below.

- `daxctl list -Mu`

  The output indicates that there are two dax devices, dax0.0 and dax9.0, assigned to NUMA node 2 and 3 respectively, each with size 256 GB and both are running in the system-ram mode.

```
daxctl list -Mu
[
  {
    "chardev":"dax9.0",
    "size":"256.00 GiB (274.88 GB)",
    "target_node":3,
    "align":2097152,
    "mode":"system-ram"
  },
  {
    "chardev":"dax0.0",
    "size":"256.00 GiB (274.88 GB)",
    "target_node":2,
    "align":2097152,
    "mode":"system-ram"
  }
]
```

- `cxl list -Hu`

  This command indicates that there are 4x CXL memory devices, mem0 to mem3, each with a size 128 GB. These devices are grouped into two regions, region0 and region9, and both regions are running in the ram type mode. This is compatible with the information from the daxctl command, which is the expected result.

  Note that in this case, the CXL command output says some misleading information about the NUMA node 0 and 1, while both daxctl and numactl command outputs say the CXL NUMA nodes are 2 and 3. We remind users about this difference to take care of, if the exact NUMA node numbers are important in the user application. We expect the future version of OS and CXL CLI will correct this issue.

```
cxl list -Hu
[
  {
    "memdevs":[
      {
        "memdev":"mem0",
        "ram_size":"128.00 GiB (137.44 GB)",
        "serial":"0x7fffffffffffffff",
        "numa_node":0,
        "host":"0000:59:00.0"
      },
      {
        "memdev":"mem1",
        "ram_size":"128.00 GiB (137.44 GB)",
        "serial":"0x7fffffffffffffff",
        "numa_node":0,
        "host":"0000:5a:00.0"
      },
      {
        "memdev":"mem3",
        "ram_size":"128.00 GiB (137.44 GB)",
        "serial":"0x7fffffffffffffff",
        "numa_node":1,
        "host":"0000:d9:00.0"
      },
      {
        "memdev":"mem2",
        "ram_size":"128.00 GiB (137.44 GB)",
        "serial":"0x7fffffffffffffff",
        "numa_node":1,
        "host":"0000:d8:00.0"
      }
    ]
  },
  {
    "regions":[
      {
        "region":"region0",
        "resource":"0x10080000000",
        "size":"256.00 GiB (274.88 GB)",
        "type":"ram",
        "interleave_ways":2,
        "interleave_granularity":256,
        "decode_state":"commit"
      },
      {
        "region":"region9",
        "resource":"0x77e80000000",
        "size":"256.00 GiB (274.88 GB)",
        "type":"ram",
        "interleave_ways":2,
        "interleave_granularity":256,
        "decode_state":"commit"
      }
    ]
  }
]
```

As shown with the above command outputs, we validated that the CXL memory modules behave correctly with both Linux commands, and CXL CLI-related commands with Ubuntu 24.04 LTS OS, on the Lenovo ThinkSystem SR630 V4 server system.

Known issues and troubleshooting

In addition to the above mentioned lspci and CXL CLI NUMA node information issue with Ubuntu 24.04, the issues below were also found in recent Linux OS distributions. Lenovo is working with the OS vendors and expecting these issues will be fixed in the future releases.

1. CXL related complaining messages shown even CXL-capable PCIe root port did not attach CXL device (on Ubuntu 24.04, RHEL 9.5)
2. Not able to run the CMM (CXL Memory Module) in the 1LM+Vol mode under SLES15.6
3. CXL CLI tool does not work correctly with CXL device under RHEL9.4

References

For more information, see these references:

- CXL 2.0 and 3.0 specifications:
  https://computeexpresslink.org/wp-content/uploads/2024/02/CXL-2.0-Specification.pdf
  https://computeexpresslink.org/wp-content/uploads/2024/02/CXL-3.0-Specification.pdf
- CXL + Memory Expander - Baseline Guide for Birch Stream (login required)
  https://www.intel.com/content/www/us/en/secure/content-details/803836/cxl-memory-expander-baseline-guide-for-birch-stream.html?wapkw=803836
- CXL Type 3 Memory Device Software Guide
  https://www.intel.com/content/www/us/en/content-details/643805/cxl-memory-device-software-guide.html?wapkw=643805
- CXL Supplemental Validation Guide – Birch Stream (login required)
  https://www.intel.com/content/www/us/en/secure/content-details/814339/compute-express-link-cxl-supplemental-validation-guide-for-birch-stream.html?wapkw=814339
- Installing the CXL-CLI Package on Linux
  https://docs.pmem.io/ndctl-user-guide/installing-ndctl/installing-the-cxl-cli-package-on-linux
- A Practical Guide to Identify Compute Express Link (CXL) Devices in Your Server
  https://stevescargall.com/blog/2023/05/a-practical-guide-to-identify-compute-express-link-cxl-devices-in-your-server/
- Using Linux Kernel Tiering with Compute Express Link (CXL) Memory
  https://stevescargall.com/blog/2024/05/using-linux-kernel-tiering-with-compute-express-link-cxl-memory/
- Memory Hot(Un)Plug
  https://docs.kernel.org/admin-guide/mm/memory-hotplug.html

Author

Kelvin Shieh is the OS Development Technical Lead for the Lenovo Infrastructure Solutions Group, based in Taipei, Taiwan.

Related product families

Product families related to this document are the following:

- Memory
- ThinkSystem SR630 V4 Server
- ThinkSystem SR650 V4 Server

Notices

This document, LP2184, was created or updated on March 21, 2025.

Send us your comments in one of the following ways:

- Use the online Contact us review form found at:
  https://lenovopress.lenovo.com/LP2184
- Send your comments in an e-mail to:
  comments@lenovopress.com

This document is available online at https://lenovopress.lenovo.com/LP2184.

Trademarks