Hardware LLC prefetch feature on 4th Gen Intel® Xeon® Scalable Processor (Codename Sapphire Rapids)

Document Type

June 2023

Revision 1.0

Notice: This document contains information on products in the design phase of development. The information here is subject to change without notice. Do not finalize a design with this information.

Intel technologies' features and benefits depend on system configuration and may require enabled hardware, software, or service activation. Learn more at intel.com, or from the OEM or retailer.

No computer system can be absolutely secure. Intel does not assume any liability for lost or stolen data or systems or any damages resulting from such losses.

You may not use or facilitate the use of this document in connection with any infringement or other legal analysis concerning Intel products described herein. You agree to grant Intel a non-exclusive, royalty-free license to any patent claim thereafter drafted which includes subject matter disclosed herein.

No license (express or implied, by estoppel or otherwise) to any intellectual property rights is granted by this document. The products described may contain design defects or errors known as errata which may cause the product to deviate from published specifications. Current characterized errata are available on request.

This document contains information on products, services and/or processes in development. All information provided here is subject to change without notice. Contact your Intel representative to obtain the latest Intel product specifications and roadmaps.

Intel disclaims all express and implied warranties, including without limitation, the implied warranties of merchantability, fitness for a particular purpose, and non-infringement, as well as any warranty arising from course of performance, course of dealing, or usage in trade.

Tests document performance of components on a particular test, in specific systems. Differences in hardware, software, or configuration will affect actual performance. Consult other sources of information to evaluate performance as you consider your purchase. For more complete information about performance and benchmark results, visit http://www.intel.com/performance.

Results have been estimated or simulated using internal Intel analysis or architecture simulation or modeling and provided to you for informational purposes. Any differences in your system hardware, software or configuration may affect your actual performance.

Intel does not control or audit third-party benchmark data or the web sites referenced in this document. You should visit the referenced web site and confirm whether referenced data are accurate.

Intel is a sponsor and member of the Benchmark XPRT Development Community and was the major developer of the XPRT family of benchmarks. Principled Technologies is the publisher of the XPRT family of benchmarks. You should consult other information and performance tests to assist you in fully evaluating your contemplated purchases.

Copies of documents which have an order number and are referenced in this document may be obtained by calling 1-800-548-4725 or by visiting www.intel.com/design/literature.htm.

Intel, the Intel logo, and Xeon are trademarks of Intel Corporation in the U.S. and/or other countries.

*Other names and brands may be claimed as the property of others

Contents

Figures

Tables

Revision History

Revision Number	Description	Date
1.0	• Initial release of the document.	June 2023

1 Description

LLC prefetch implementation includes software (SW) and hardware (HW) LLC prefetch features; in this document we are discussing with hardware LLC prefetch which is a feature for Intel® Xeon® systems. The term "LLC prefetch" in this document is an abridge of "hardware LLC prefetch."

In previous platforms like the 3rd Generation Intel® Xeon® Scalable processor (codename Ice Lake), LLC prefetch is enabled by default; after the 4th Gen Intel® Xeon® Scalable processor (codename Sapphire Rapids), the same control knob is disabled by default. As a performance tuning guidance, this document would help end-users to select the best benefit option for HW LLC prefetch setting.

2 Terms

Term	Meaning
LLC	CPU last level cache / CPU Level 3 cache
LLC prefetch	Hardware based LLC prefetch
ICX	3rd Generation Intel® Xeon® Scalable processor (codename Ice Lake)
SPR	4th Gen Intel® Xeon® Scalable processor (codename Sapphire Rapids)
EGS	Eagle Stream system platform
MSR	Model Specific Register
Intel® MLC	Intel® Memory Latency Checker
DRAM	Dynamic Random Access Memory
HBM	High Bandwidth Memory

3 About HW LLC Prefetch Enabling

Hardware prefetch can bring cache lines into the unified last-level cache, based on prior data misses. It will attempt to prefetch cache lines ahead of the prefetch stream. Characteristics of the hardware prefetcher requires regularity in the data access pattern:

- The hardware prefetcher may consume extra system bandwidth if the application's memory traffic has significant portions with strides of cache misses, greater than the trigger distance threshold of hardware prefetch (large stride memory traffic).
- The effectiveness with existing applications depends on the proportions of small stride versus large stride accesses in the application's memory traffic. An application with a preponderance of small stride memory traffic with good temporal locality will benefit greatly from the automatic hardware prefetcher.

Except hardware prefetch effectiveness, a start-up penalty before the prefetcher triggers and there may fetch an array finish. For short arrays, overhead can reduce effectiveness.

The LLC prefetcher is a prefetcher added to the Intel® Xeon® Scalable family because 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 data cache unit and L2 cache. Enabling LLC prefetch gives the core prefetcher the ability to prefetch data directly into the LLC without necessarily filling into the Intel® Memory Latency Checker (Intel® MLC). In some cases, setting this option to disabled can improve performance.

Values for this BIOS option can be:

- **Disabled:** Disables the LLC prefetcher. The other core prefetchers are unaffected. If the LLC prefetch feature is disabled, the system would keep higher memory access efficiency that conservatively makes performance flat.
- **Enabled:** Gives the core prefetcher the ability to prefetch data directly to the LLC. With prefetch enabled, the system would radically leverage more memory access traffic, memory performance expression can be improved if memory controller is not the critical performance bottleneck.

All memory prefetch features that include hardware LLC prefetch can help reduce the long latency typically associated with reading data from memory, and thus help prevent processor "stalls." However, data should be used judiciously. Overuse can lead to resource conflicts and reduce the performance of an application.

4 MSR Indicator

MSR 0x6D would show more details above all prefetch status of the CPU, especially the offset #42 that indicates LLC prefetch feature.

Table 4-1. MSR Indicator Example

4th Gen Intel® Xeon® Scalable Processor Server – MSR 0x6D		
Bit#	**Description**	**Explain**
…		
42	L3 prefetch disable	1: HW LLC prefetch disabled
		0: HW LLC prefetch enabled
…		

Under Linux*, using the `rdmsr` tool may read this configuration. If your system has not installed `msr-tools`, use the command `apt install msr-tools` or `dnf install msr-tools` to install the `msr-tool` set.

```
[root@localhost ~]# rdmsr 0x6d
4004000c000
```

5 Performance Comparison

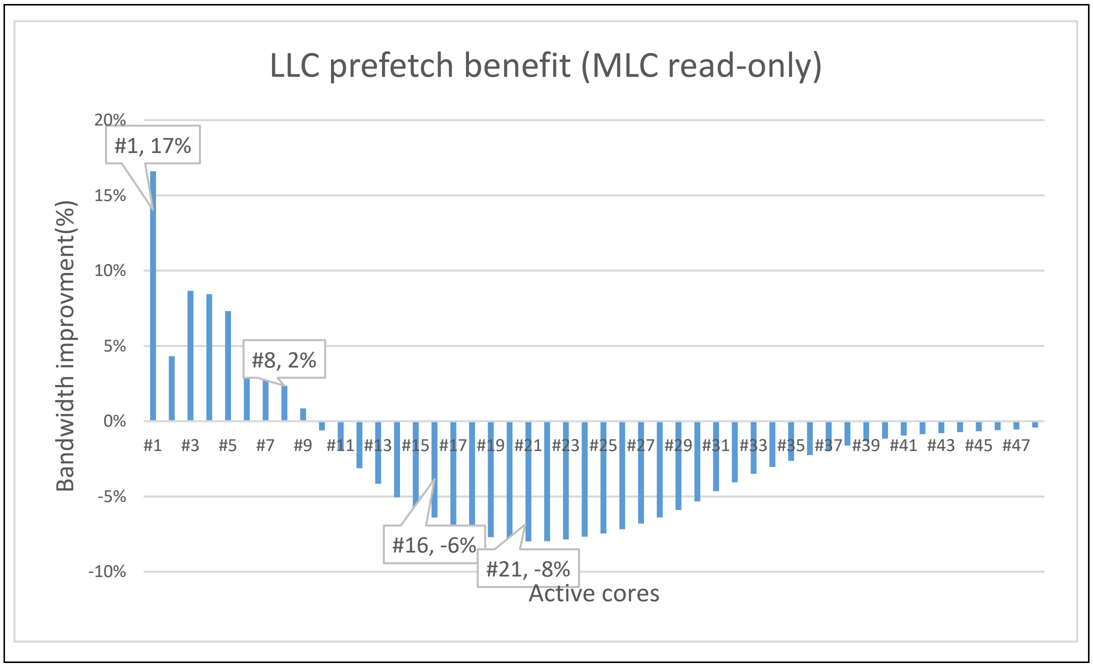
5.1 Memory Bandwidth Benchmark – Intel® MLC

This benchmark can help to represent the best and worst interference on memory bandwidth for LLC prefetch feature.

As a typical memory bandwidth benchmark tool, Intel® MLC may provide more evidence on memory bandwidth performance potential. Based on LLC prefetch implementation, memory requests that contain more read operations are more sensitive on prefetch enable/disable config change.

For this experiment, Intel® MLC stresses memory bandwidth by dedicating CPU core counts that simulates work conditions changing gradually from lighter to heavier.

Figure 5-1. LLC Prefetch Benefit (Intel® MLC Read-Only)



These Intel® MLC benchmark results clearly show the impactions of LLC prefetch on/off:

1. Under single core scenario, enabled LLC prefetch may largely increase memory bandwidth.
2. For lighter memory stress, for example, two cores to nine cores scenarios, LLC prefetch improves memory bandwidth.
3. For heavier memory stress, for example, 10 cores to 36 cores scenarios, memory bandwidth became the most critical bottleneck for the entire system. Under same situations, LLC prefetch enabling might exacerbate

memory bandwidth saturation, and memory performance would be impacted. The worst case is around 8% performance drop.

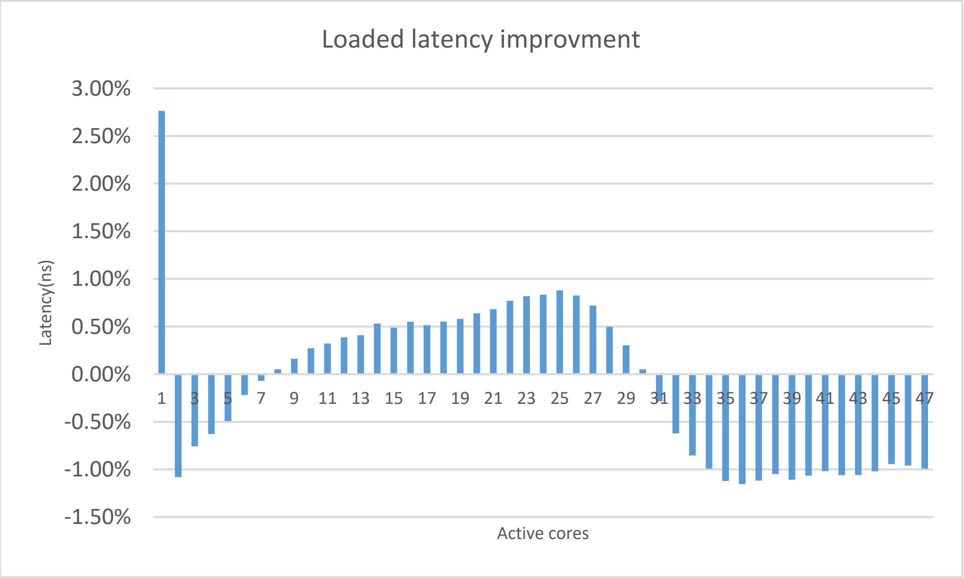
4. On the heaviest memory stressing scenarios, memory controller or even memory DIMM are the most critical bottlenecks for the entire system. Enable/disable prefetch did not observably change the results.

5.2 Memory Access Latency Measurement – Intel® MLC

This experiment is depended on Intel® MLC "loaded_latency" benchmark component what can measure memory bandwidth and latency under dedicate memory stressing. Within heavier memory stress, higher memory bandwidth and higher memory latency is expected.

Better memory performance systems will show better expressions under heavier memory stressing.

Figure 5-2. Loaded Latency Improvement



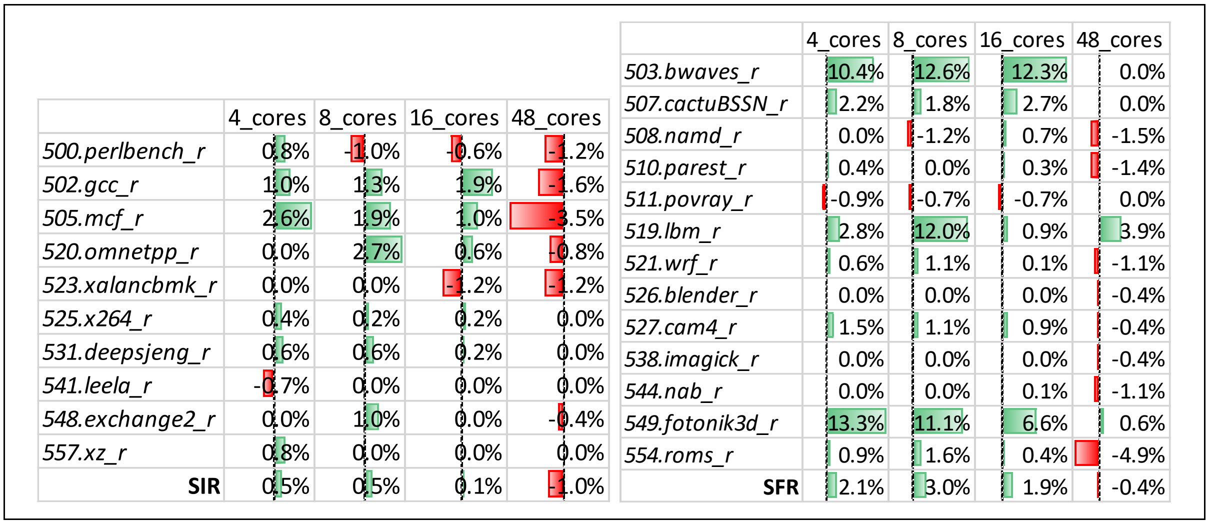
Depending on different memory loading, LLC prefetch shows behavior changes.

- For single core case, LLC prefetch brings the best performance benefits, it reduced >2.5% memory latency time by prefetch enabling.
- In 2~8 core sets cases, LLC prefetch does not show more benefits of memory latency.
- In 9~30 core sets, memory latency got 0.5%~1% reduction.
- In 30 or more core cases, since memory bandwidth gradually became to the critical bottleneck, prefetch also becomes a hindrance of test results.

5.3 Workload Performance Benefits – SPECcpu 2017*

Last experiment shows what would happen for LLC prefetch configure change. Intel® MLC is a memory benchmark tool that represents most intensive situations of the system assessed. SPECcpu 2017* is a benchmark tool that helps to measure the impacts of a universal workload list.

Table 5-1. LLC Prefetch Configure Change



	4_cores	8_cores	16_cores	48_cores
500.perlbench_r	0.8%	-1.0%	-0.6%	-1.2%
502.gcc_r	1.0%	1.3%	1.9%	-1.6%
505.mcf_r	2.6%	1.9%	1.0%	-3.5%
520.omnetpp_r	0.0%	2.7%	0.6%	-0.8%
523.xalancbmk_r	0.0%	0.0%	-1.2%	-1.2%
525.x264_r	0.4%	0.2%	0.2%	0.0%
531.deepsjeng_r	0.6%	0.6%	0.2%	0.0%
541.leela_r	-0.7%	0.0%	0.0%	0.0%
548.exchange2_r	0.0%	1.0%	0.0%	-0.4%
557.xz_r	0.8%	0.0%	0.0%	0.0%
SIR	0.5%	0.5%	0.1%	-1.0%

	4_cores	8_cores	16_cores	48_cores
503.bwaves_r	10.4%	12.6%	12.3%	0.0%
507.cactuBSSN_r	2.2%	1.8%	2.7%	0.0%
508.namd_r	0.0%	-1.2%	0.7%	-1.5%
510.parest_r	0.4%	0.0%	0.3%	-1.4%
511.povray_r	-0.9%	-0.7%	-0.7%	0.0%
519.lbm_r	2.8%	12.0%	0.9%	3.9%
521.wrf_r	0.6%	1.1%	0.1%	-1.1%
526.blender_r	0.0%	0.0%	0.0%	-0.4%
527.cam4_r	1.5%	1.1%	0.9%	-0.4%
538.imagick_r	0.0%	0.0%	0.0%	-0.4%
544.nab_r	0.0%	0.0%	0.1%	-1.1%
549.fotonik3d_r	13.3%	11.1%	6.6%	0.6%
554.roms_r	0.9%	1.6%	0.4%	-4.9%
SFR	2.1%	3.0%	1.9%	-0.4%

SPECcpu 2017 results present possibility of LLC prefetch impaction:

1. * Workloads show different performance trends by prefetch enabling. Briefly, workload with memory bound gets the more performance interference.
2. Briefly, under less core counts scenarios, more workloads got performance improvement; but when used 48 cores in the SPECcpu* test, most workloads had not performance increasing, even some workloads, for example, 505.mcf or 554.roms, got observable performance drop from LLC prefetch.
3. SFR workloads shows more sensitive than SIR workloads, but if we focus on the summary results of whole SPECcpu benchmark, performance difference is exceedingly small. (As a reference: normally, SPECcpu results contain around ±2% performance different as round-to-round variation).

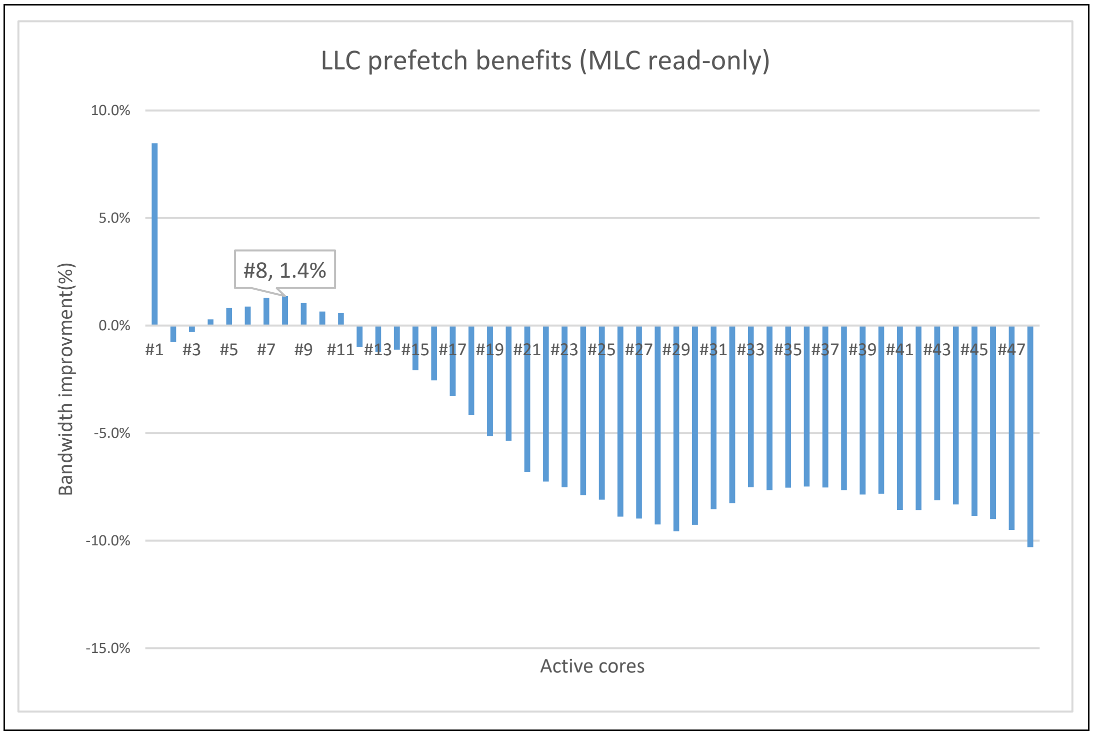
Note: Reference Appendix B for detailed subcomponent profiles.

6 HBM Relevant

Unlike typical DRAM (DDR-4/DDR-5) media, HBM system has higher memory bandwidth and more frequently memory traffic. Prefetch feature brings a different behavior on HBM compared to DRAM memory.

Following the previous experiment steps, the use cases in HBM systems (HBM only mode) test results are different.

Table 6-1. LLC Prefetch Benefits (Intel® MLC Read-Only)



1. Under less core scenarios, there is around 1% bandwidth benefits.
2. For more cores scenarios, prefetch brings increased bandwidth degradation to HBM systems.

Based on these memory bandwidth data, we strongly suggest disabling LLC prefetch feature if you are using HBM.

7 Reference

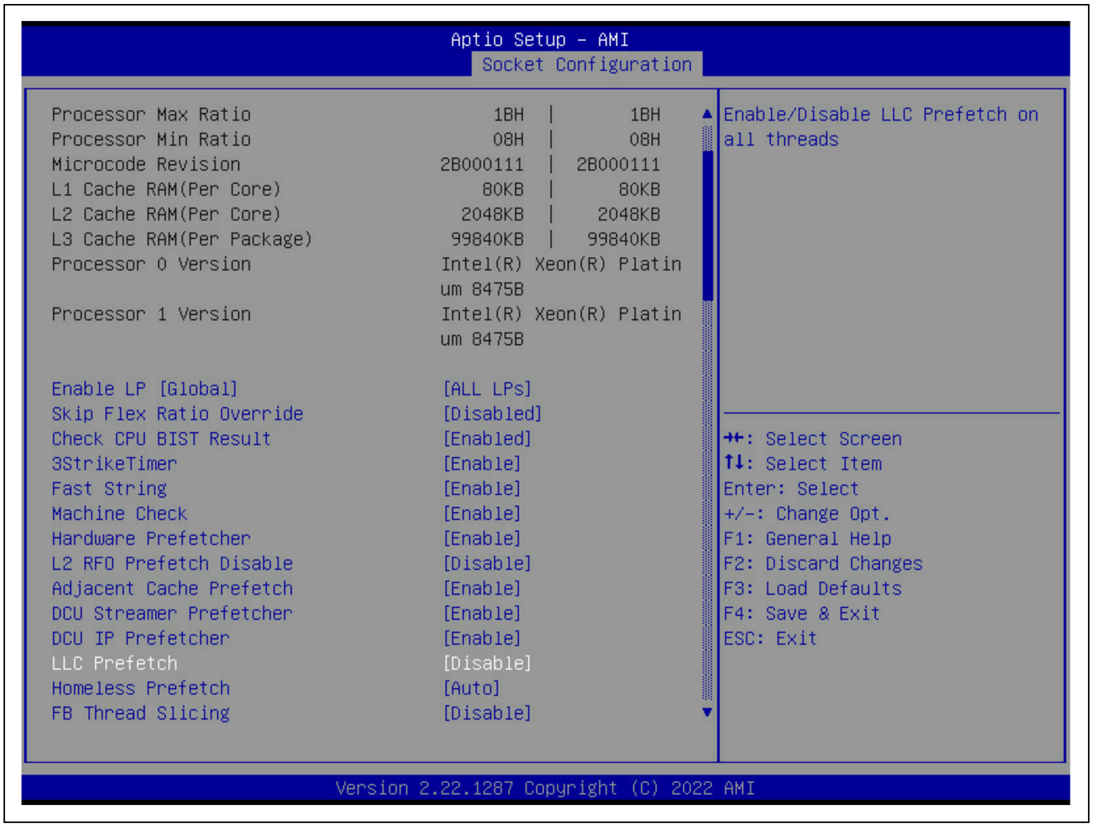
- 248966, Intel® 64 and IA-32 Architectures Optimization Reference Manual
- 325462, Intel® 64 and IA-32 Architectures Software Developer's Manual
- 634434, Intel® Xeon® Scalable Family [Whitley] Platform Performance and Power Optimization Guide
- Computer Architecture Lecture 18: Prefetching, Onur Mutlu, YouTube
- iVE University: Intel HW Prefetchers- An Overview
- A memory benchmark tool: Intel® Memory Latency Checker v3.10 (Intel® MLC) https://www.intel.com/content/www/us/en/developer/articles/tool/intelr-memory-latency-checker.html

A How to Enable LLC Prefetch

Inside BIOS interface, go to the **Socket Configuration** section, you may find the “LLC prefetch” option listed.

By default, on the Eagle Stream platform, this option is set to *Disabled*, you may change it to *Enabled* and save this change.

Table 7-1. LLC Prefetch on BIOS



```
                               Aptio Setup - AMI
                             Socket Configuration

  Processor Max Ratio                 1BH  |     1BH    Enable/Disable LLC Prefetch on
  Processor Min Ratio                 08H  |     08H    all threads
  Microcode Revision             2B000111  | 2B000111
  L1 Cache RAM(Per Core)             80KB  |    80KB
  L2 Cache RAM(Per Core)           2048KB  |  2048KB
  L3 Cache RAM(Per Package)       99840KB  | 99840KB
  Processor 0 Version            Intel(R) Xeon(R) Platin
                                 um 8475B
  Processor 1 Version            Intel(R) Xeon(R) Platin
                                 um 8475B

  Enable LP [Global]             [ALL LPs]
  Skip Flex Ratio Override       [Disabled]
  Check CPU BIST Result          [Enabled]              →←: Select Screen
  3StrikeTimer                   [Enable]               ↑↓: Select Item
  Fast String                    [Enable]               Enter: Select
  Machine Check                  [Enable]               +/-: Change Opt.
  Hardware Prefetcher            [Enable]               F1: General Help
  L2 RFO Prefetch Disable        [Disable]              F2: Discard Changes
  Adjacent Cache Prefetch        [Enable]               F3: Load Defaults
  DCU Streamer Prefetcher        [Enable]               F4: Save & Exit
  DCU IP Prefetcher              [Enable]               ESC: Exit
  LLC Prefetch                   [Disable]
  Homeless Prefetch              [Auto]
  FB Thread Slicing              [Disable]

                    Version 2.22.1287 Copyright (C) 2022 AMI
```

B SPECcpu2017 Subcomponents

SPECrate 2017*	Language	Application Area	Profile
500.perlbench_r	C	Perl interpreter	Core-bound
502.gcc_r	C	GNU C compiler	Memory-bound
505.mcf_r	C	Route planning	Memory-bound
520.omnetpp_r	C++	Discrete Event simulation - computer network	Memory-bound
523.xalancbmk_r	C++	XML to HTML conversion via XSLT	Mixed
525.x264_r	C	Video compression	Core-bound
531.deepsjeng_r	C++	Artificial Intelligence: alpha-beta tree search (Chess)	Core-bound
541.leela_r	C++	Artificial Intelligence: Monte Carlo tree search (Go)	Core-bound
548.exchange2_r	Fortran	Artificial Intelligence: recursive solution generator (Sudoku)	Core-bound
557.xz_r	C	General data compression	Latency-bound
503.bwaves_r	Fortran	Explosion modeling	Memory-bound
507.cactuBSSN_r	C++, C, Fortran	Physics: relativity	Latency-bound
508.namd_r	C++	Molecular dynamics	Core-bound
510.parest_r	C++	Biomedical imaging: optical tomography with finite elements	Memory-bound
511.povray_r	C++, C	Ray tracing	Core-bound
519.lbm_r	C	Fluid dynamics	Memory-bound
521.wrf_r	Fortran, C	Weather forecasting	Memory-bound
526.blender_r	C++, C	3D rendering and animation	Core-bound
527.cam4_r	Fortran, C	Atmosphere modeling	Mixed
538.imagick_r	C	Image manipulation	Core-bound
544.nab_r	C	Molecular dynamics	Core-bound
549.fotonik3d_r	Fortran	Computational Electromagnetics	Memory-bound
554.roms_r	Fortran	Regional ocean modeling	Memory-bound

C Reference Code for Intel® MLC Core Scaling Evaluation

```bash
#!/bin/bash

#
# Measurement MAX memory bandwidth by core counts.
#

MLC="mlc"     # MLC command
OPERATION="R" # Read-only as default
DRATION=5     # Run 5 seconds as default
RAND=0        # 1: random memory access, 0: sequential memory access

function memorybw_core()
{
    core_count=$1
    core_set=0-$(($core_count - 1))

    if [ $RAND == "0" ]
    then
        bw=$(${MLC} --loaded_latency -d0 -${OPERATION} -t${DRATION} -T -k${core_set} | grep 00000 | awk '{print $3}')
    else
        bw=$(${MLC} --loaded_latency -d0 -${OPERATION} -t${DRATION} -T -k${core_set} -r | grep 00000 | awk '{print $3}')

    fi
    echo $bw
}
```

```
function get_core_counts()
{
    skt=$(lscpu|grep "Socket(s)" | awk -F: '{print $2}')
    cps=$(lscpu|grep "Core(s) per socket" | awk -F: '{print $2}')

    echo $(($skt*$cps))
}

for num in $(seq 0 $(($(get_core_counts)-1)))
do
    bw=$(memorybw_core $num)
    echo "#$num $bw"
done
```