LINUXCON @ THE LINUX FOUNDATION OPEN SOURCE SUMMIT EUROPE

SK hynix

DAMON Usage for CXL Memory (with HMSDK)

Sep 16, 2024

SK hynix

Honggyu Kim

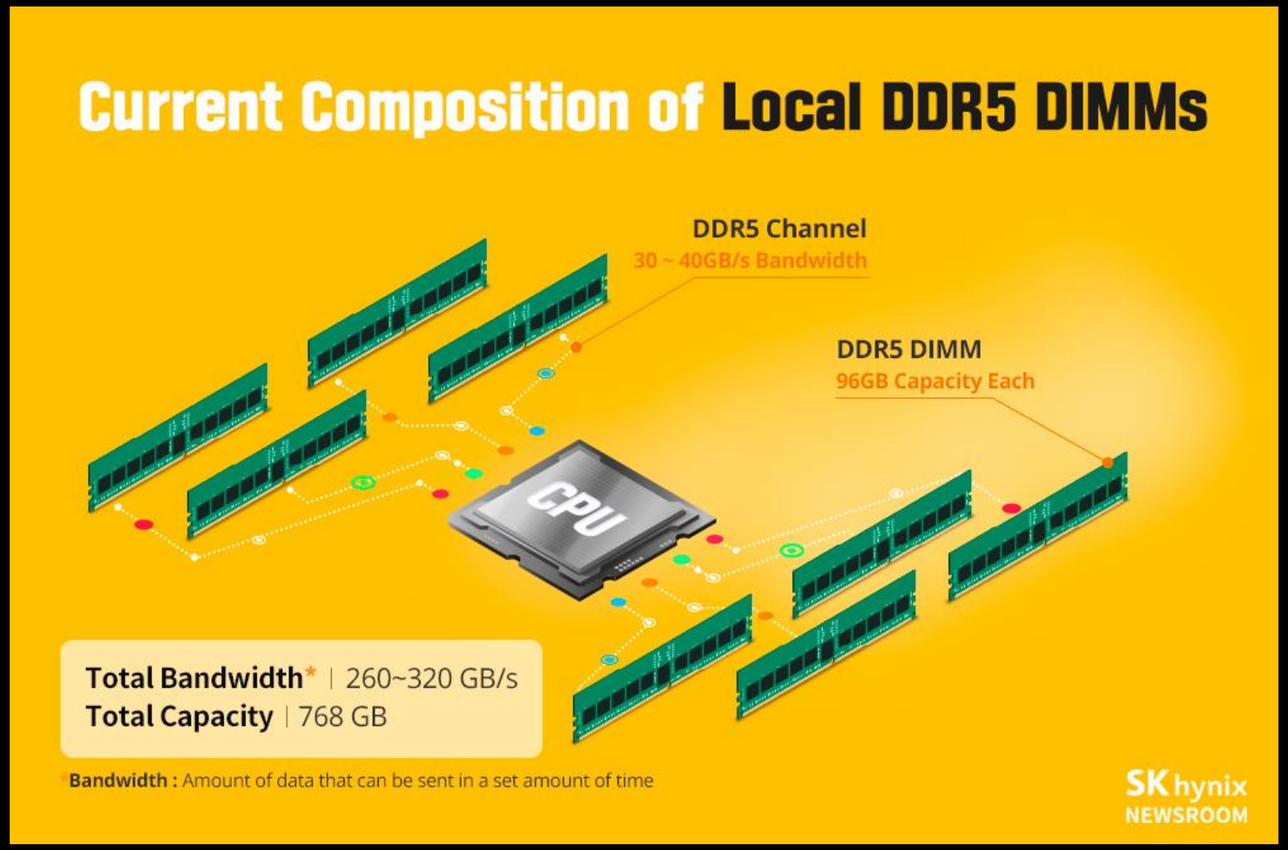
honggyu.kim@sk.com

THE LINUX FOUNDATION

What is CXL Memory?

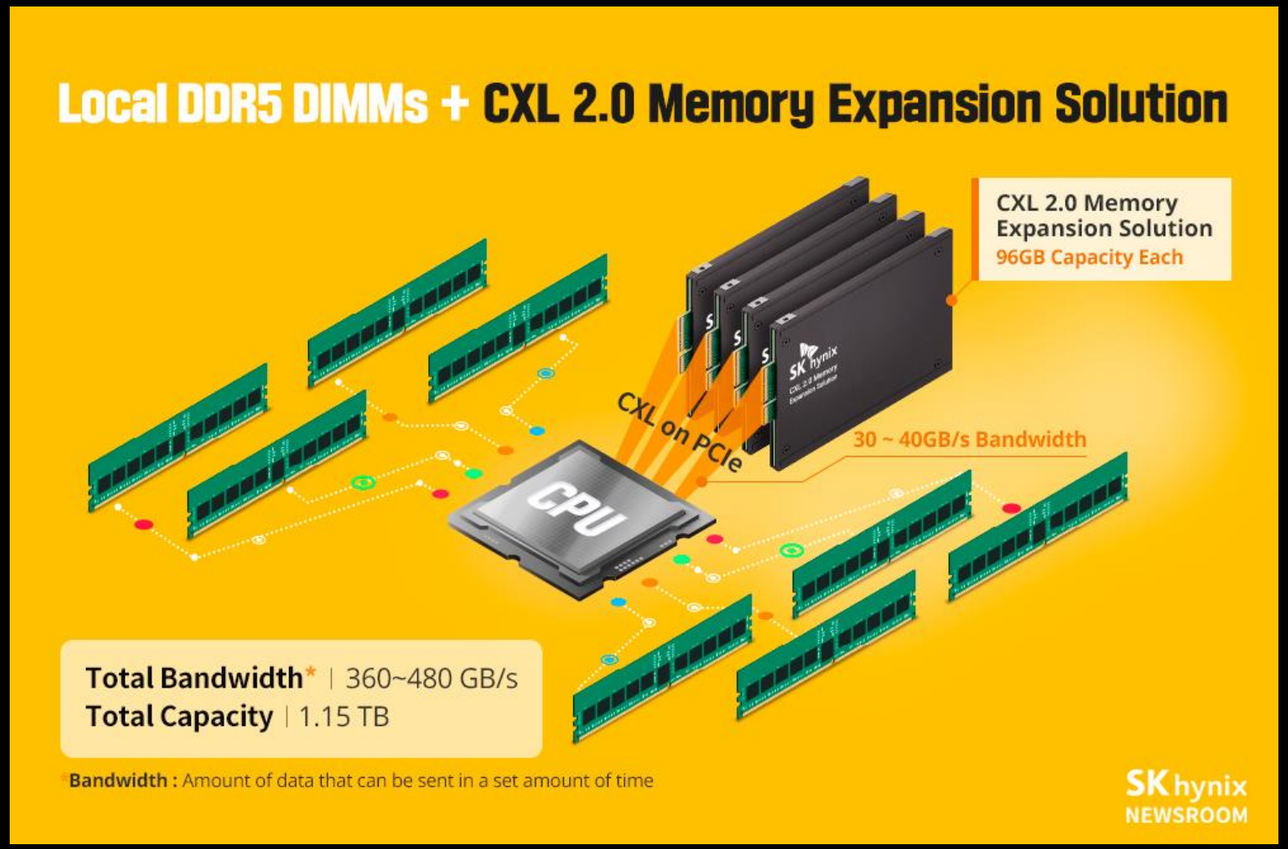
CXL Memory

- **Typical (Homogeneous) Memory System**
  - Most systems have the same type of DRAM in their DIMM slots.



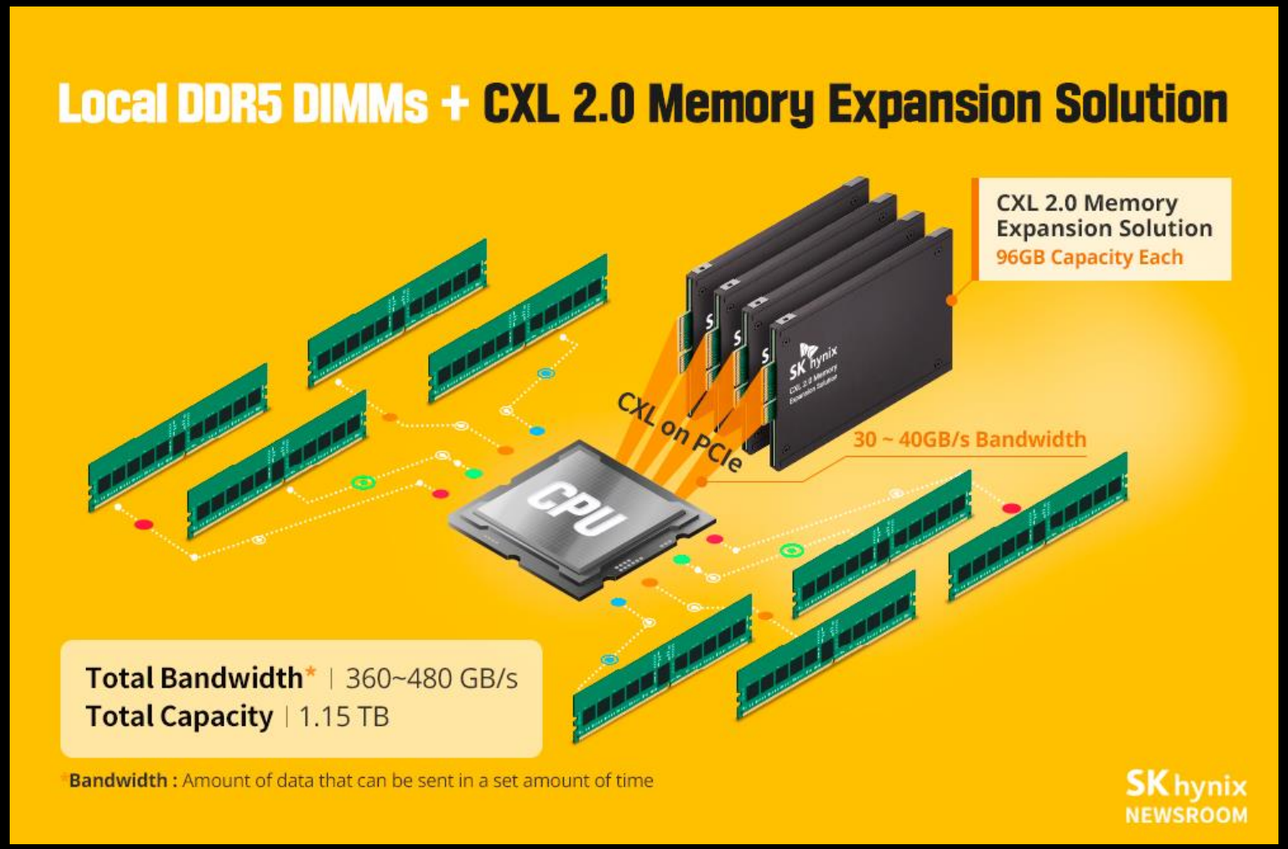
CXL Memory

- **Heterogeneous Memory System**
  - CXL memory can be flexibly attached via CXL on PCIe interface.



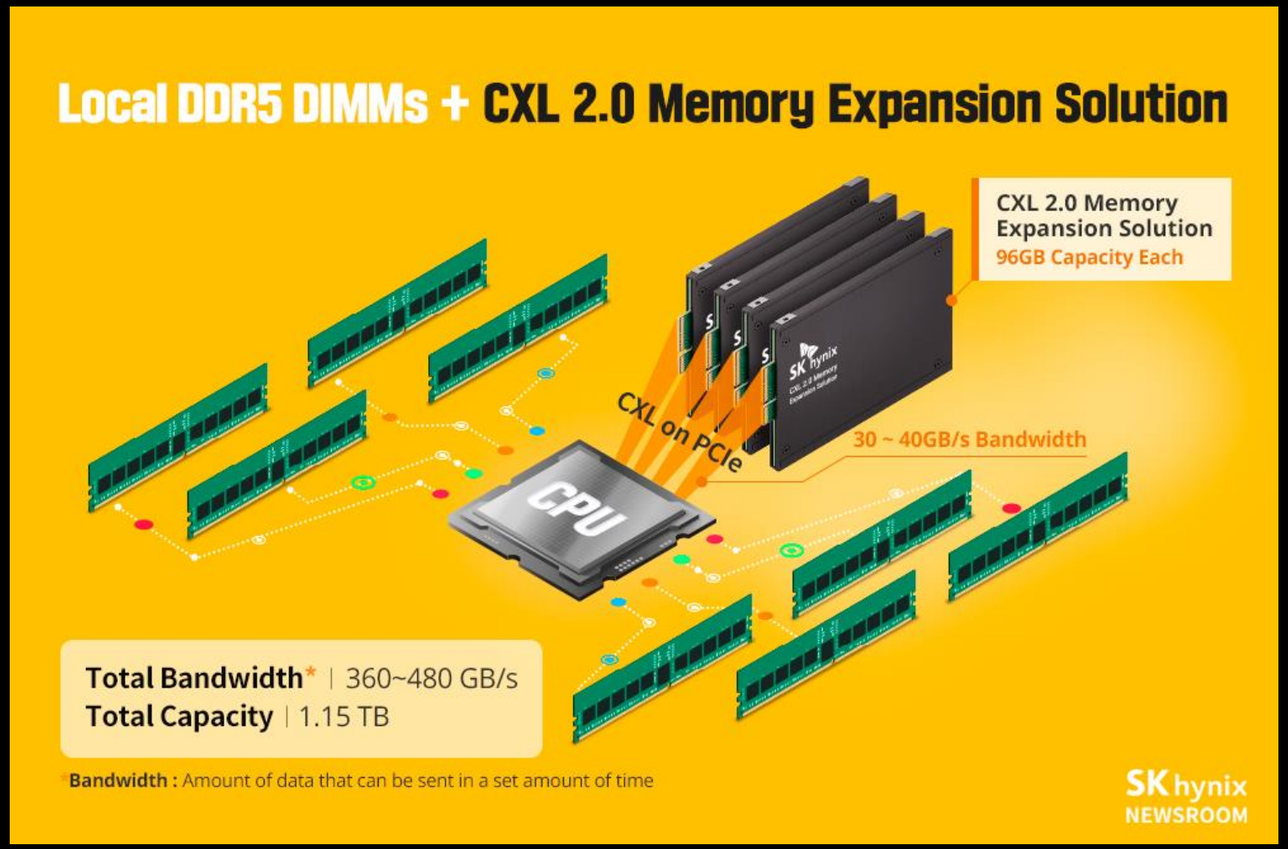
CXL Memory Expander

- **Pros: flexible memory expansion**
  - for both bandwidth and capacity



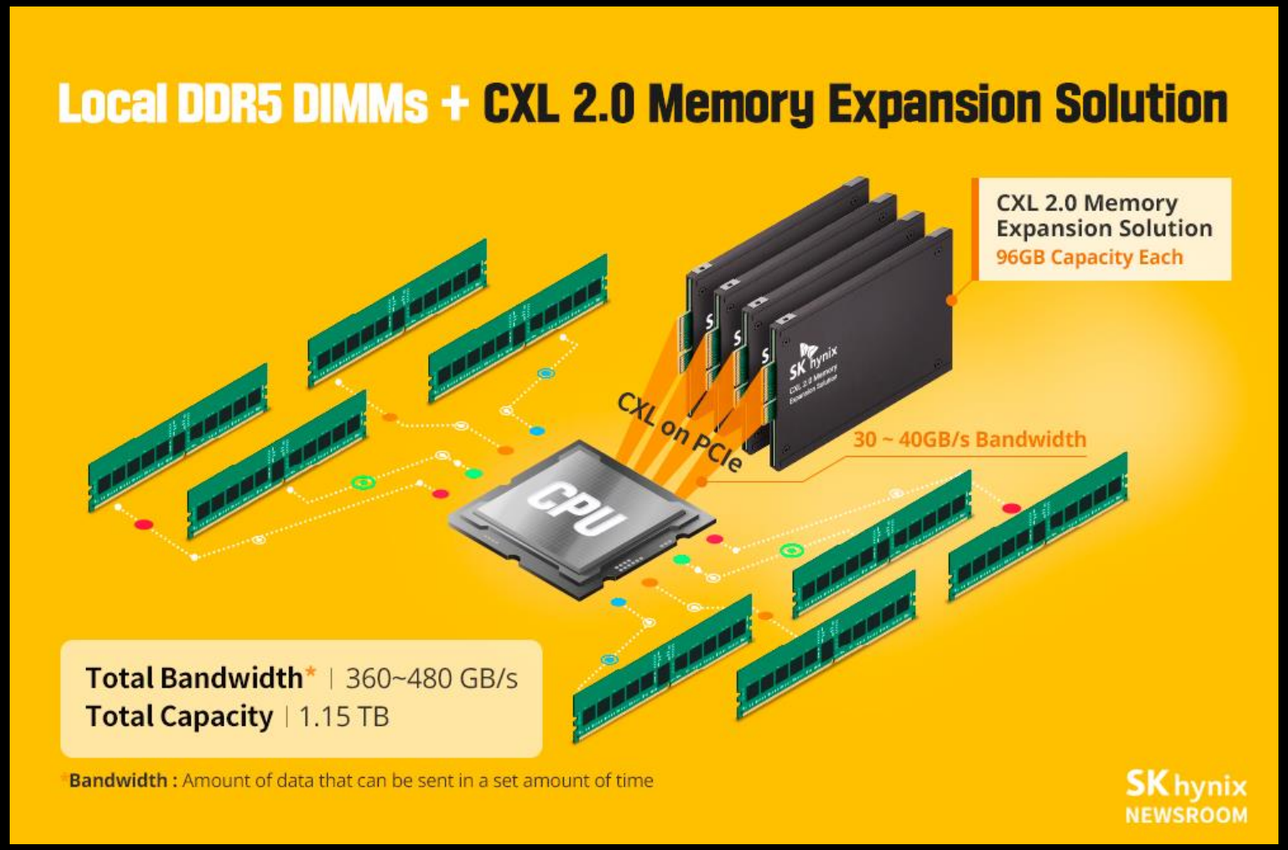
CXL Memory Expander

- **Cons: latency issue**
  - about 2 times longer than local DRAM in DIMM slots



CXL Memory Expander

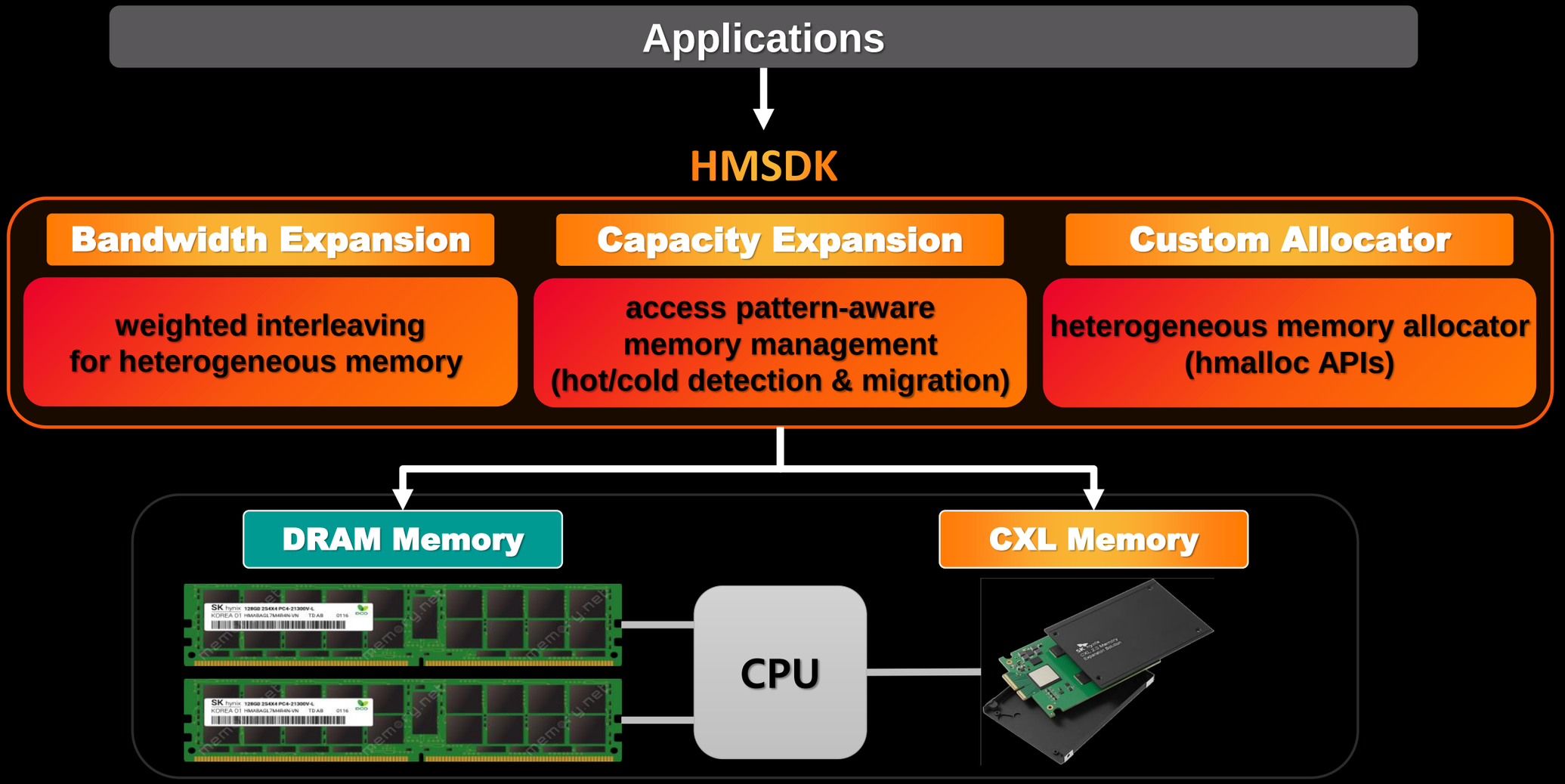
- **Cons: latency issue**
  - about 2 times longer than local DRAM in DIMM slots
  - DAMON based tiered memory management is used for this!



What is HMSDK then?

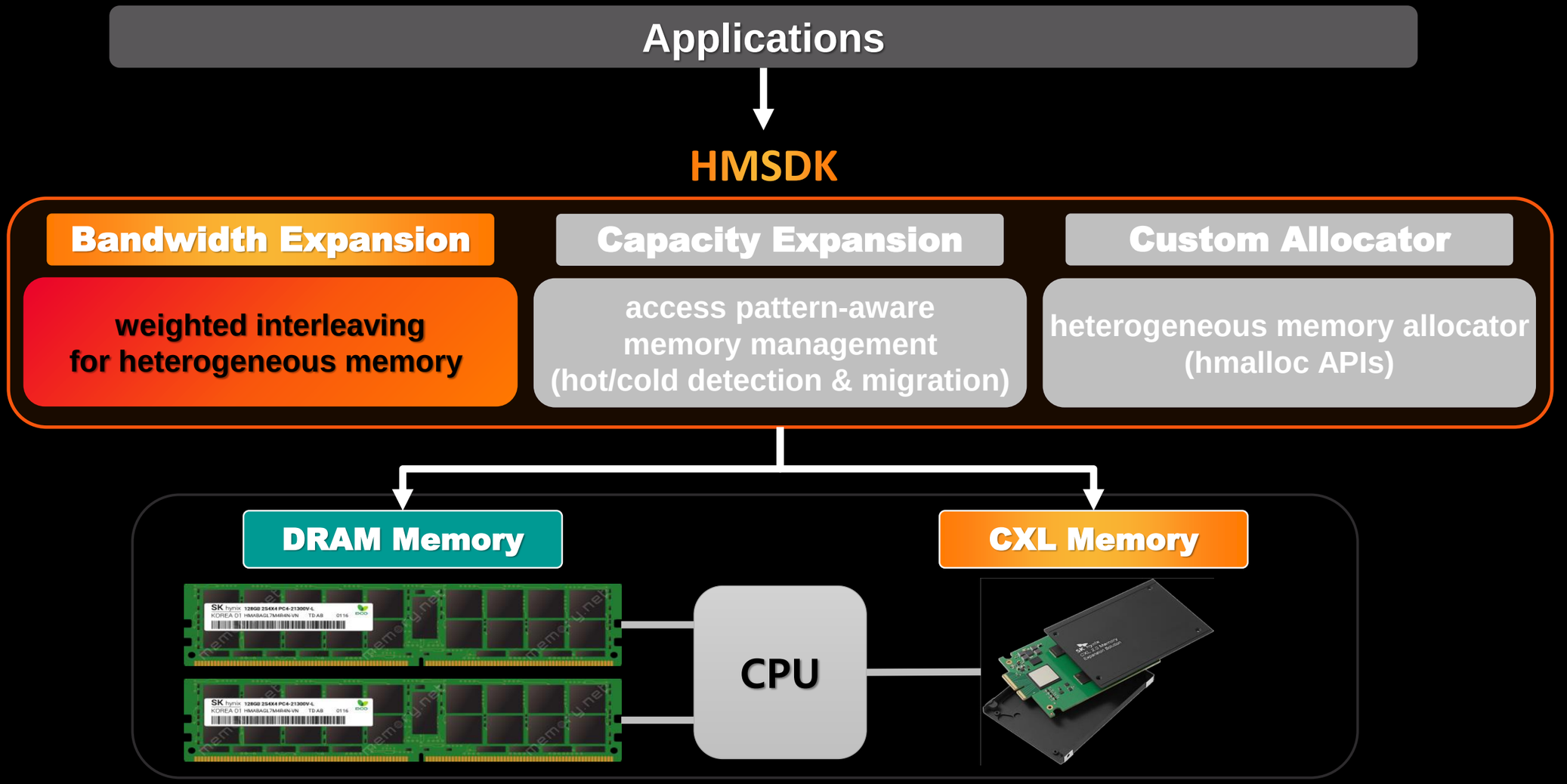
HMSDK for CXL Memory

- Heterogeneous Memory Software Development Kit



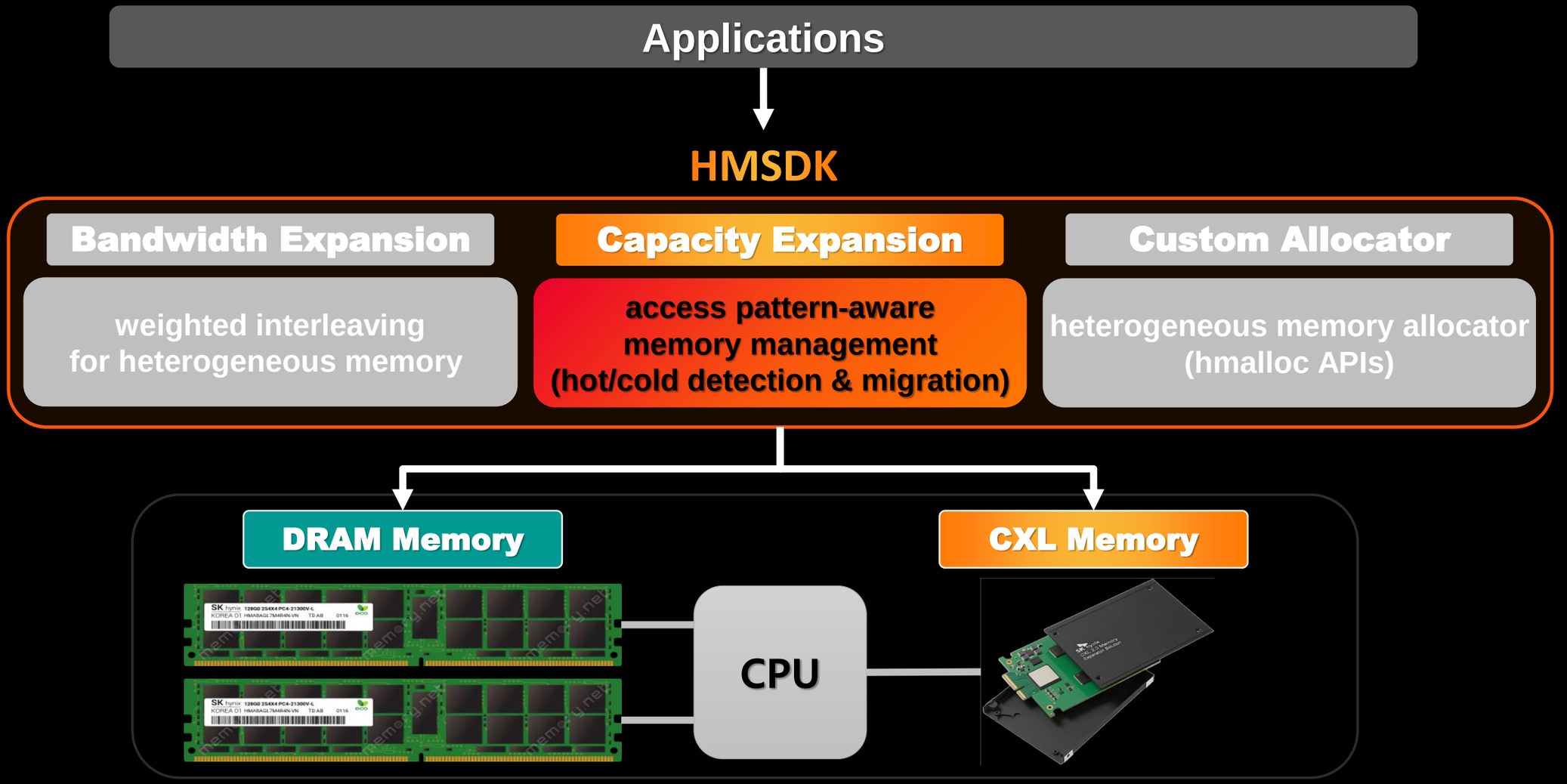
HMSDK for CXL Memory

- **Bandwidth Expansion** is used when target workloads are **bandwidth hungry**.



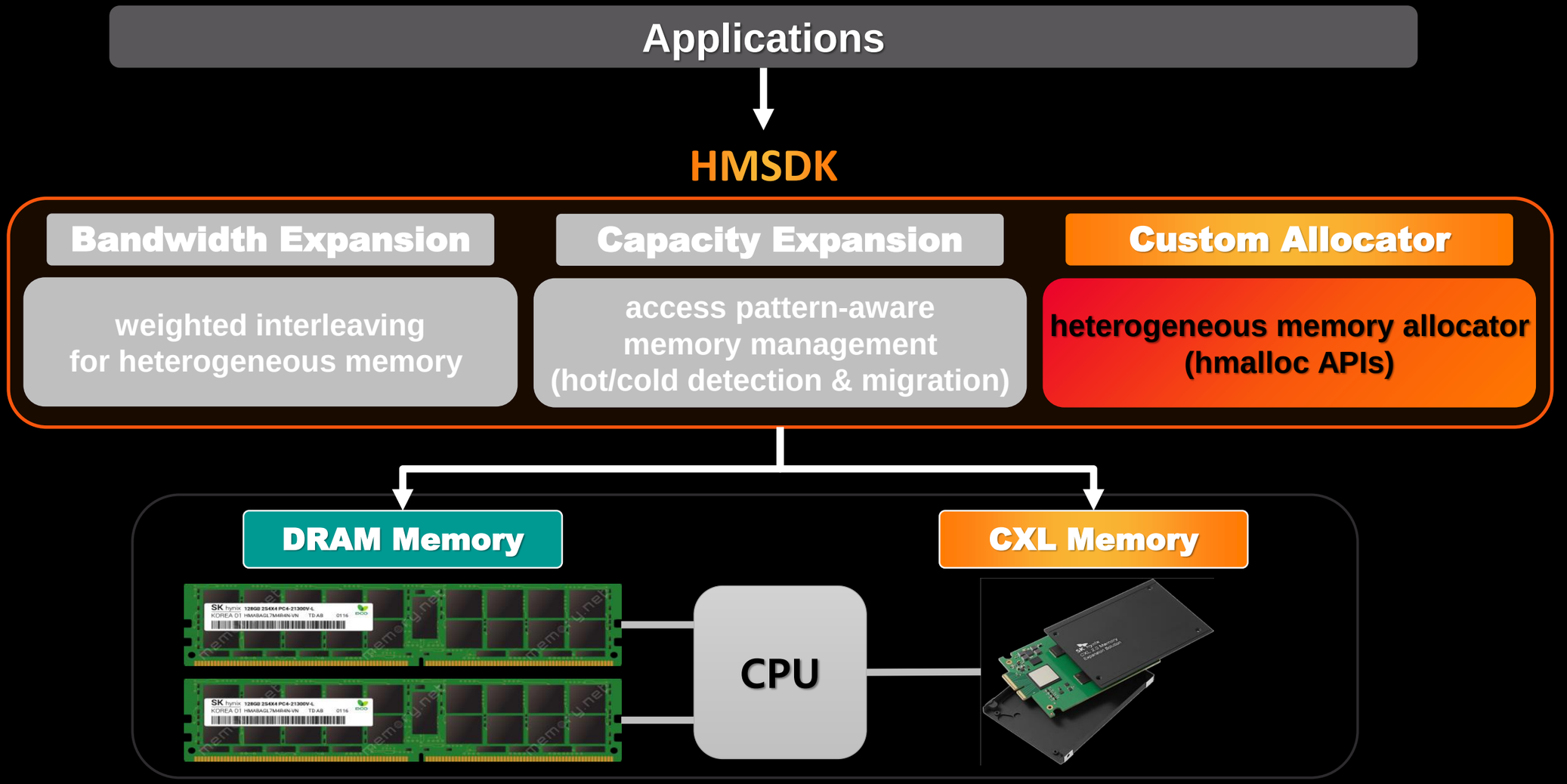
HMSDK for CXL Memory

- **Capacity Expansion** is used when target workloads **needs more capacity**.



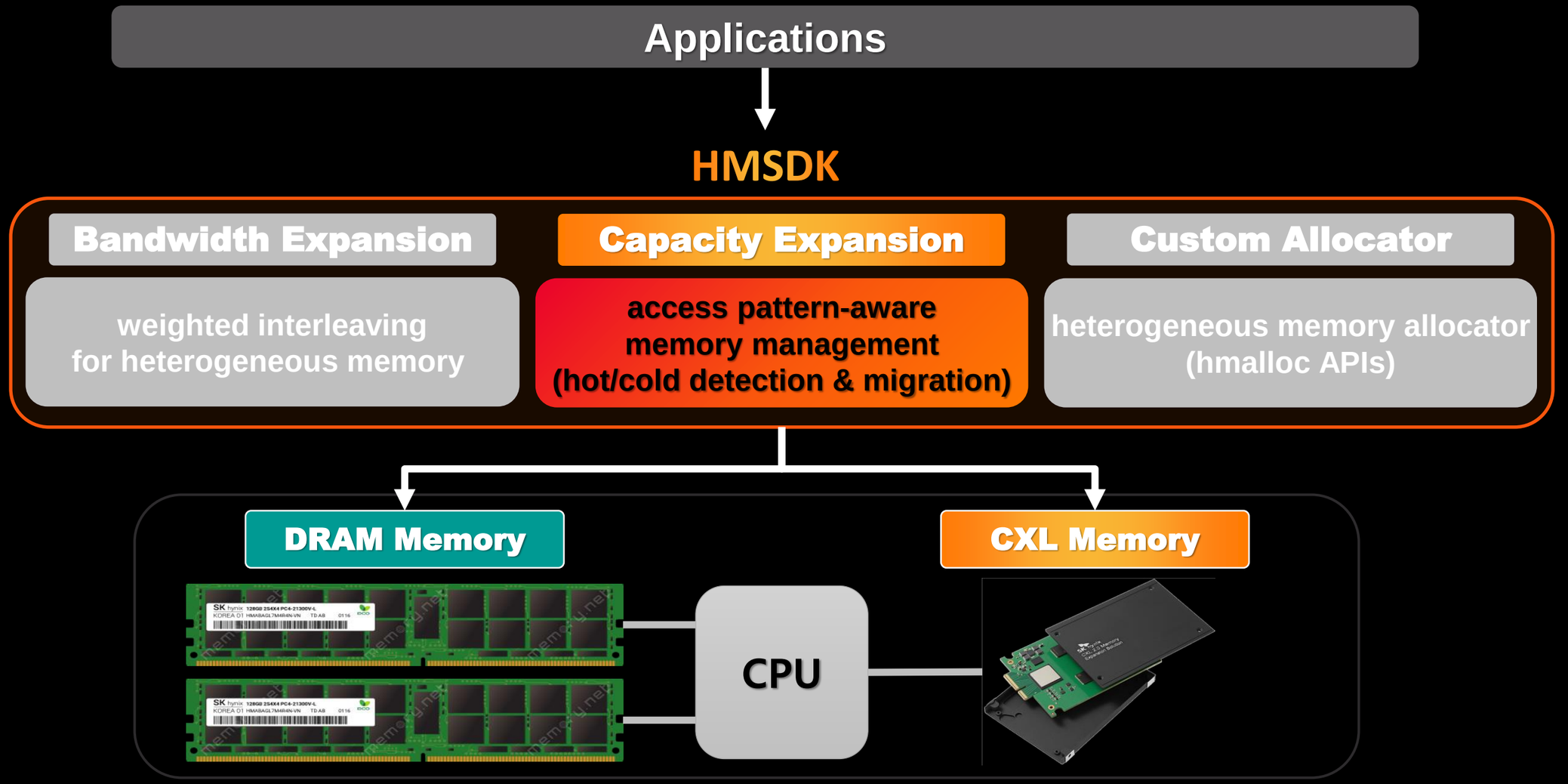
HMSDK for CXL Memory

- **Heterogeneous Memory Allocator** is used when **users have knowledge of their programs**.



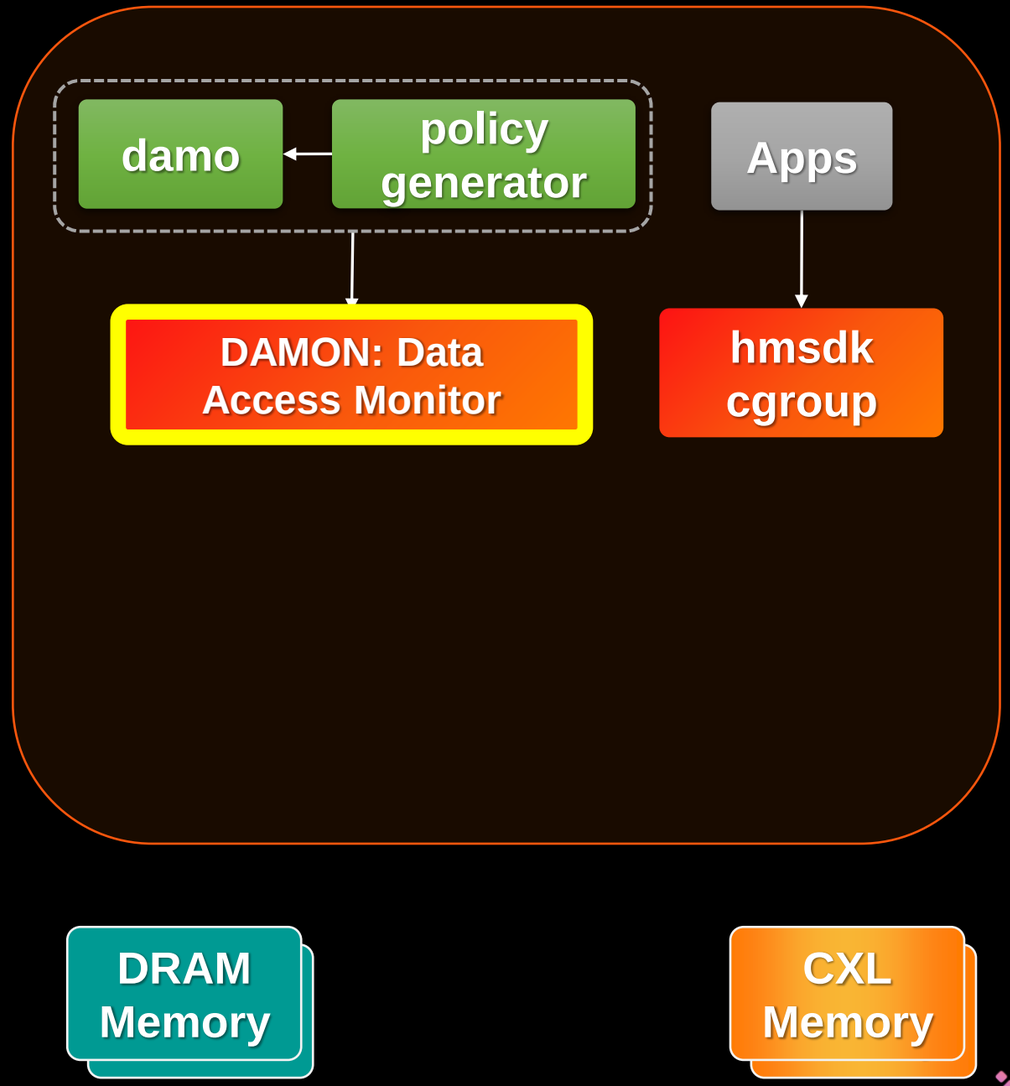
DAMON and DAMOS for Tiered Memory Support

- DAMON for hot/cold detection and DAMOS for migration!



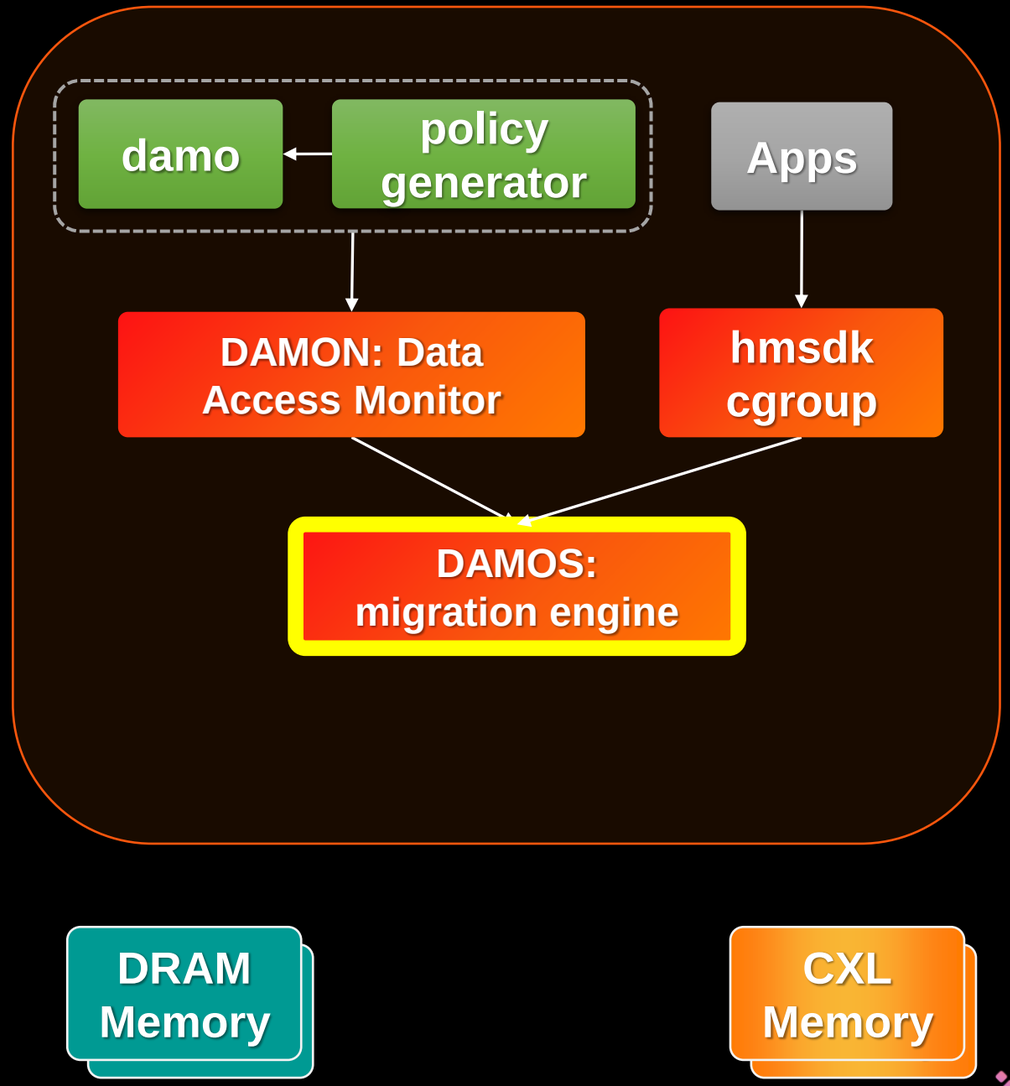
DAMON based Tiered Memory Management

- Memory access profiling is done via DAMON
  - Data Access MONitor



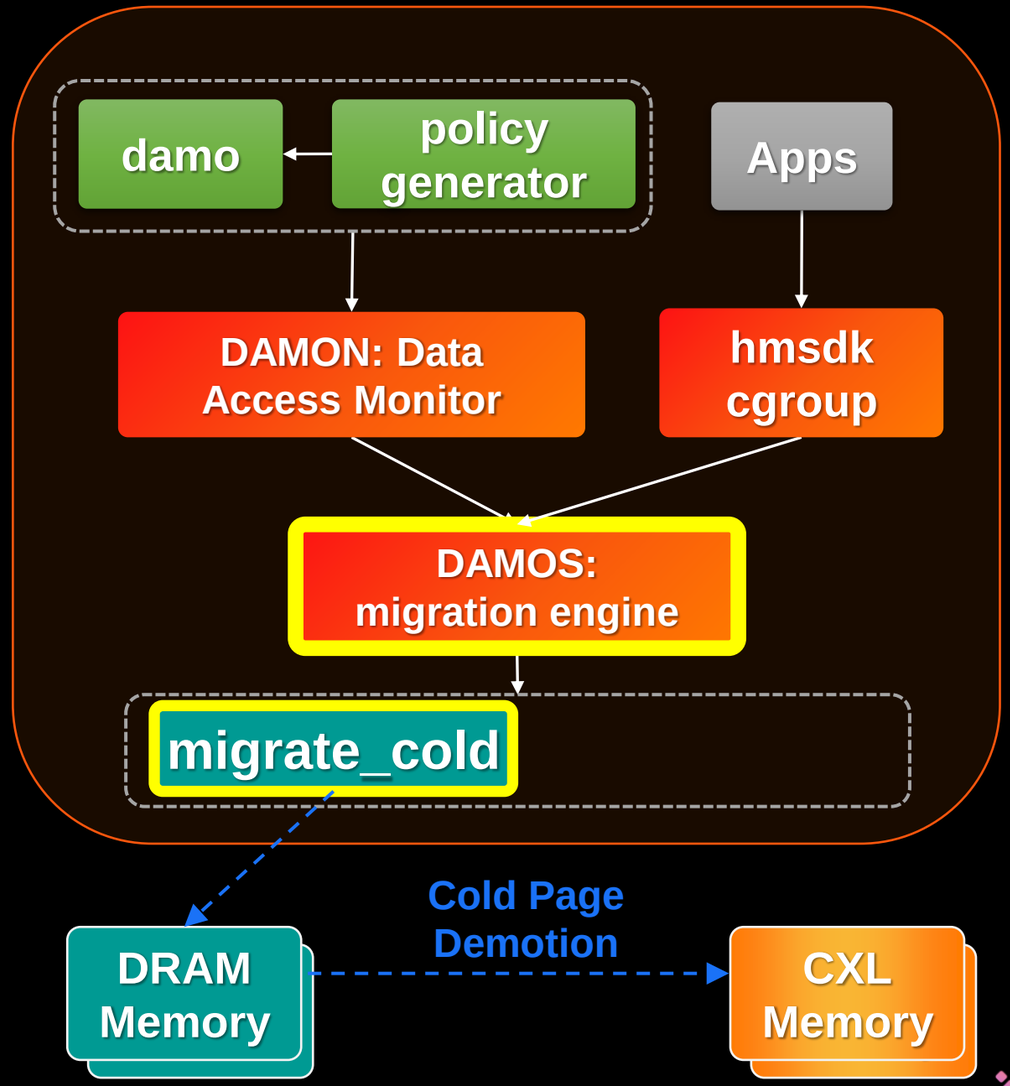
DAMON based Tiered Memory Management

- Memory access profiling is done via DAMON
  - Data Access MONitor
- DAMOS is the migration engine



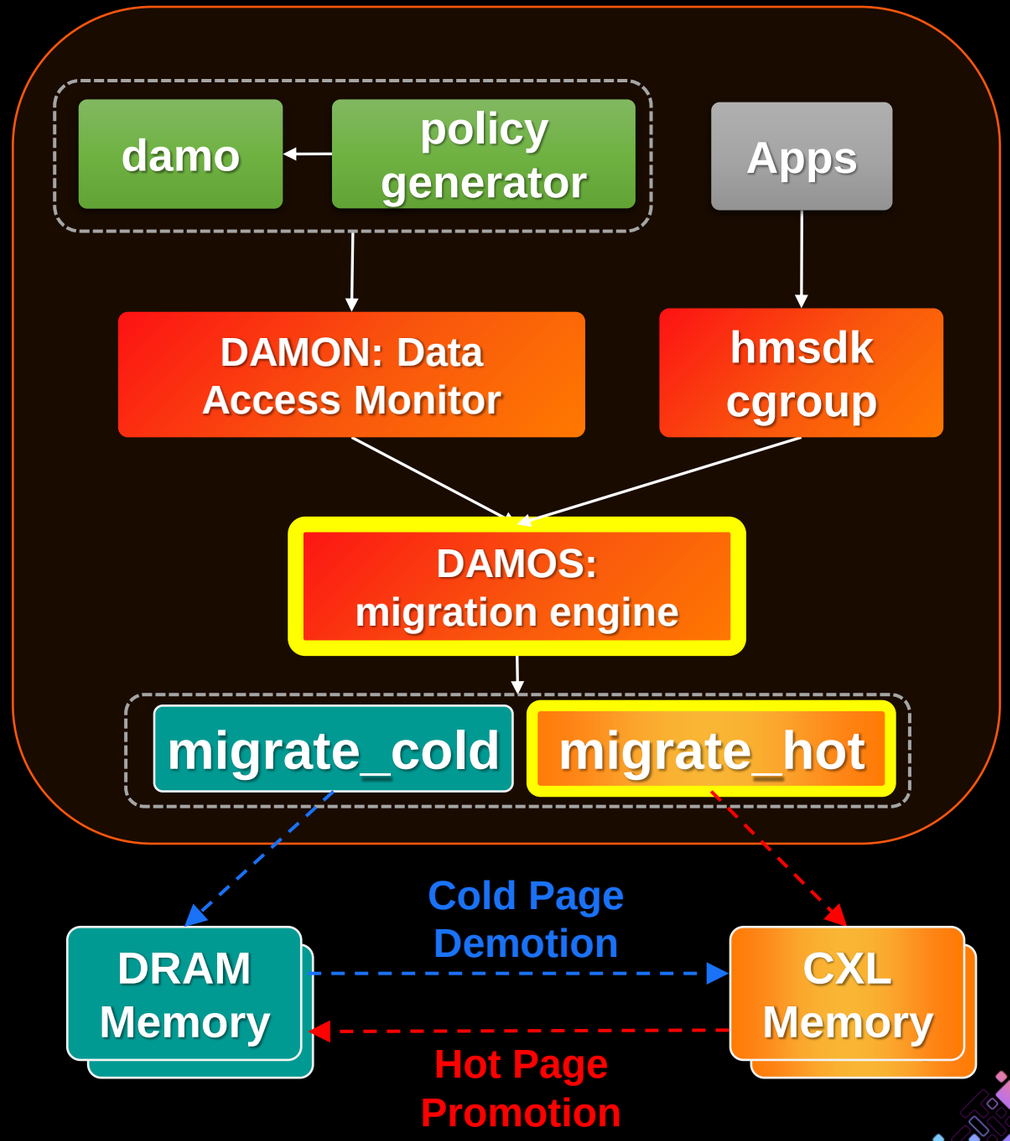
DAMON based Tiered Memory Management

- **Memory access profiling is done via DAMON**
  - Data Access MONitor
- **DAMOS is the migration engine**
  - migrate_cold action
- **Page migration based on access frequency**
  - Cold data goes to CXL memory

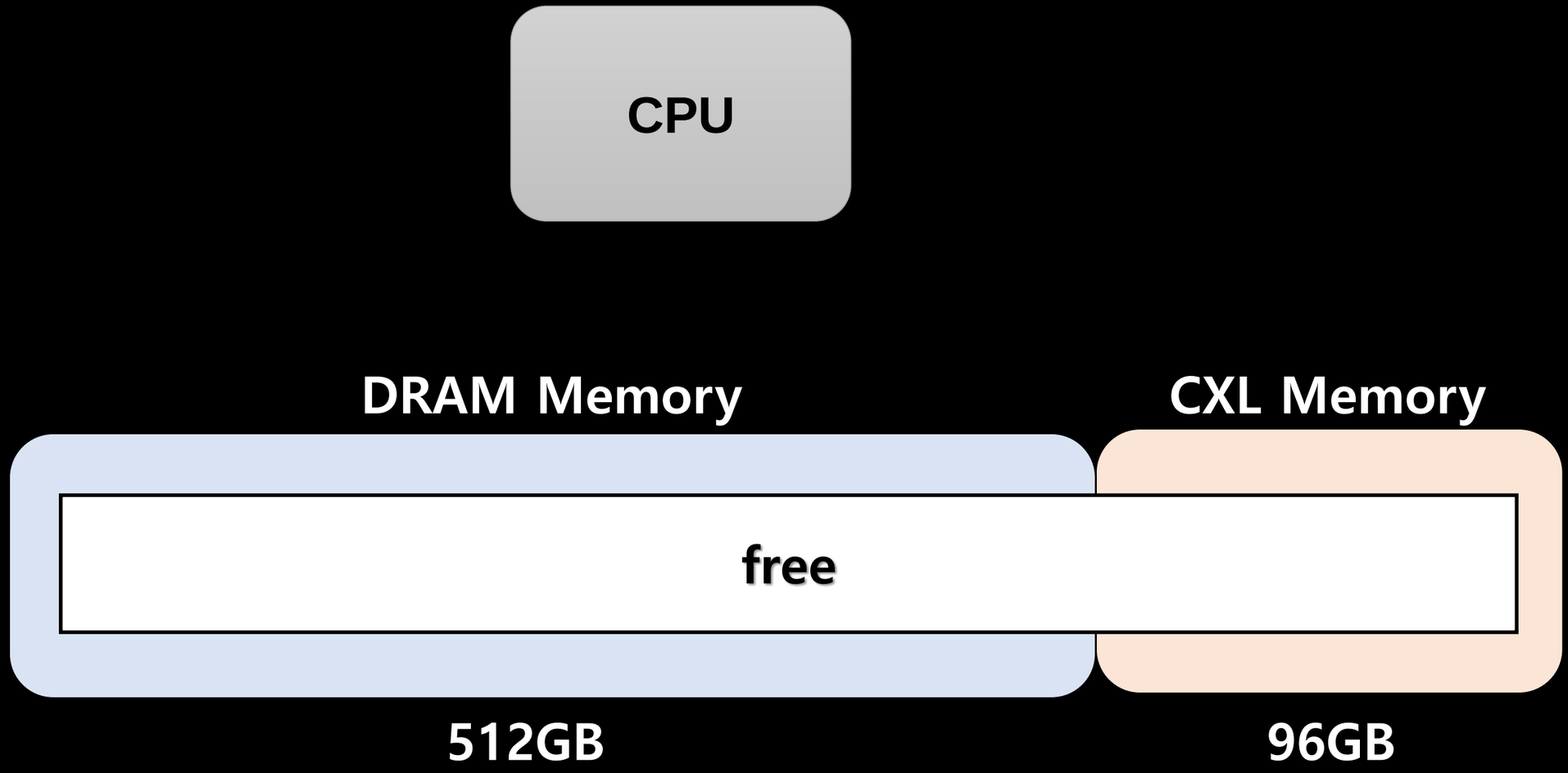


DAMON based Tiered Memory Management

- **Memory access profiling is done via DAMON**
  - Data Access MONitor
- **DAMOS is the migration engine**
  - migrate_cold action
  - migrate_hot action
- **Page migration based on access frequency**
  - Cold data goes to CXL memory
  - Hot data goes to DRAM memory

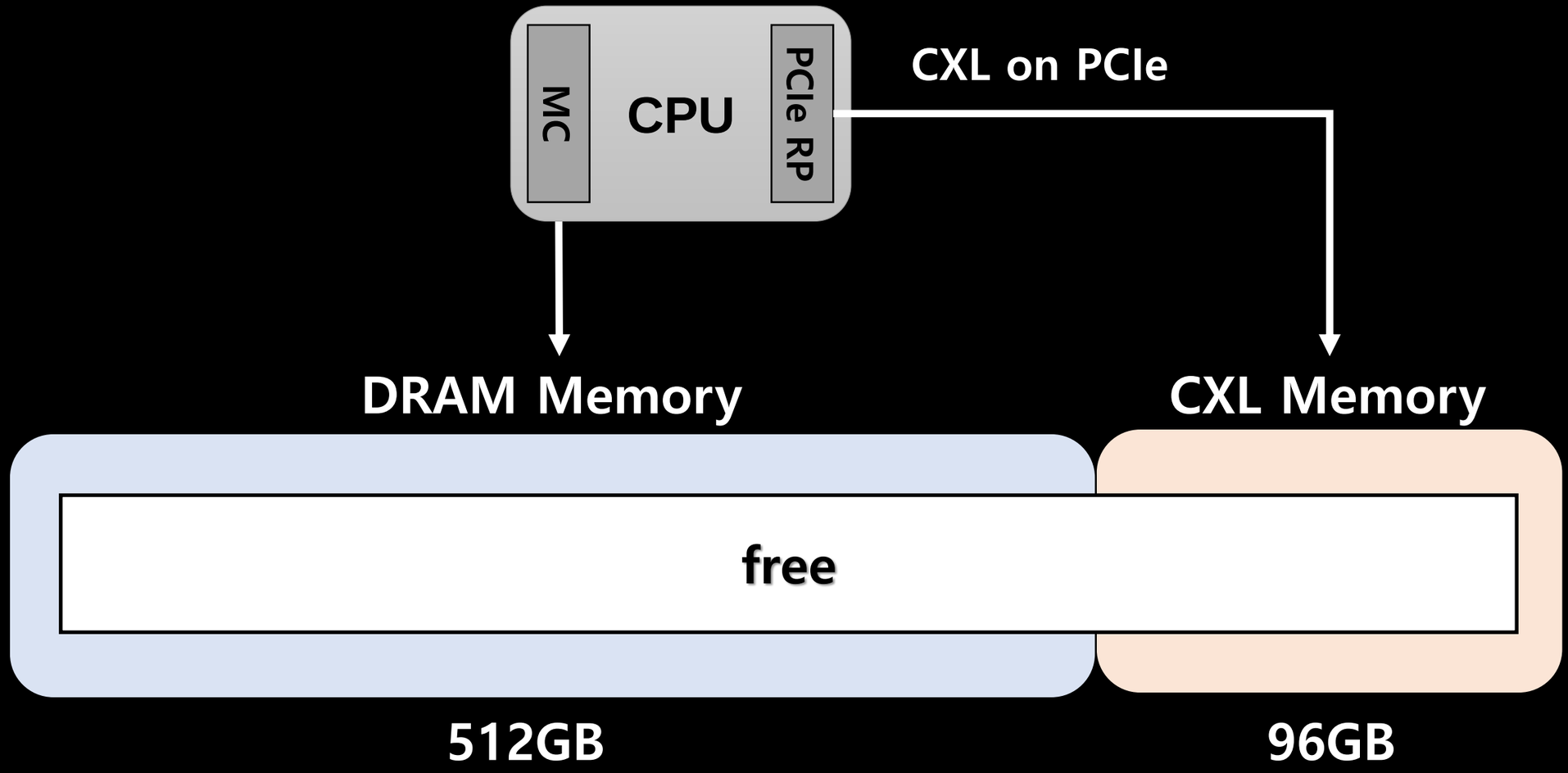


Workload performance based on allocated location



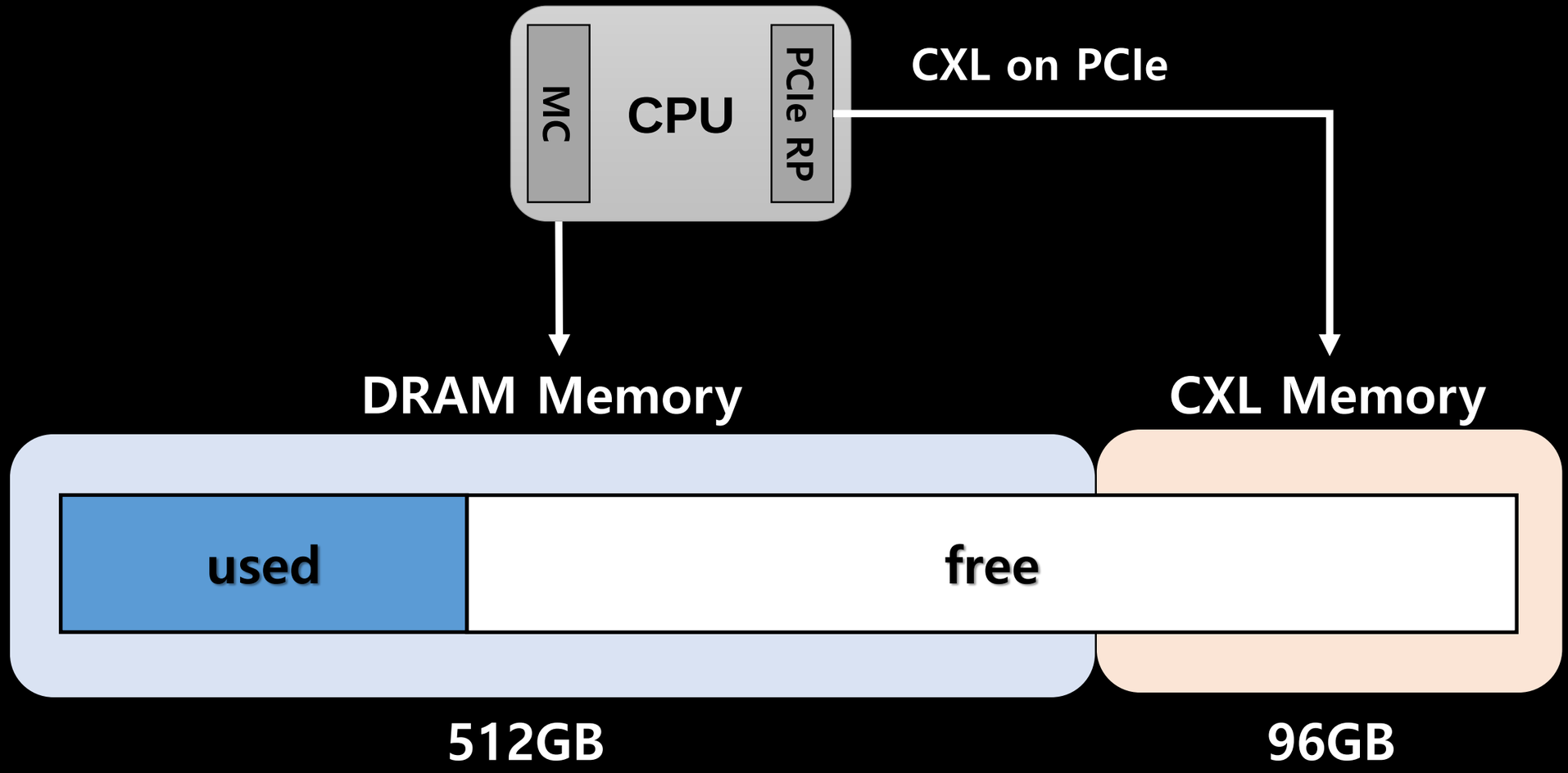
Workload performance based on allocated location

- Different Memory Access Path



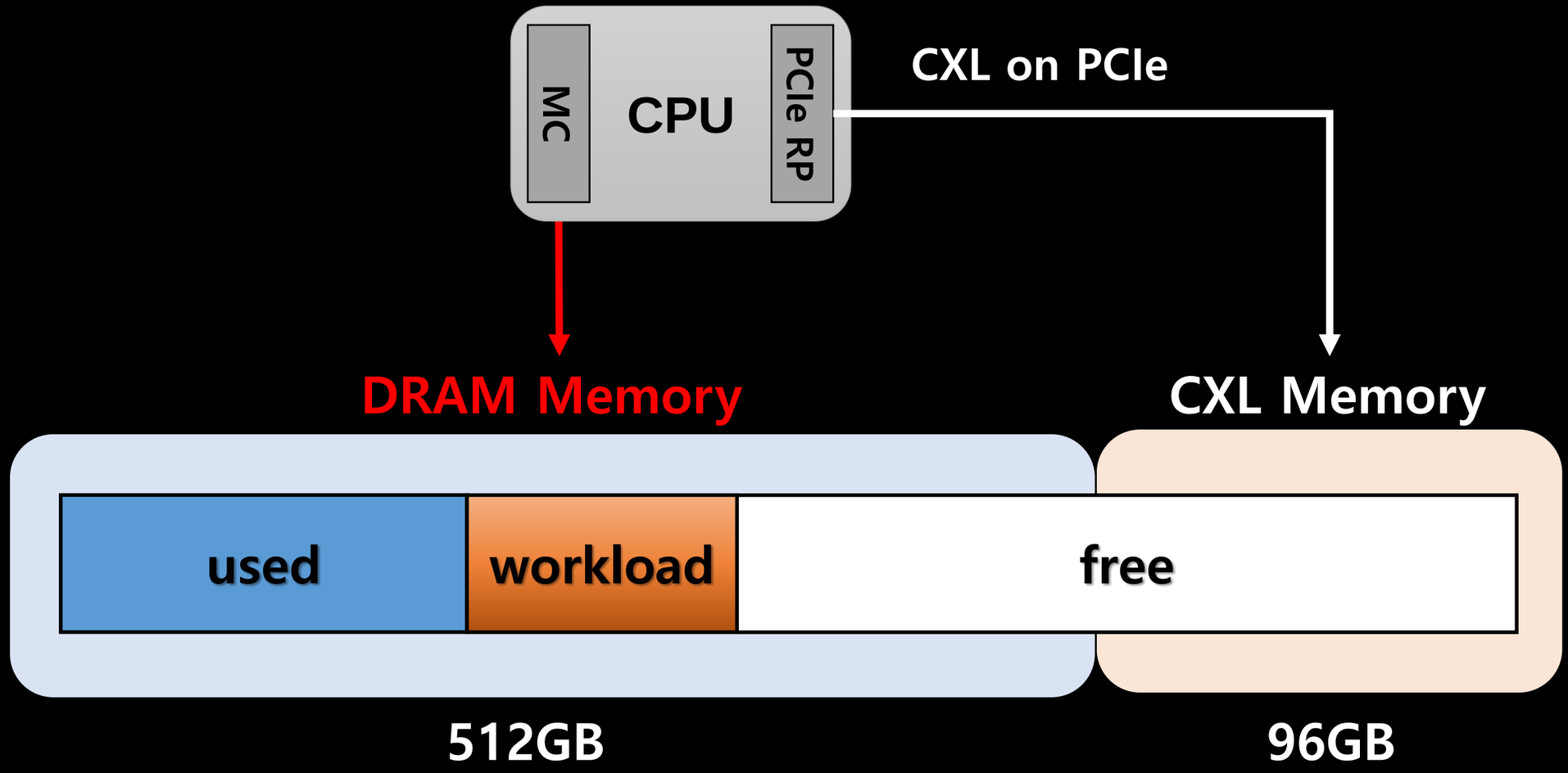
Workload performance based on allocated location

- Partial memory space can be used by others



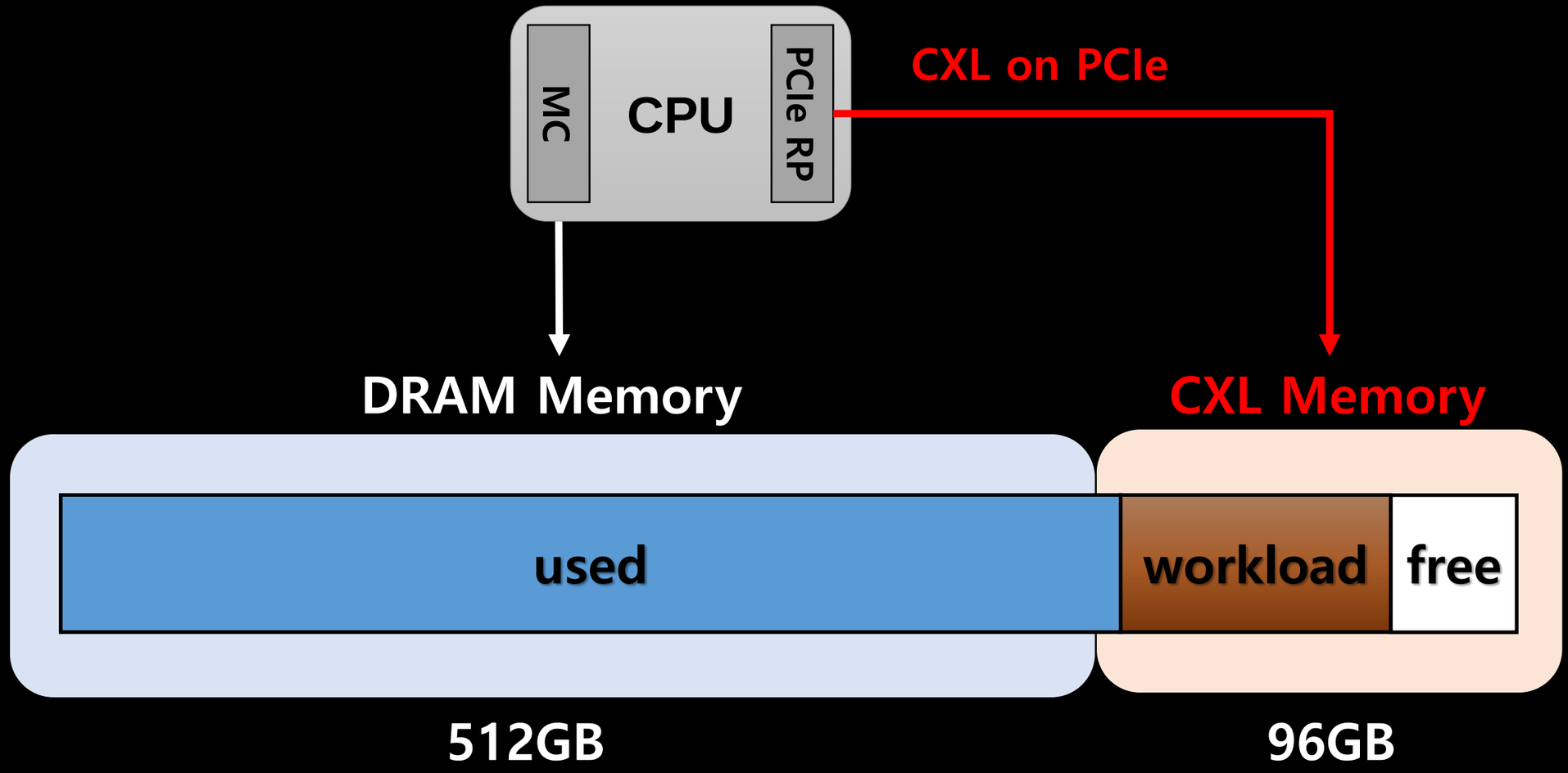
Workload performance based on allocated location

- **Case 1: The workload fully fits into fast DRAM.**
  - Fastest execution case



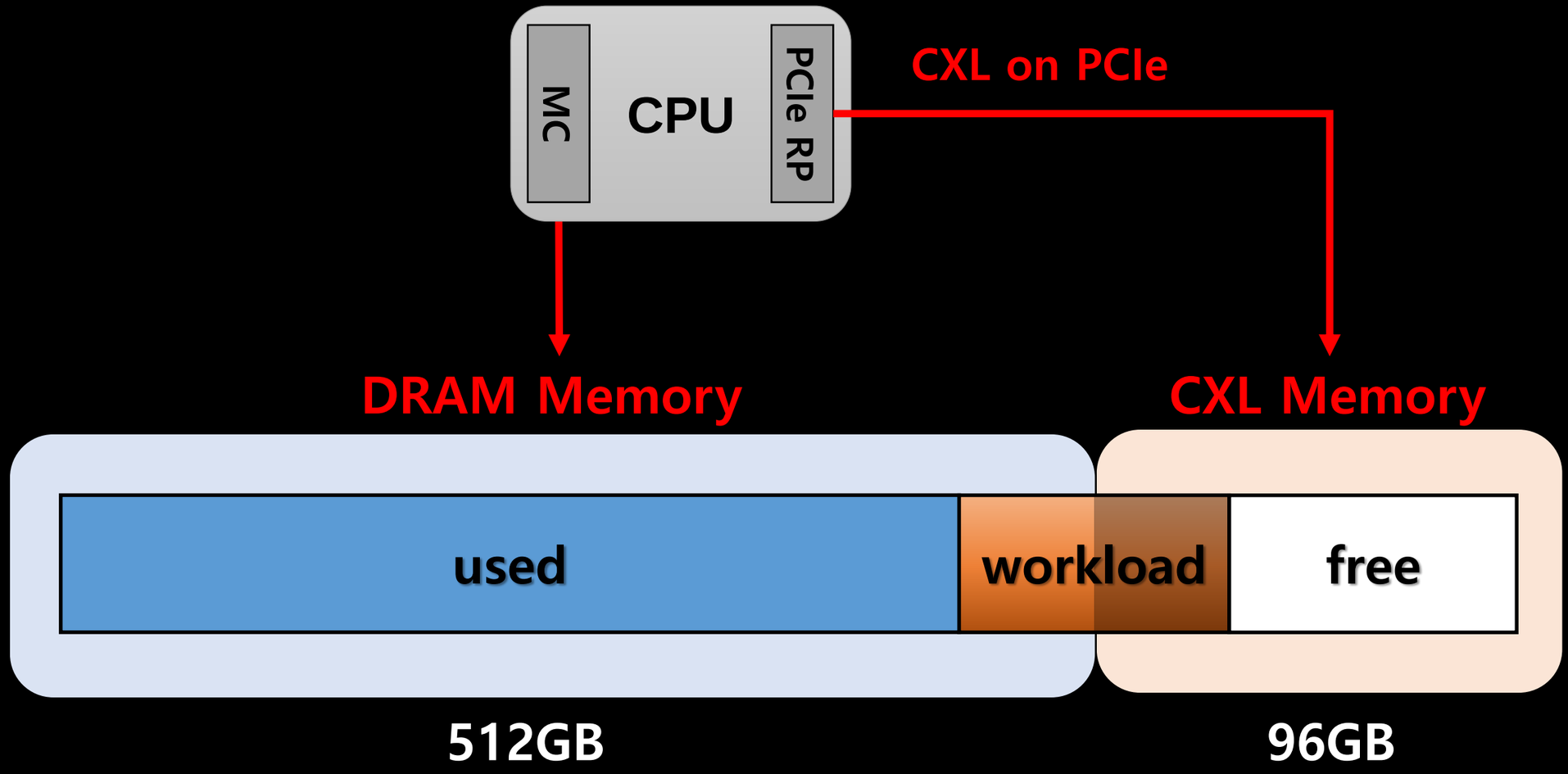
Workload performance based on allocated location

- **Case 2: The workload fully allocated in CXL memory.**
  - Slowest execution case



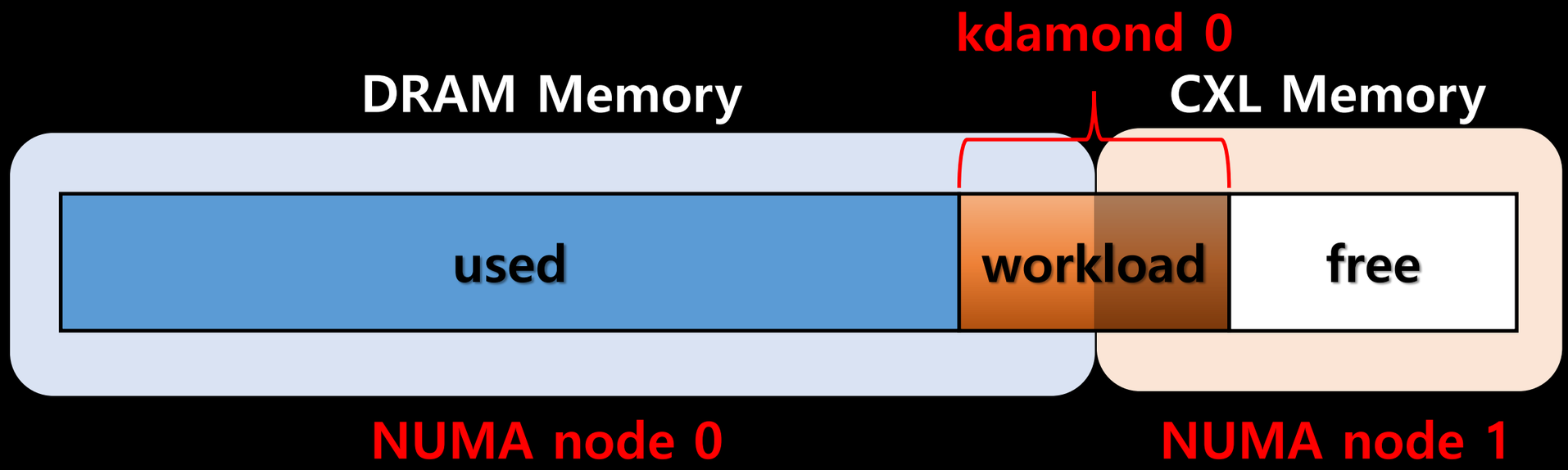
Workload performance based on allocated location

- **Case 3: The workload allocated in both DRAM and CXL.**
  - Partial slowdown from the data of workload on CXL memory



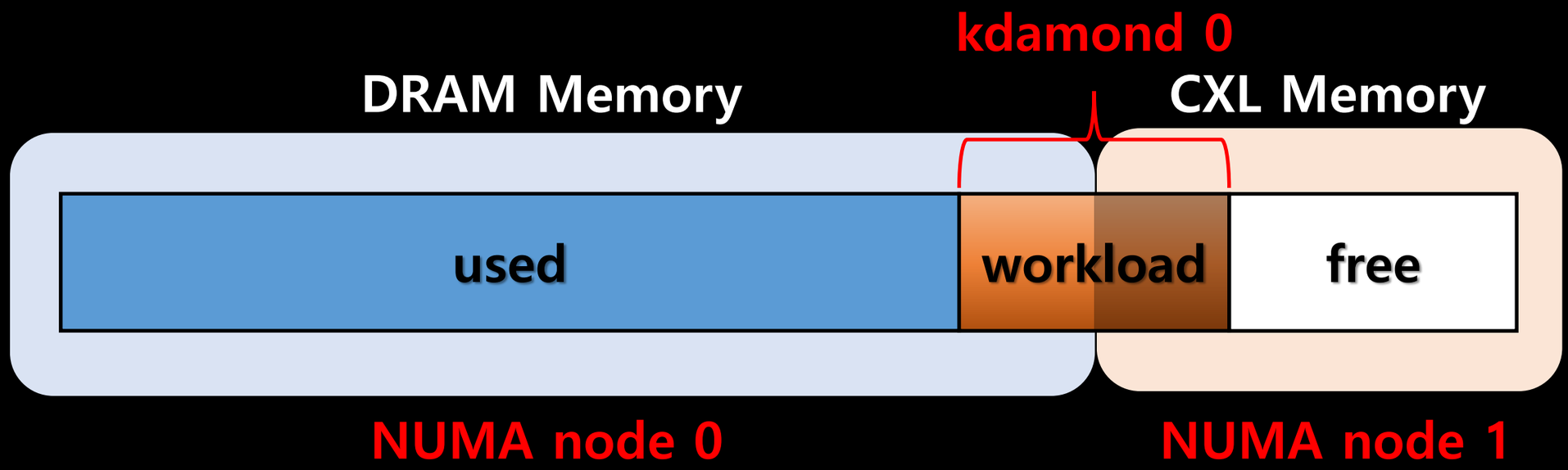
DAMON Usage (virtual address mode)

- vaddr mode has to find hot/cold only inside a single process



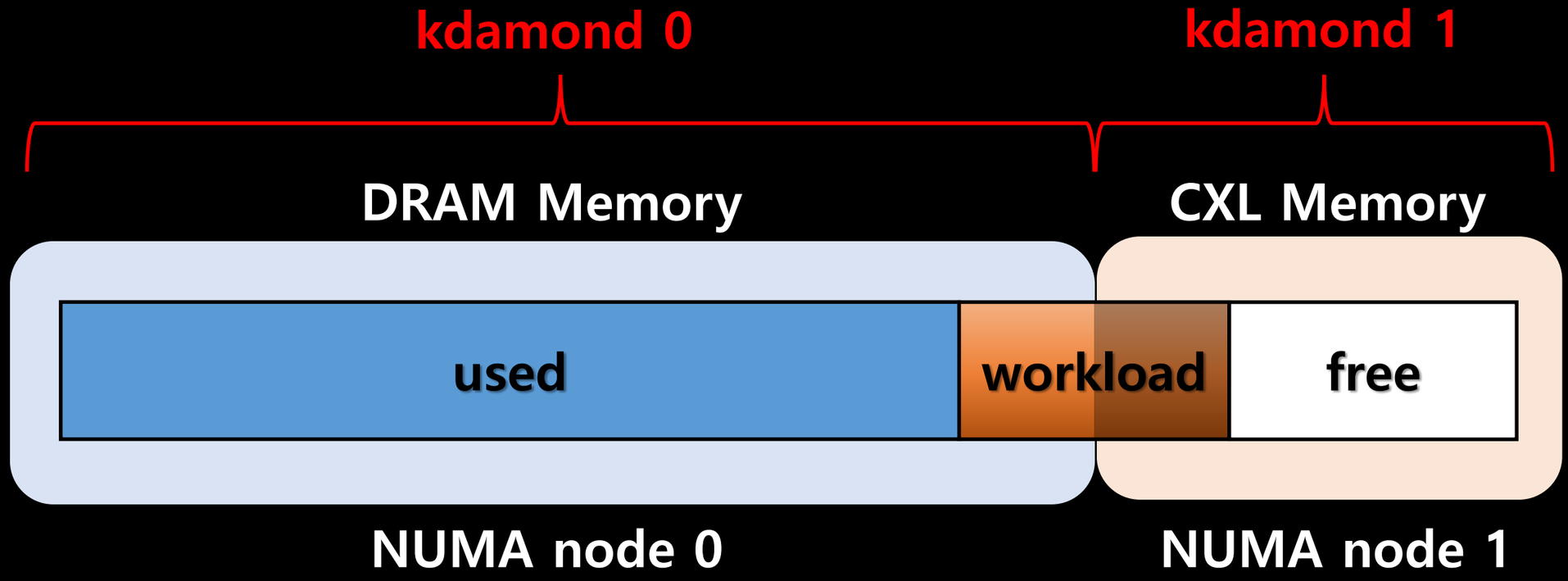
DAMON Usage (virtual address mode)

- vaddr mode has to find hot/cold only inside a single process
- Not very helpful in systems with huge memory capacity!
  - too many kdamonds are needed for multiple processes



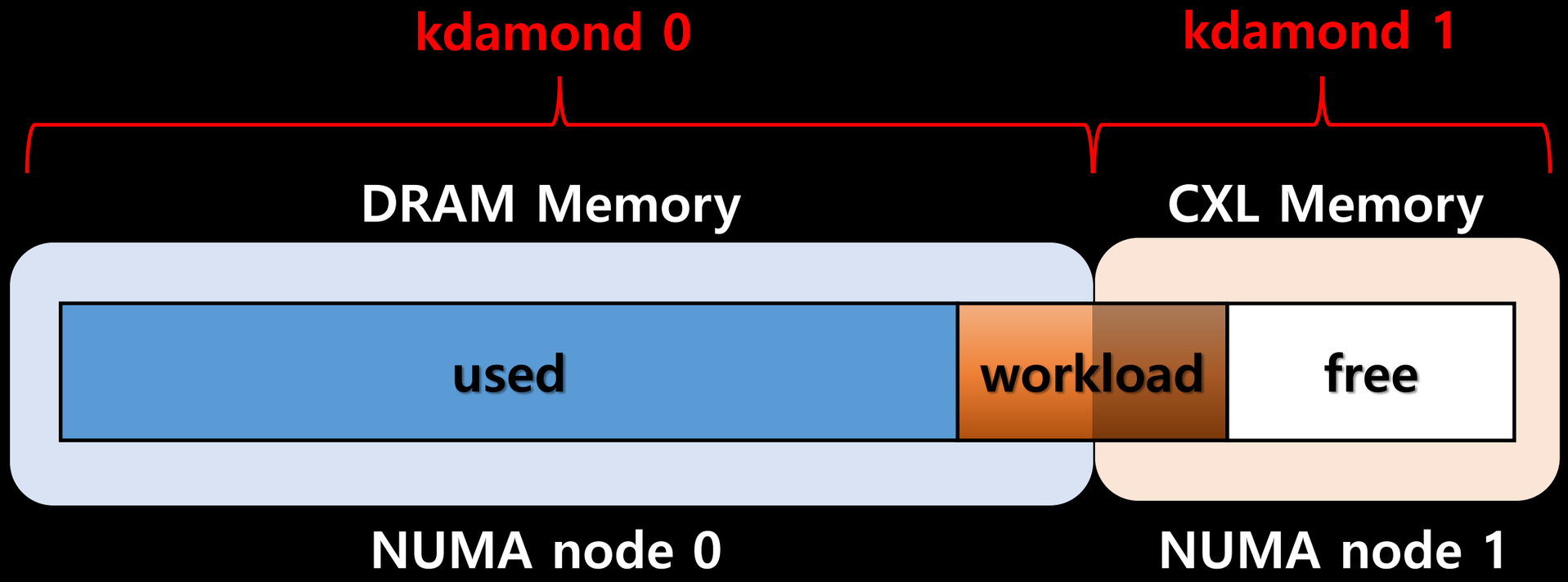
DAMON Usage (physical address mode)

- In paddr mode, separate kdamonds can monitor each NUMA node
  - System-wide monitoring and DAMOS actions



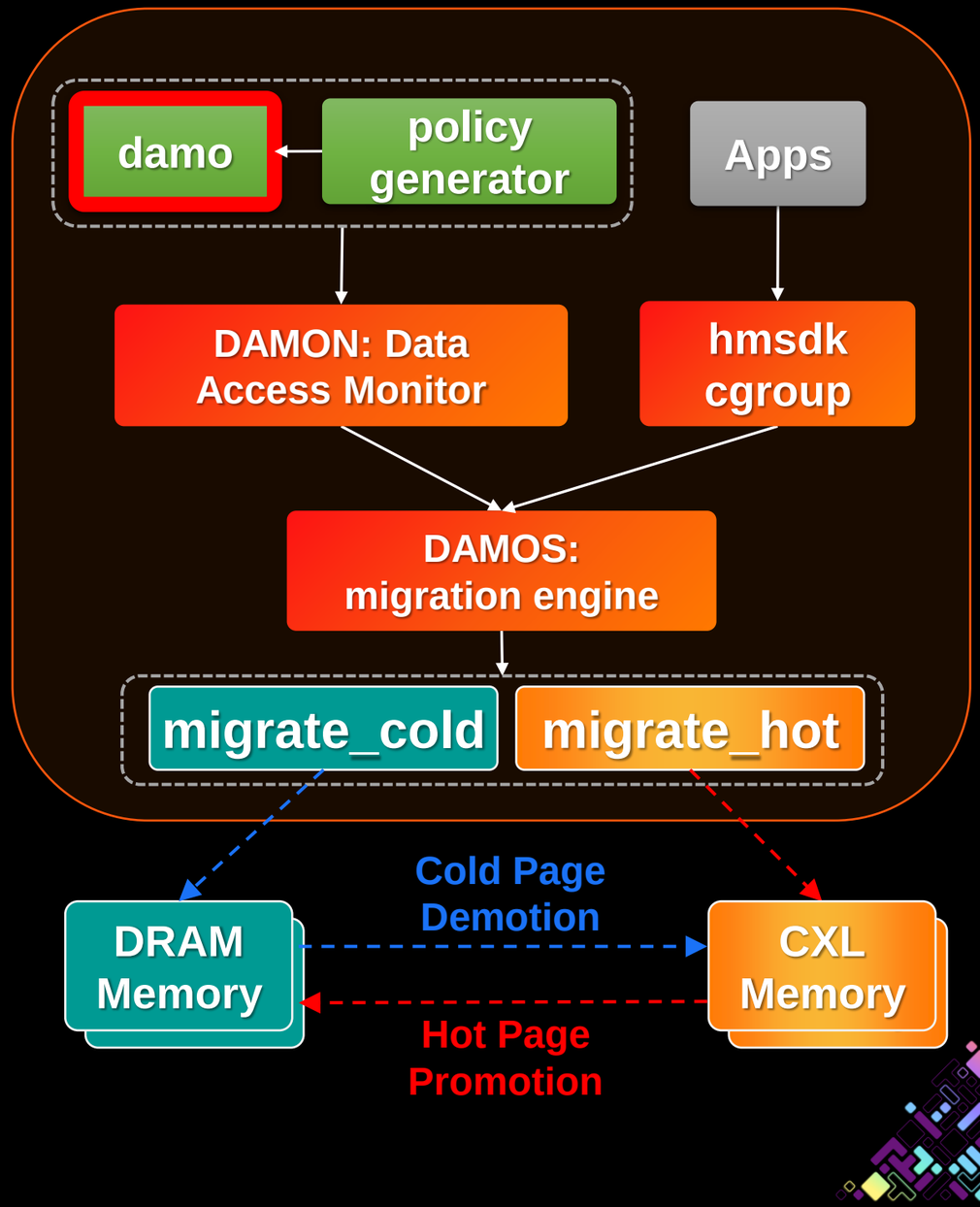
DAMON Usage (physical address mode)

- **In paddr mode, separate kdamonds can monitor each NUMA node**
  - System-wide monitoring and DAMOS actions
  - Much easier finding cold data from idle processes



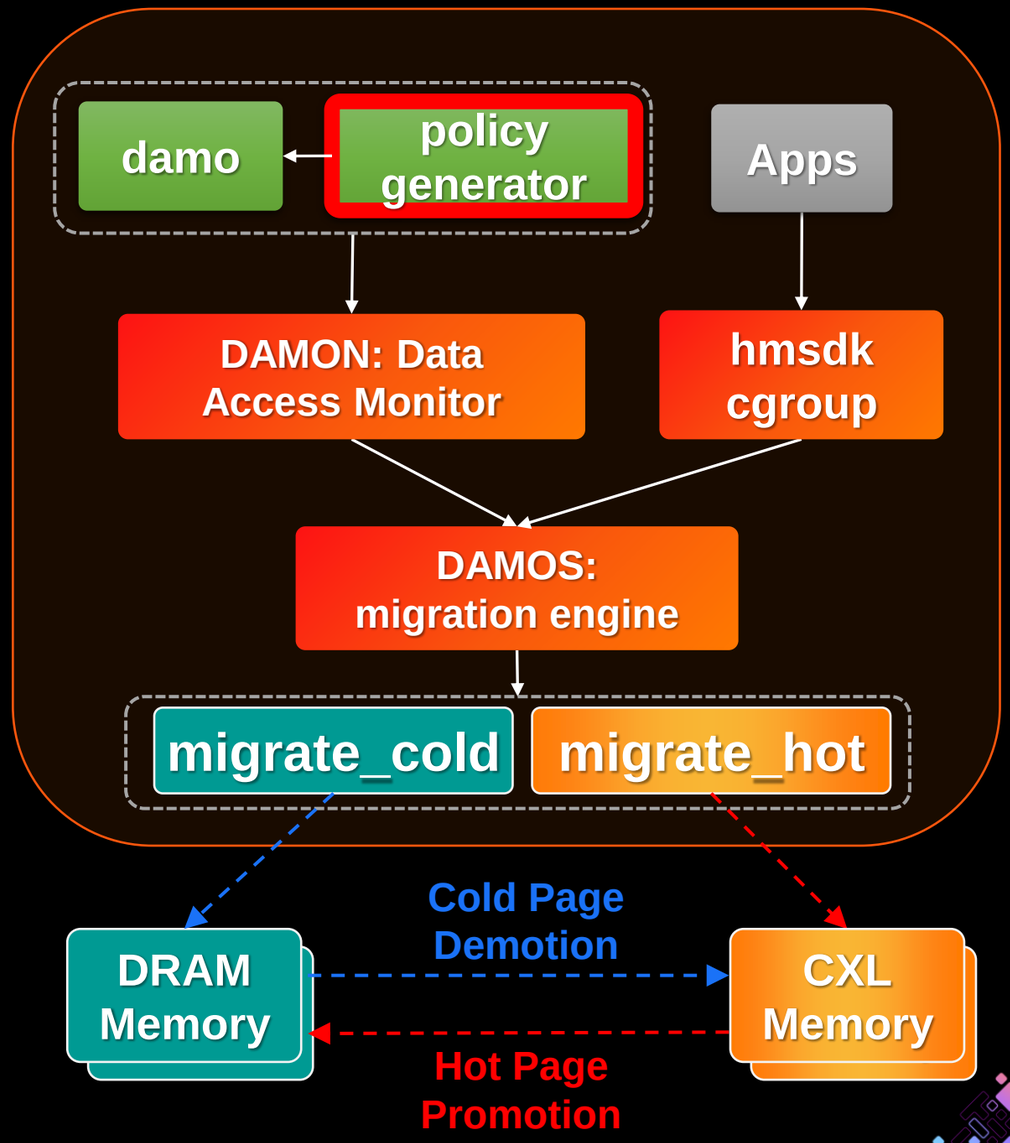
DAMON based Tiered Memory Management

- damo is the userspace tool
  - controls all the DAMON sysfs knobs



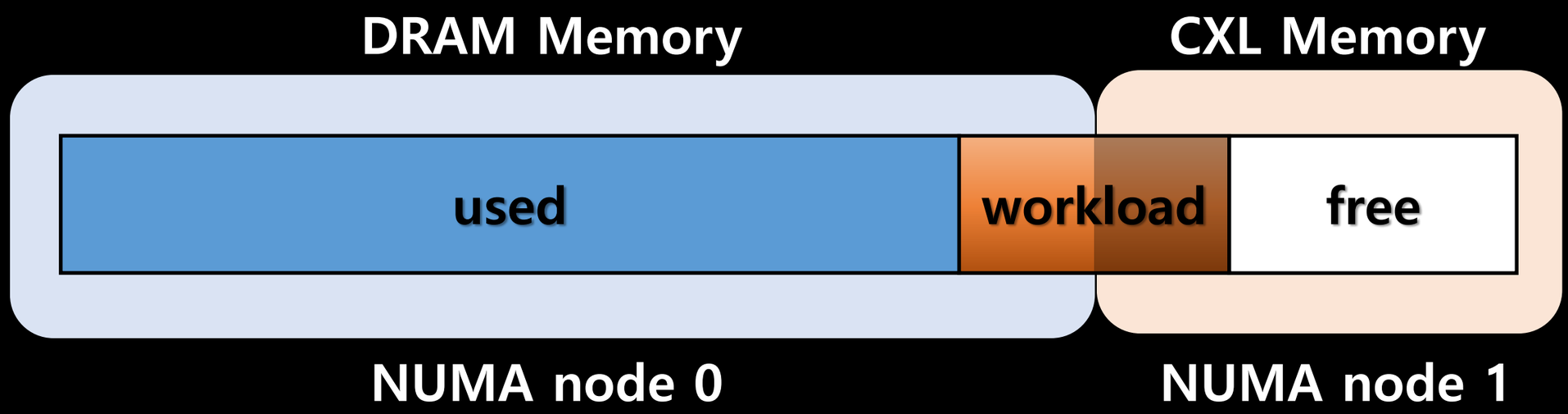
DAMON based Tiered Memory Management

- **damo is the userspace tool**
  - controls all the DAMON sysfs knobs
- **HMSDK policy generator**
  - generates kdamond operation policy
    - for each **migrate_{hot,cold}**
  - each scheme based on NUMA node
  - hmsdk/tools/gen_migpol.py



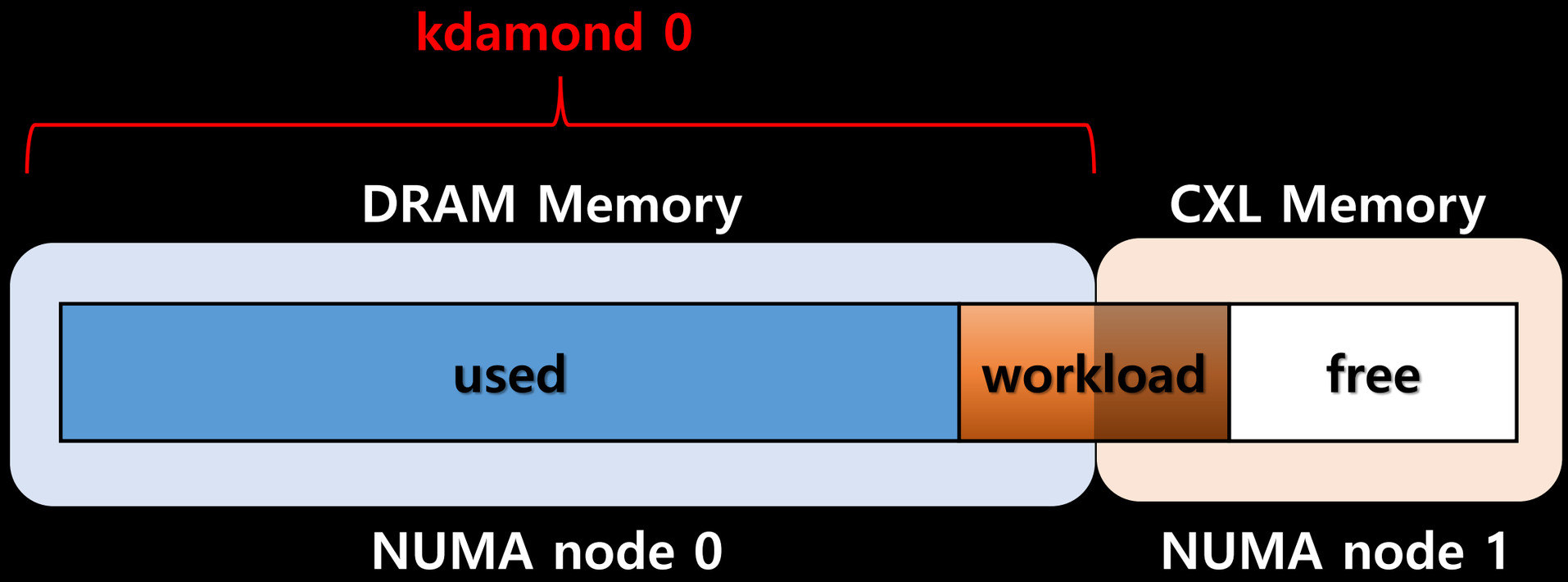
DAMON Usage

- **gen_migpol.py usage**
  - ./tools/gen_migpol.py -o hmsdk.yaml



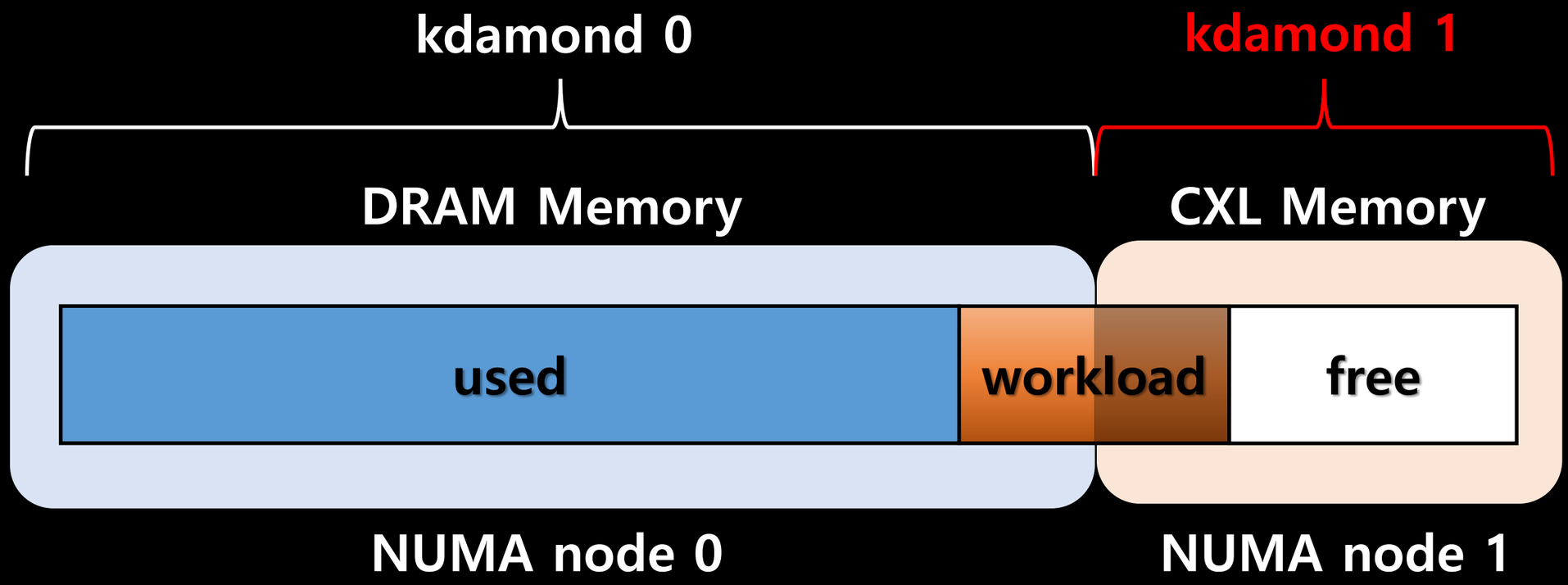
DAMON Usage

- **gen_migpol.py usage**
  - ./tools/gen_migpol.py **--demote 0 1** -o hmsdk.yaml



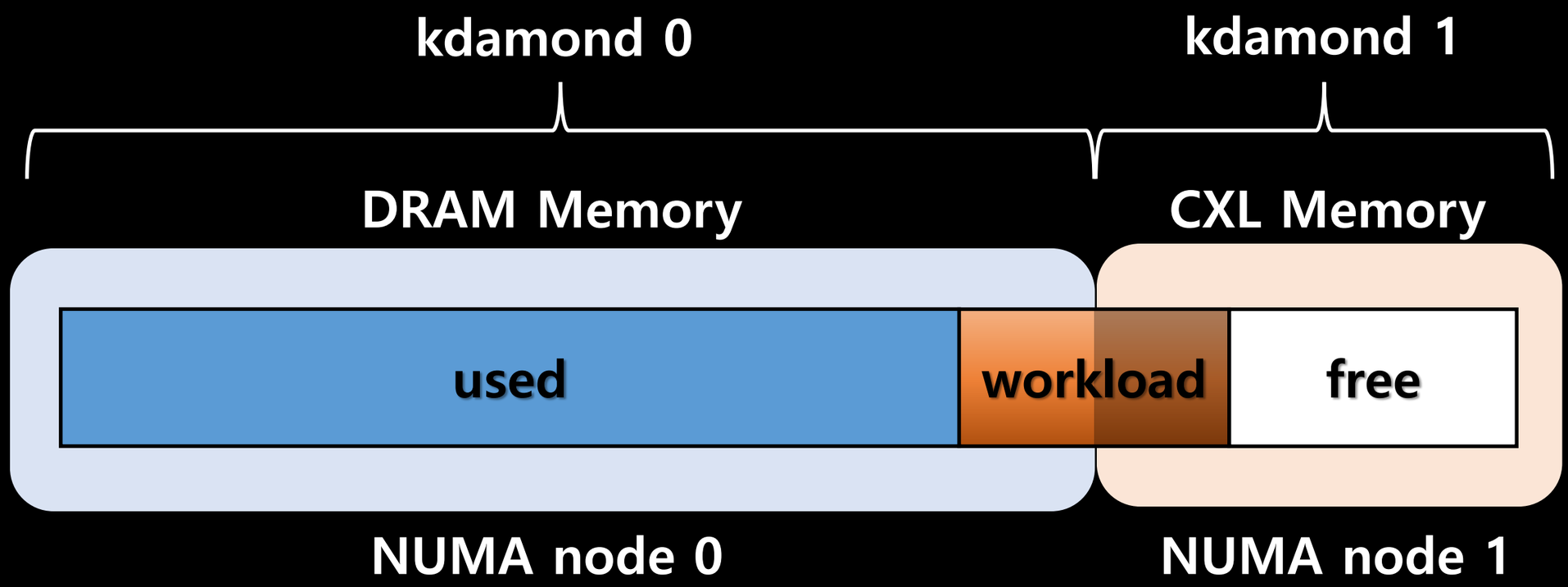
DAMON Usage

- **gen_migpol.py usage**
  - ./tools/gen_migpol.py --demote 0 1 **--promote 1 0** -o hmsdk.yaml



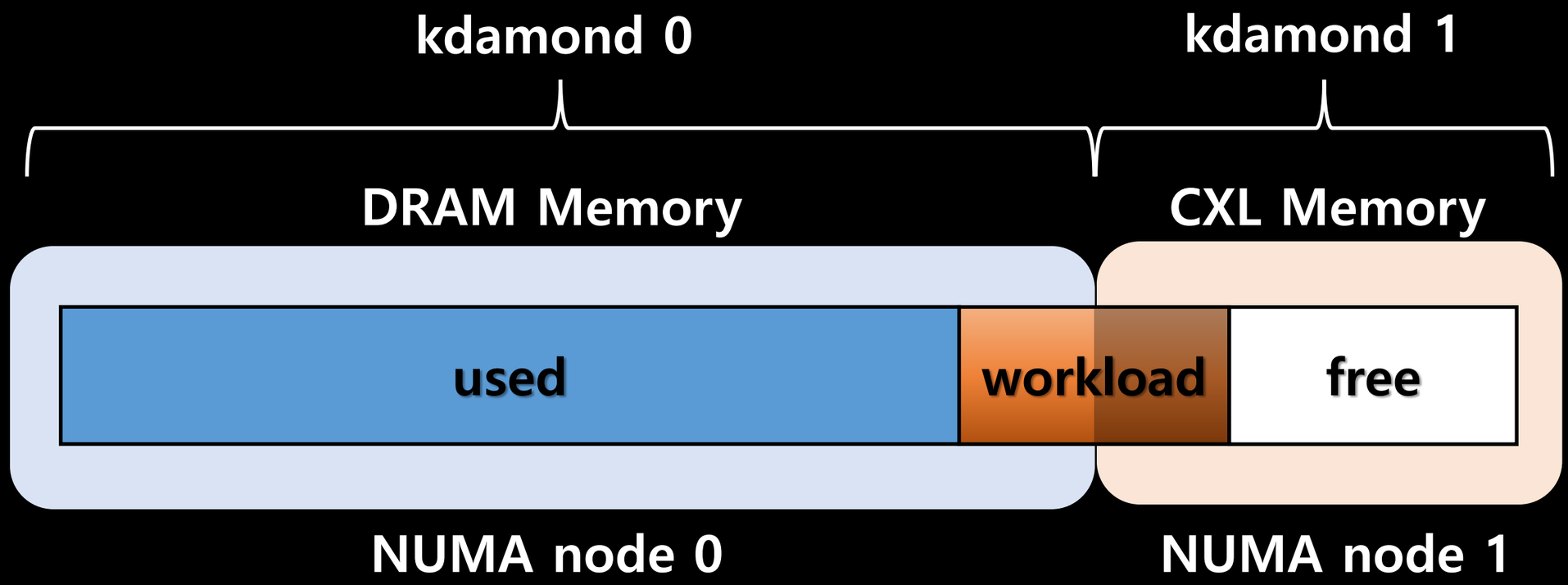
DAMON Usage

- **gen_migpol.py usage**
  - ./tools/gen_migpol.py --demote 0 1 --promote 1 0 **-o hmsdk.yaml**

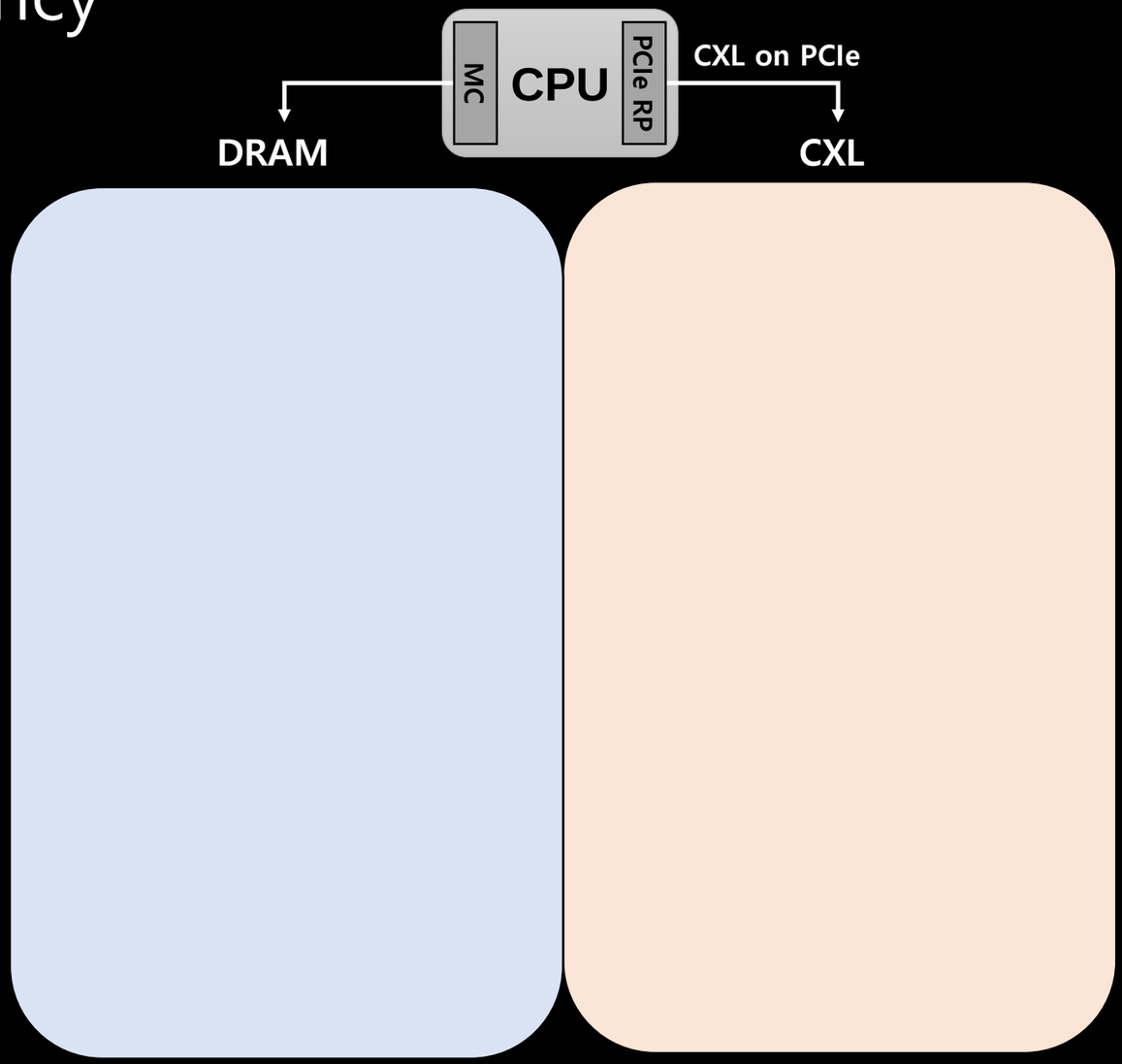


DAMON Usage

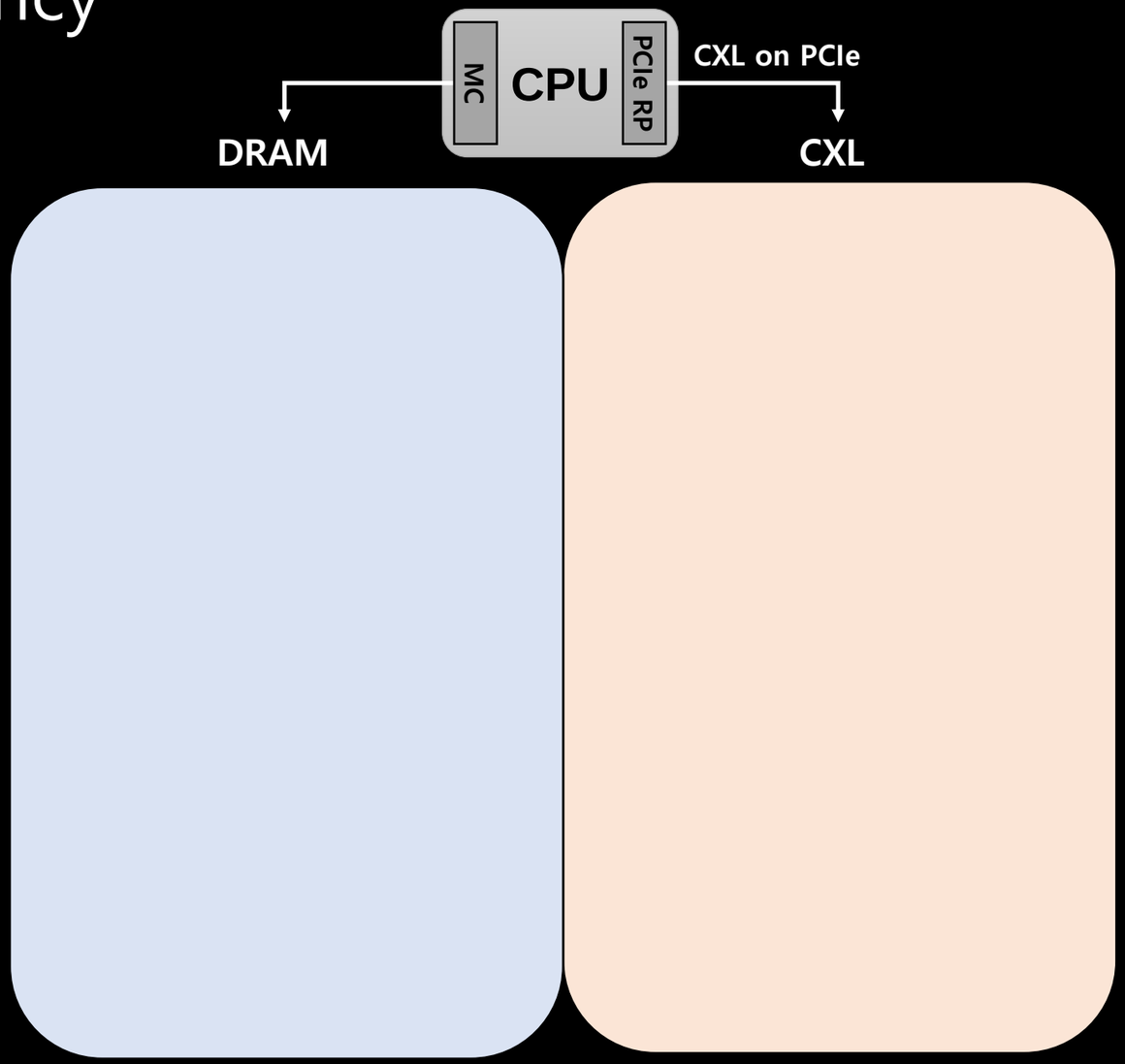
- Activate the DAMON scheme
  - damo start hmsdk.yaml



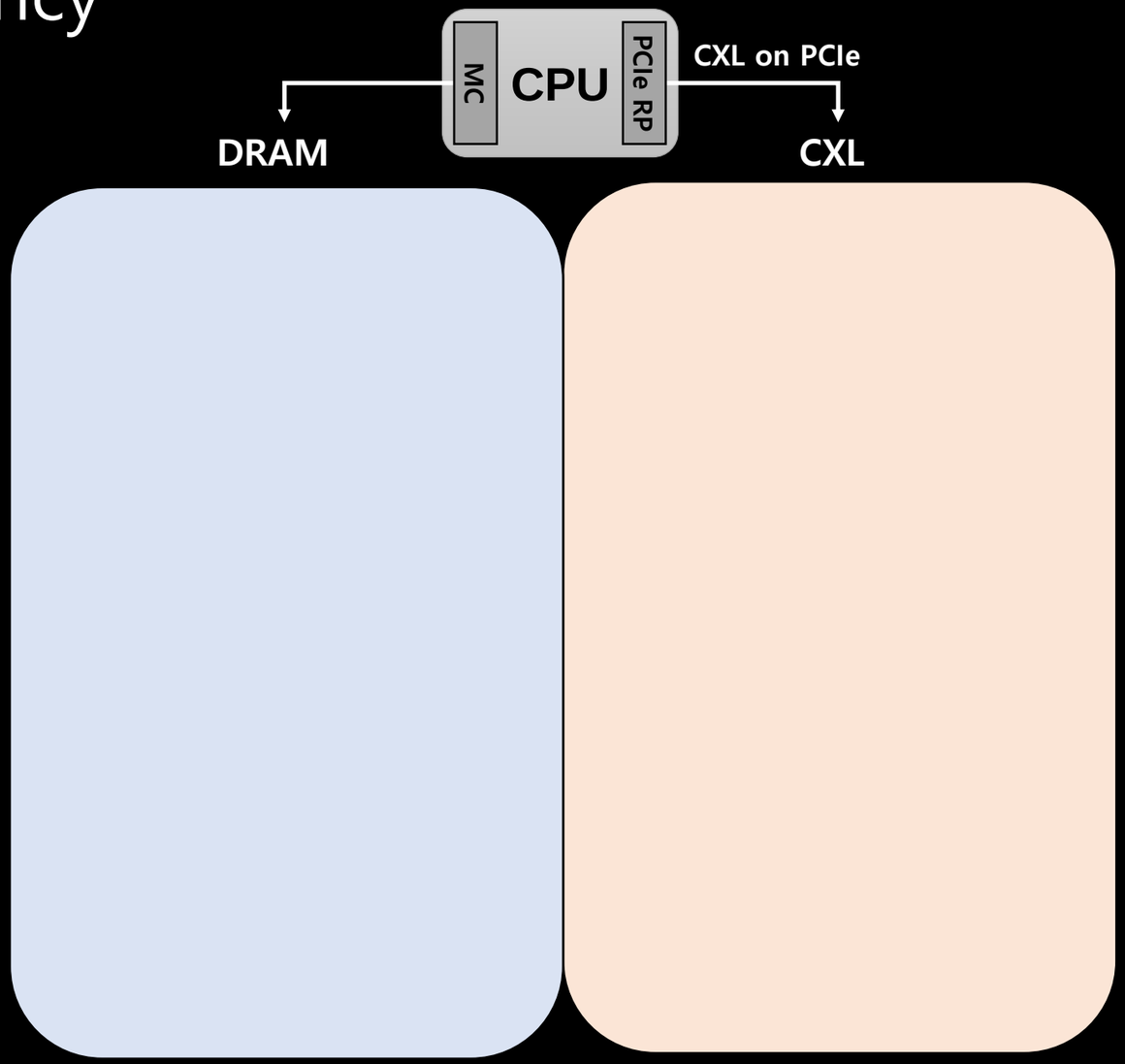
HMSDK: Enhancing CXL Memory Efficiency



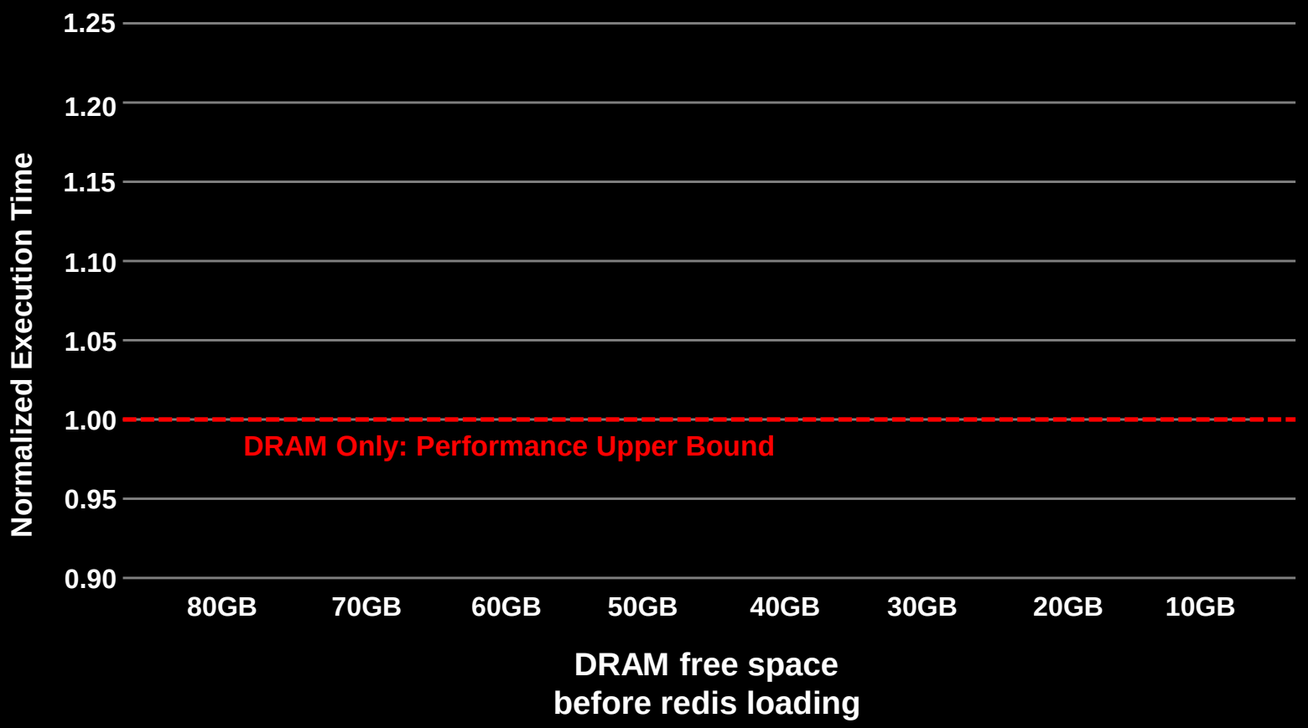
HMSDK: Enhancing CXL Memory Efficiency

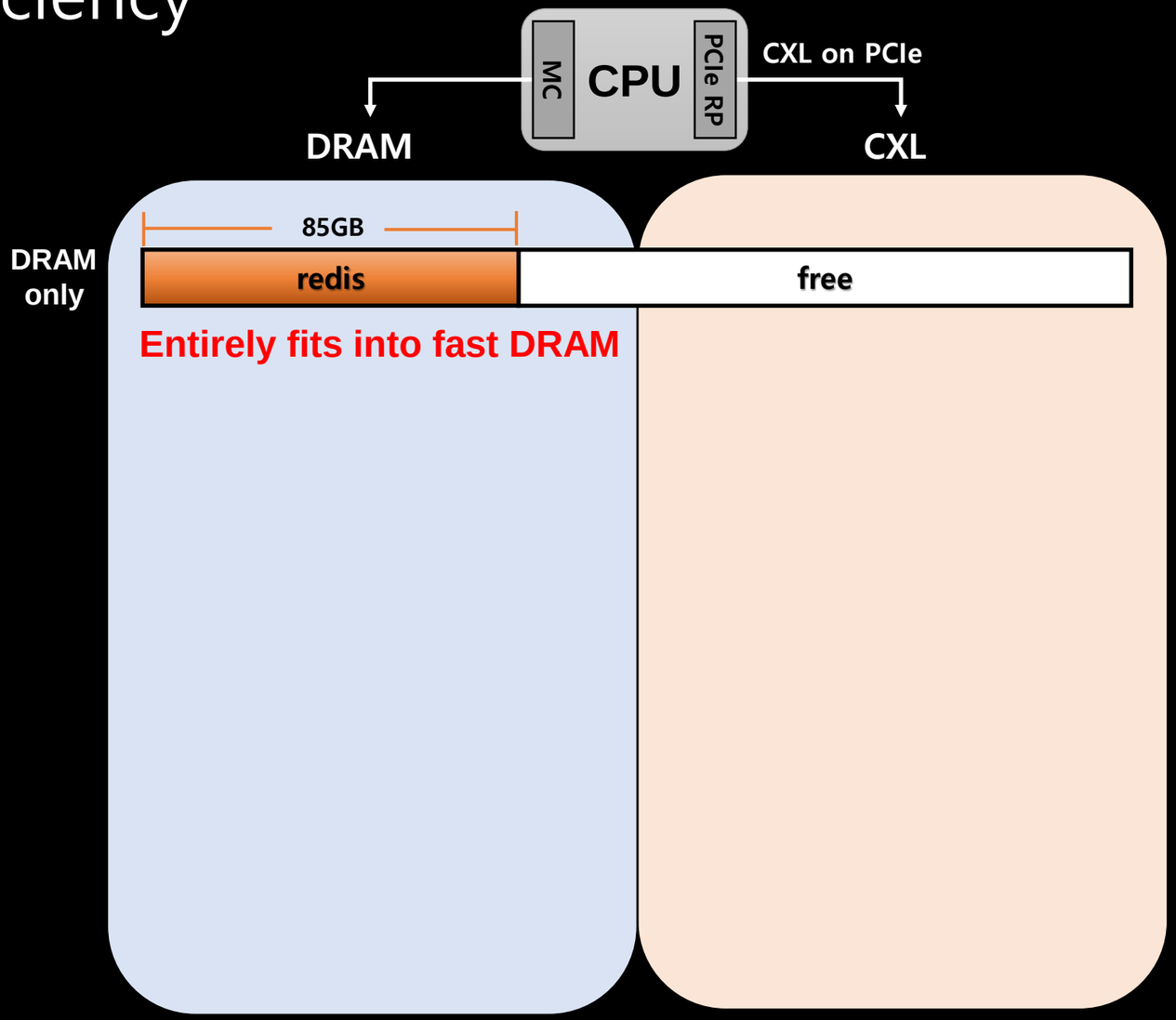


HMSDK: Enhancing CXL Memory Efficiency



HMSDK: Enhancing CXL Memory Efficiency

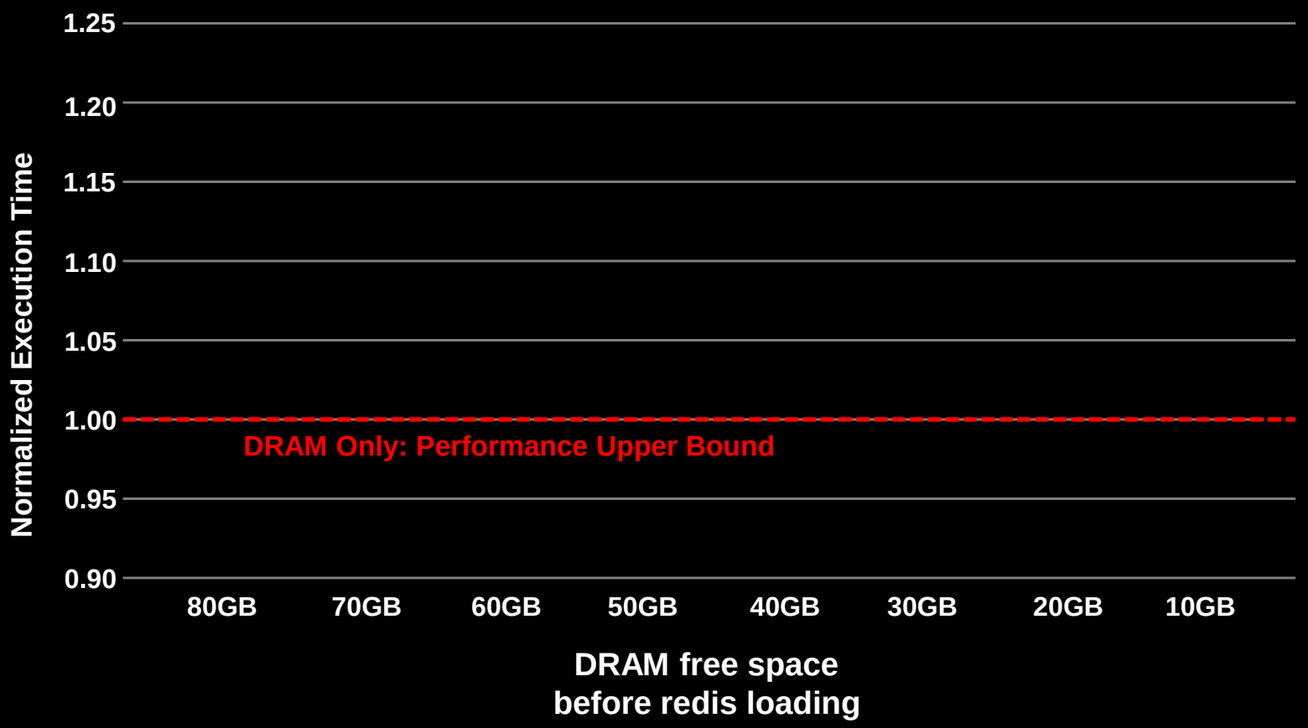


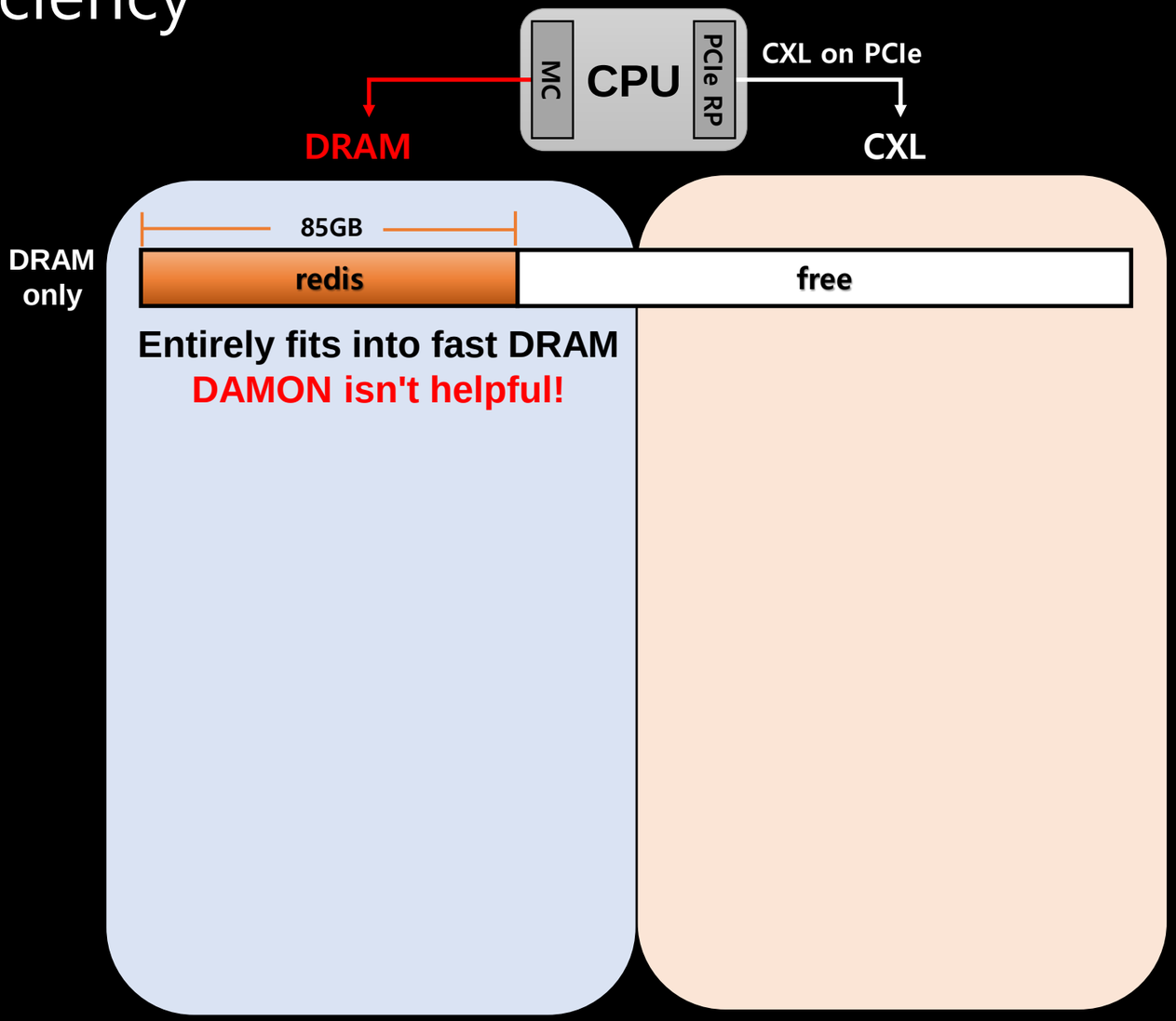


<DRAM only>

1. Redis fully fits into fast DRAM.
2. Set its performance as baseline

HMSDK: Enhancing CXL Memory Efficiency

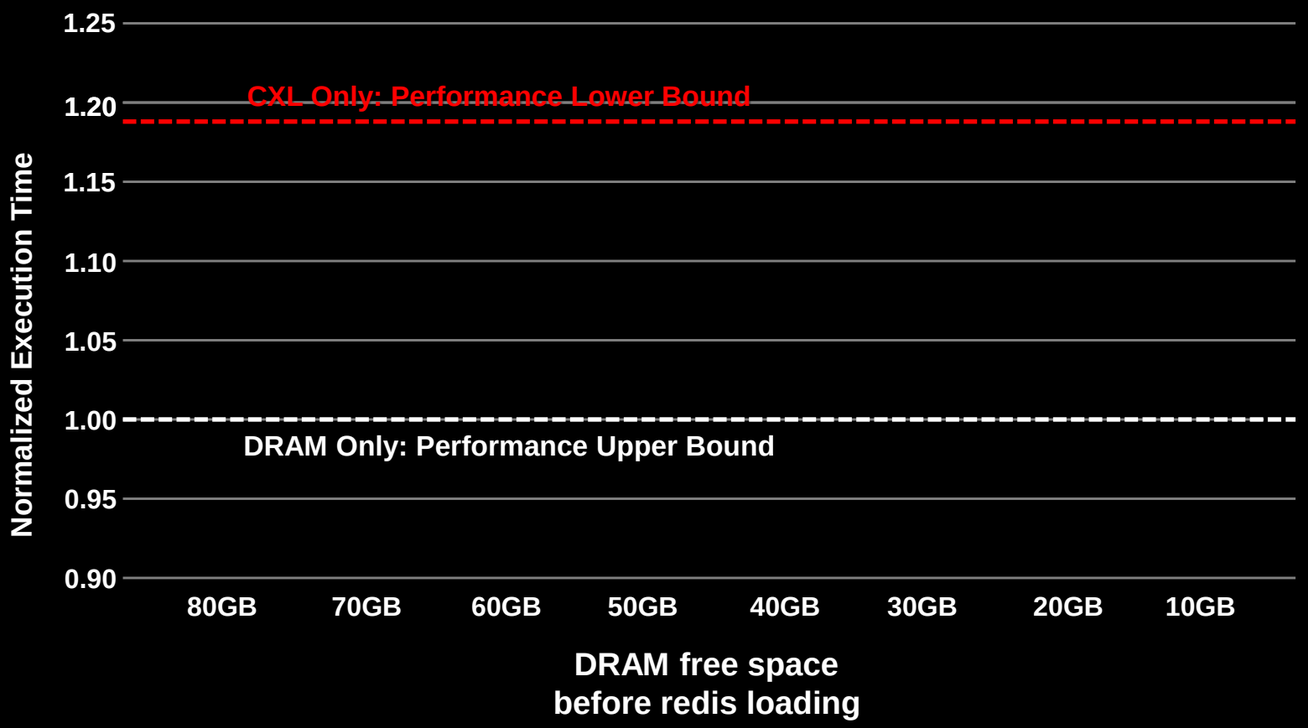


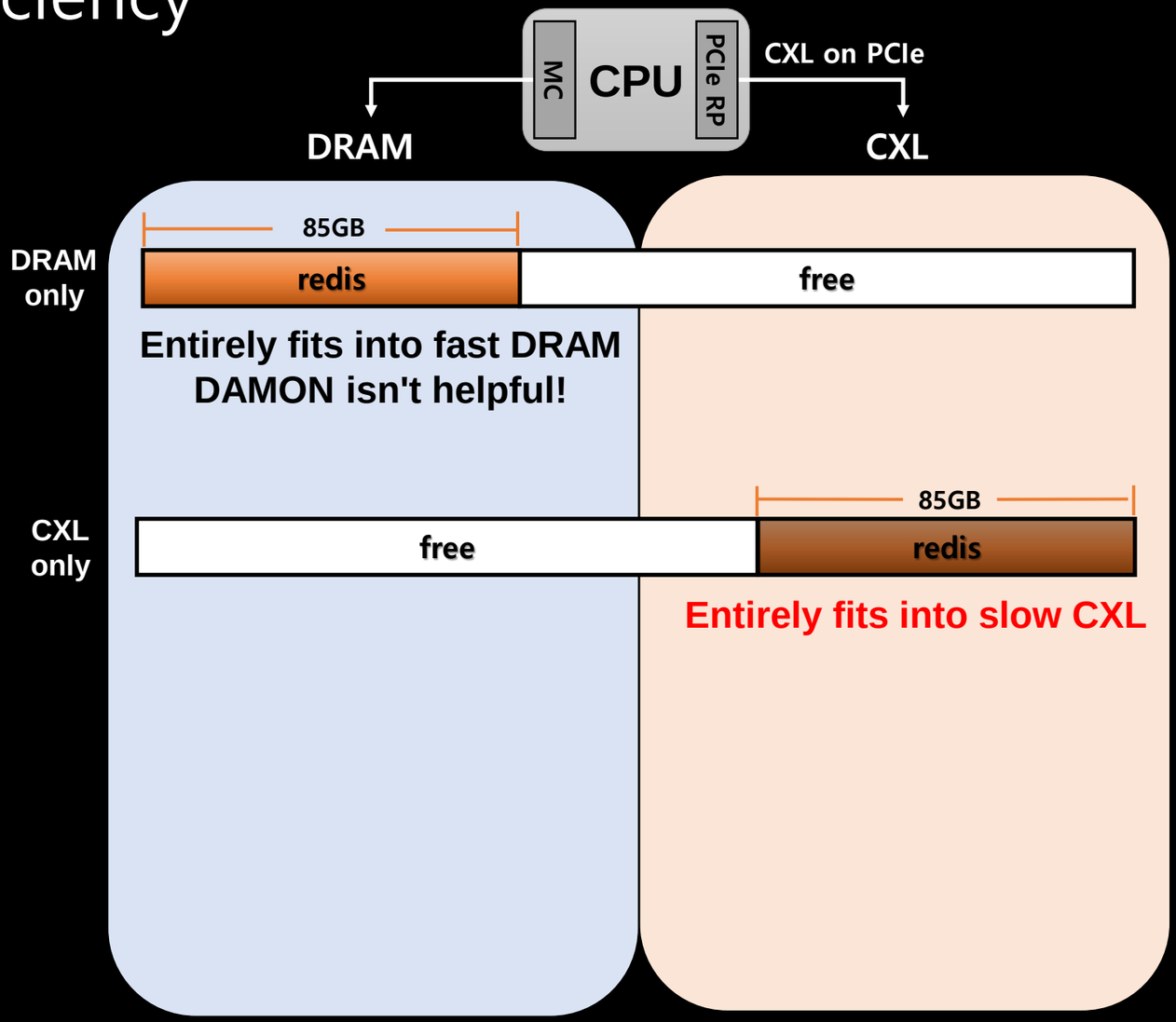


<DRAM only>

1. Redis fully fits into fast DRAM.
2. Set its performance as baseline

HMSDK: Enhancing CXL Memory Efficiency





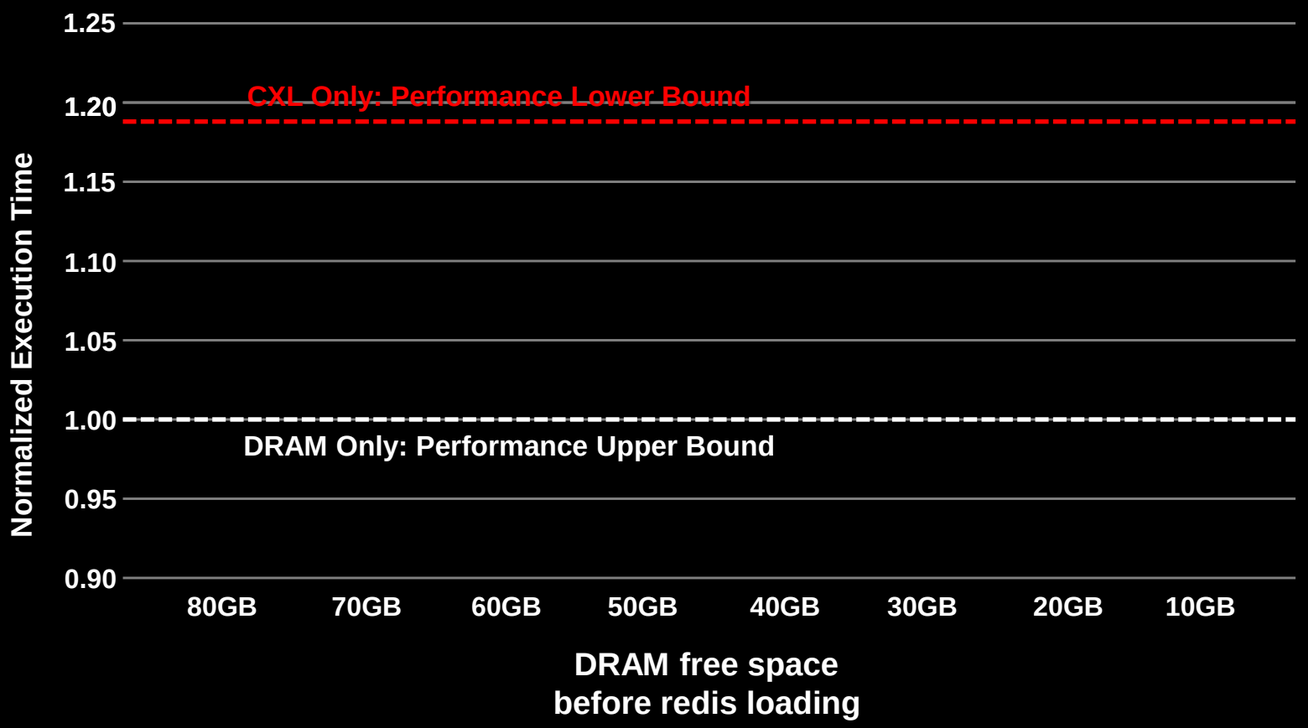
<DRAM only>

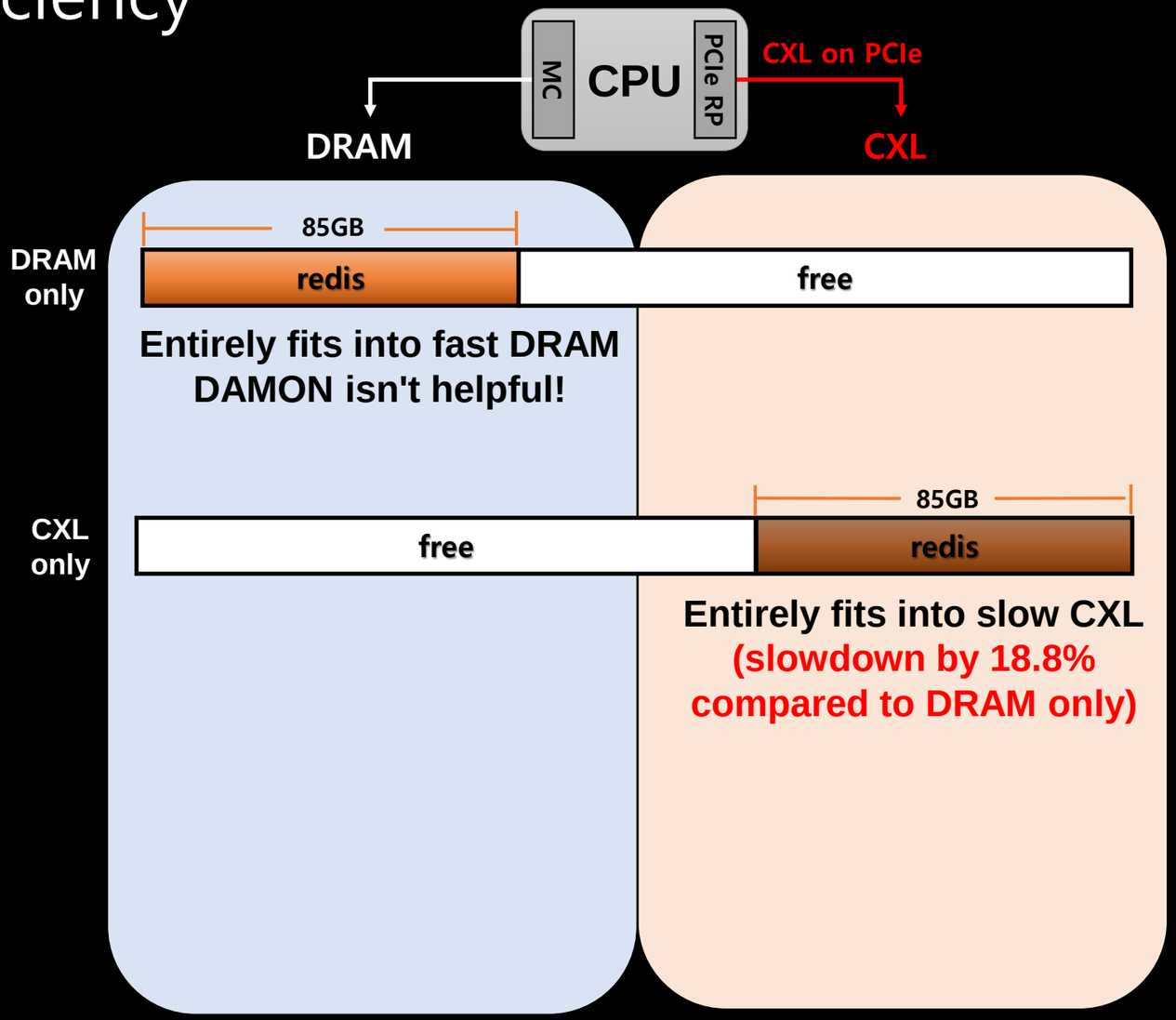
1. Redis fully fits into fast DRAM.
2. Set its performance as baseline

<CXL only>

1. Redis fully fits into slow CXL memory.
2. Set its performance as lower bound

HMSDK: Enhancing CXL Memory Efficiency

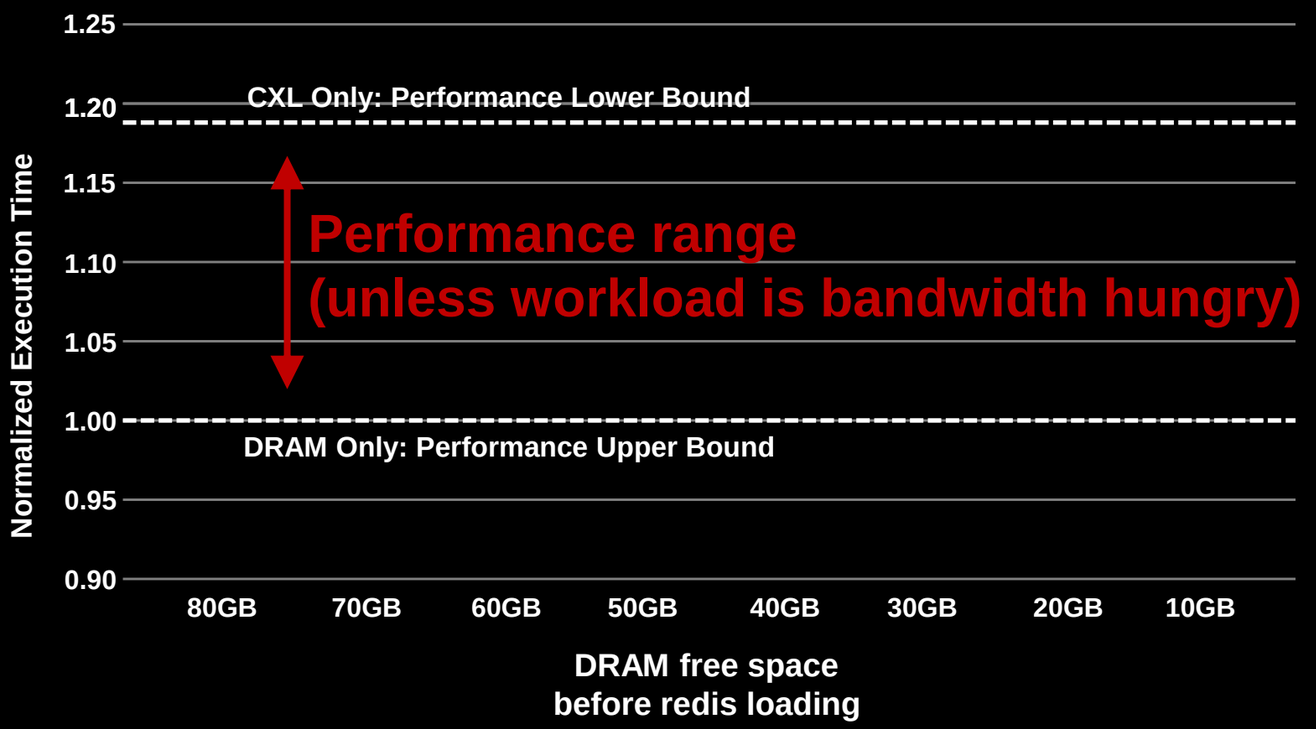


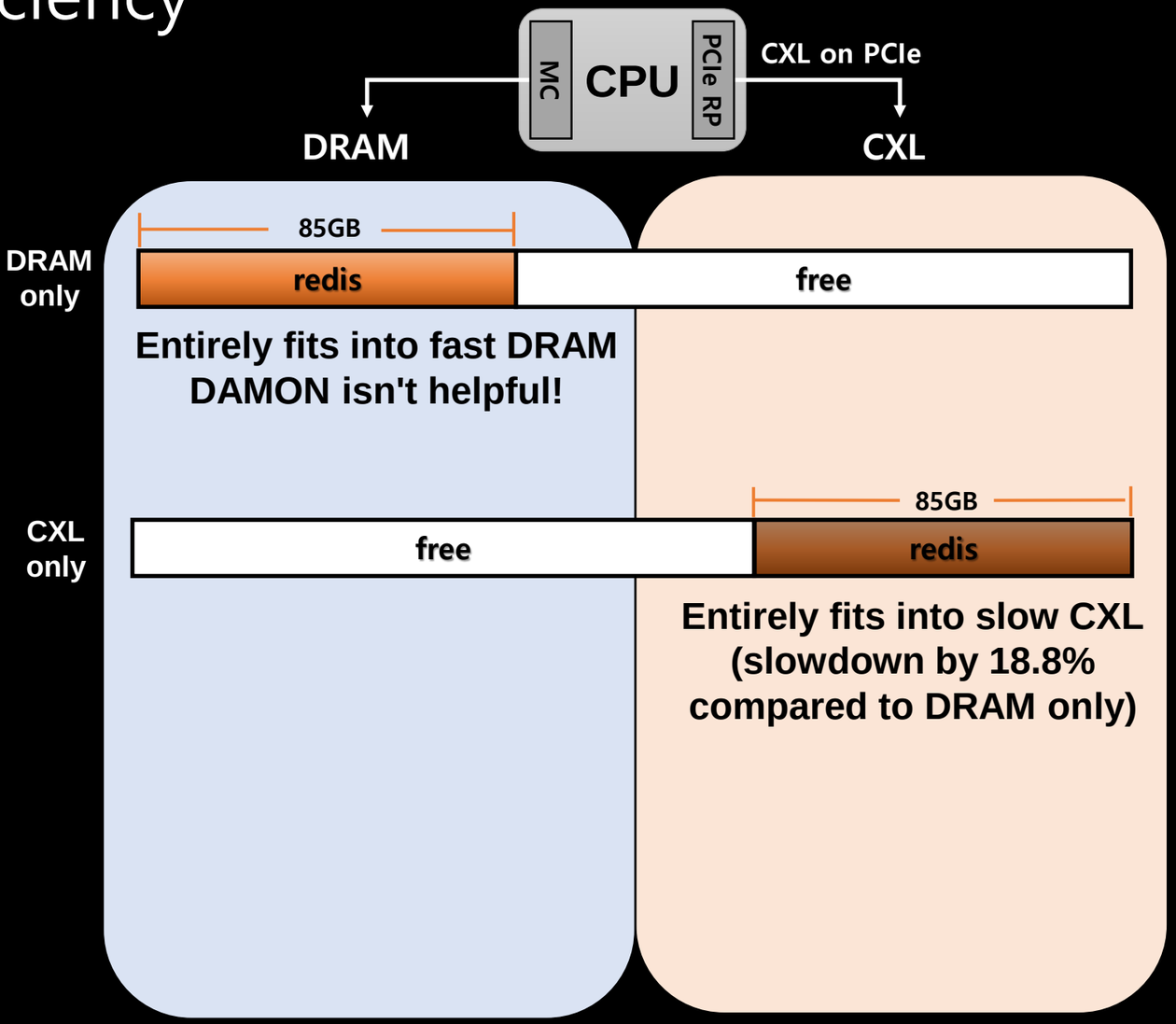


<DRAM only>
1. Redis fully fits into fast DRAM.
2. Set its performance as baseline

<CXL only>
1. Redis fully fits into slow CXL memory.
2. Set its performance as lower bound

HMSDK: Enhancing CXL Memory Efficiency

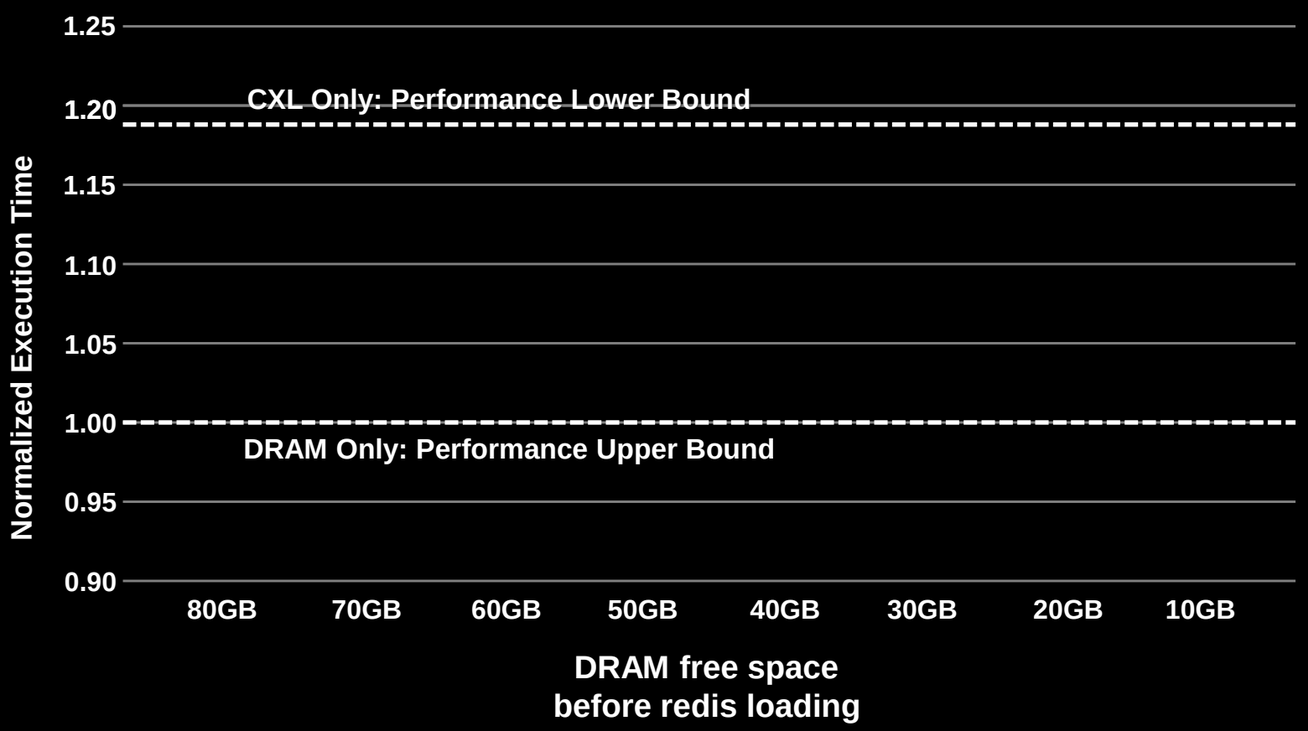


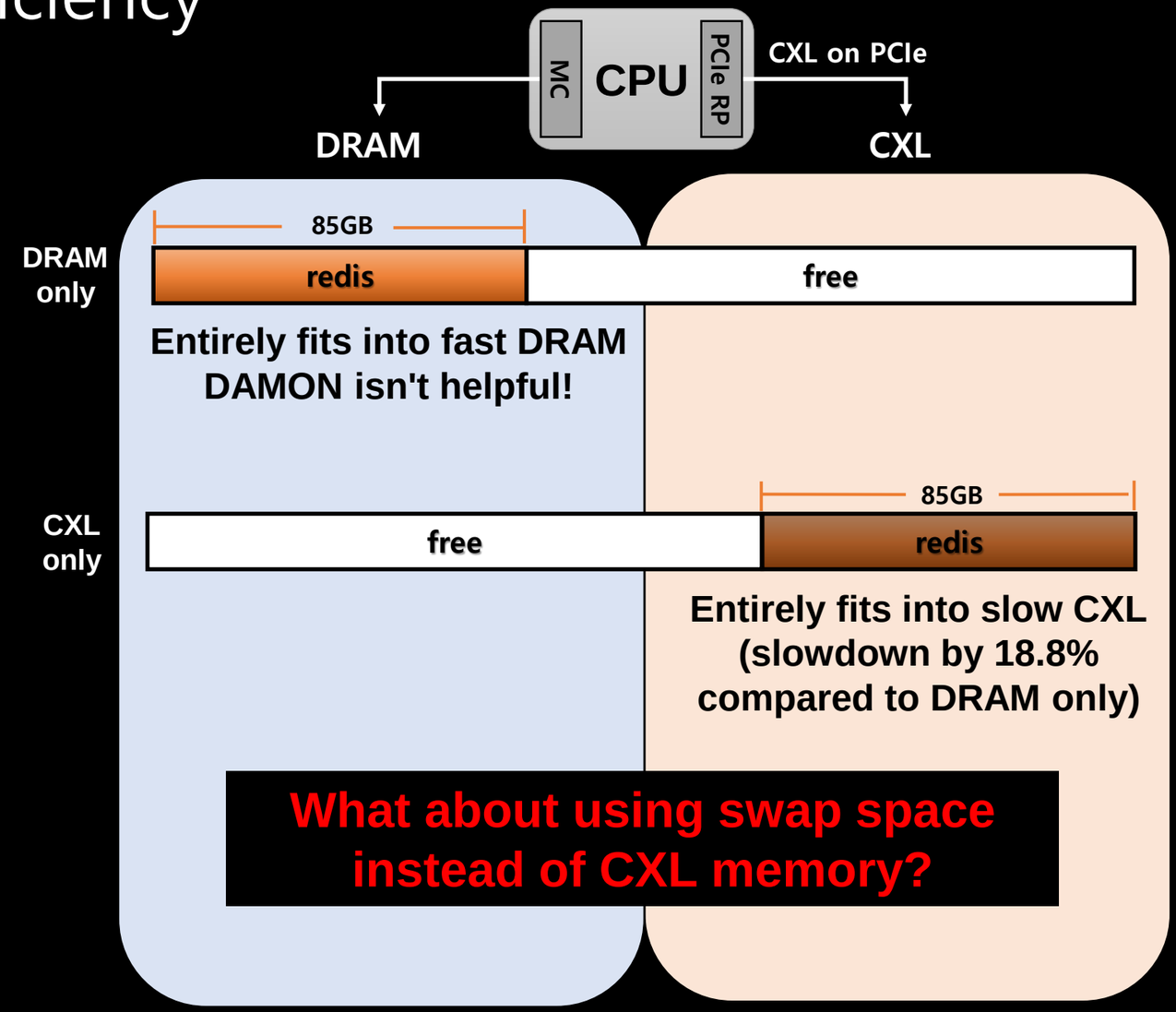


<DRAM only>

1. Redis fully fits into fast DRAM.
2. Set its performance as baseline

HMSDK: Enhancing CXL Memory Efficiency

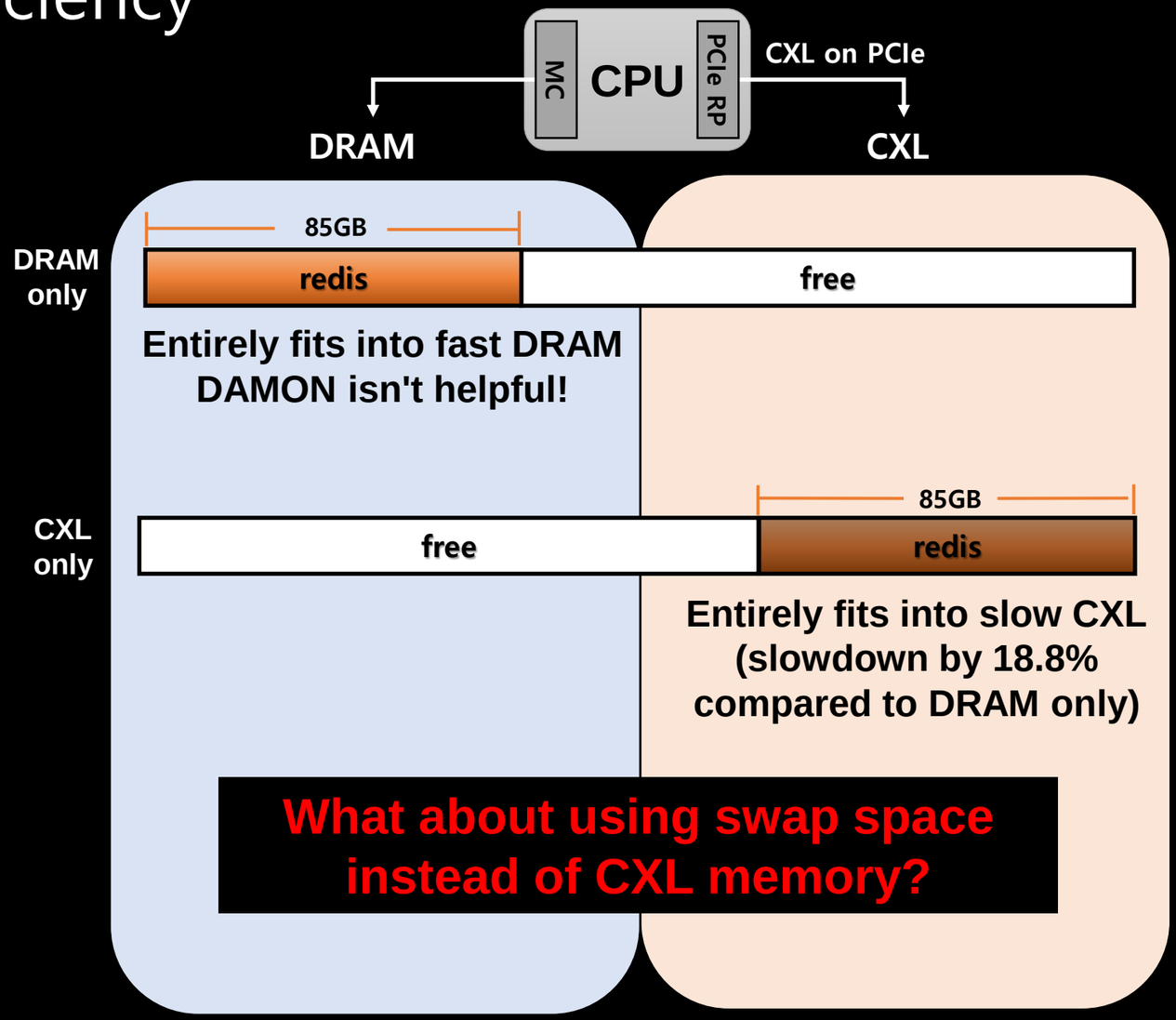




<DRAM only>

1. Redis fully fits into fast DRAM.
2. Set its performance as baseline

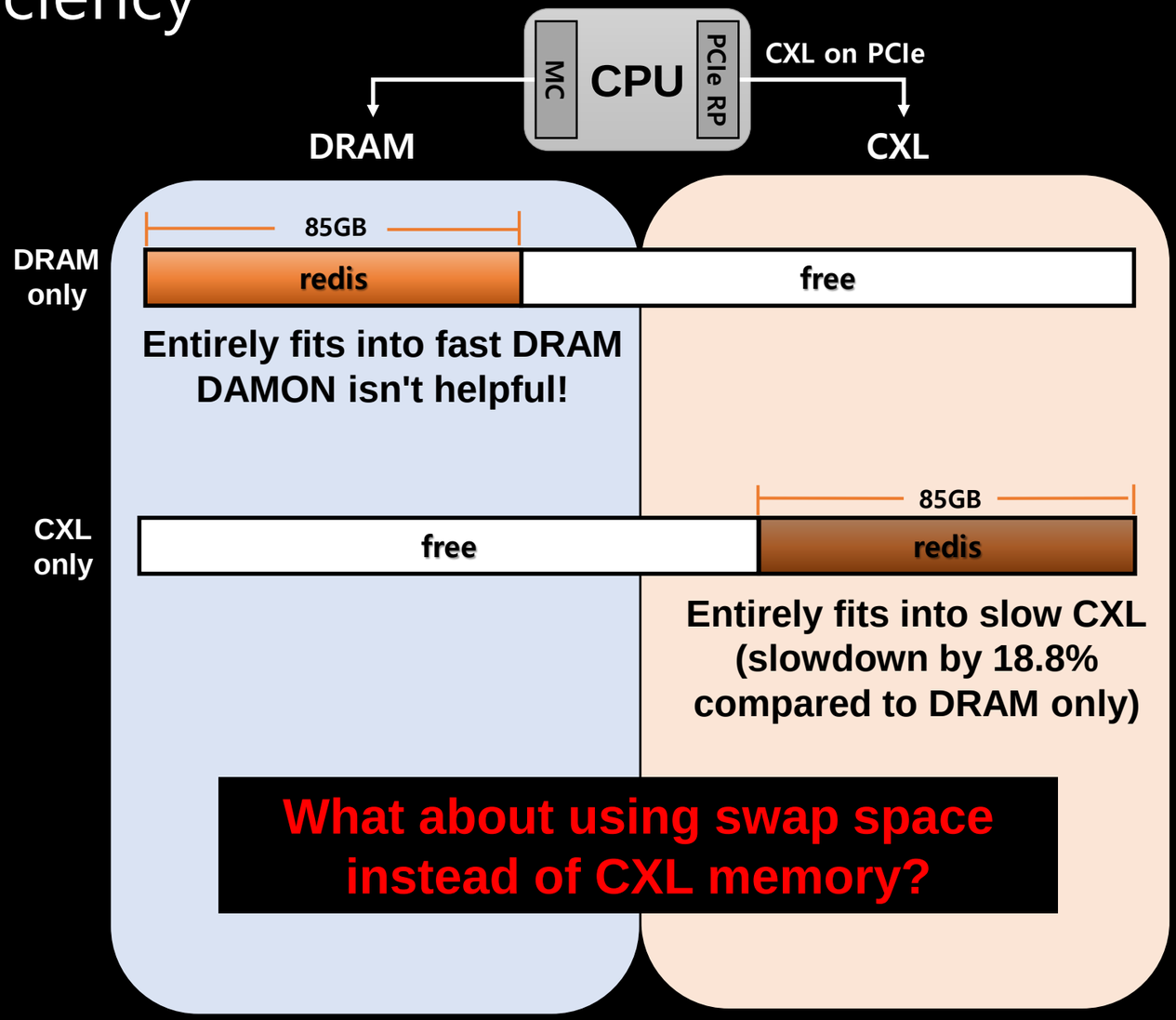
HMSDK: Enhancing CXL Memory Efficiency



<DRAM only>
1. Redis fully fits into fast DRAM.
2. Set its performance as baseline

<Using swap space>
1. Pushing redis to swap device instead of CXL.

HMSDK: Enhancing CXL Memory Efficiency



<DRAM only>

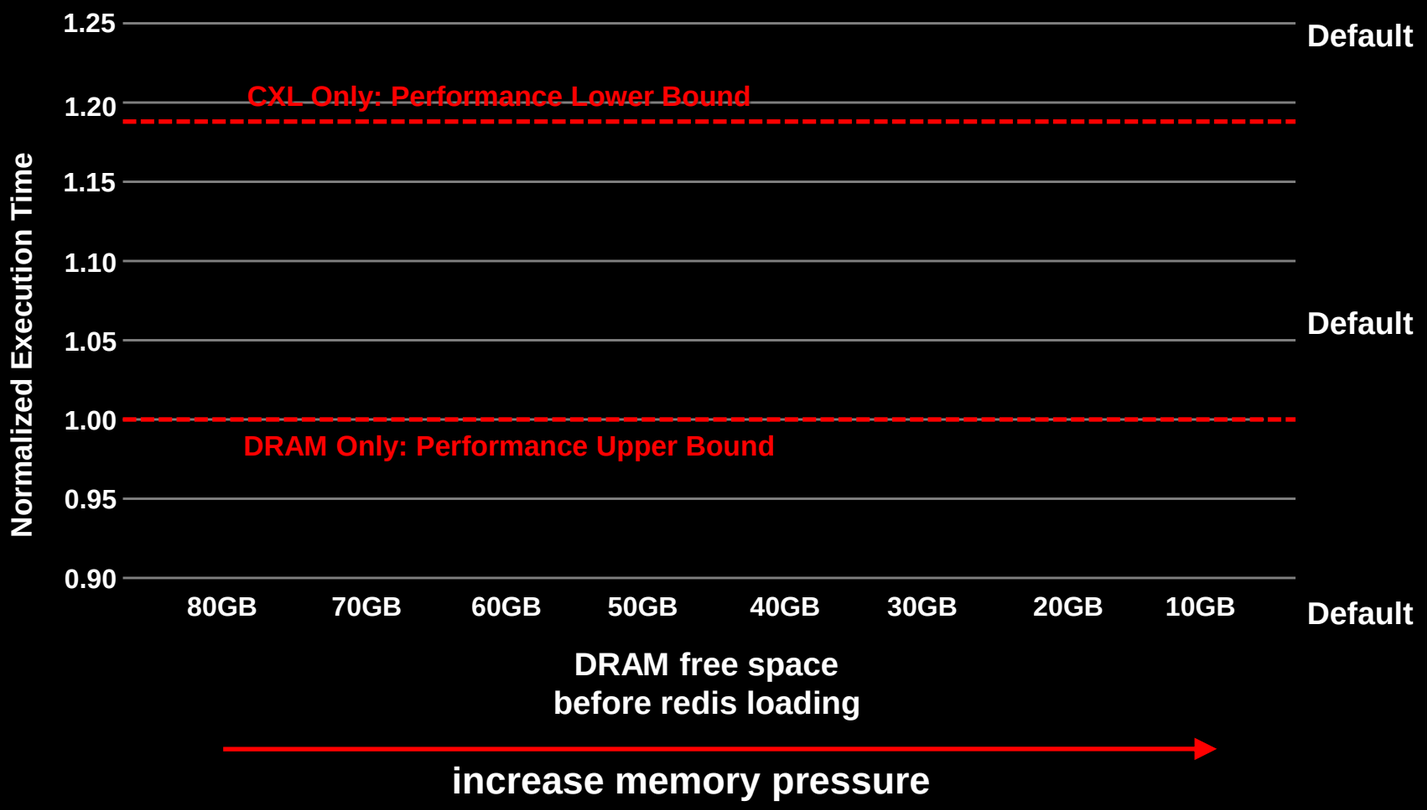
1. Redis fully fits into fast DRAM.
2. Set its performance as baseline

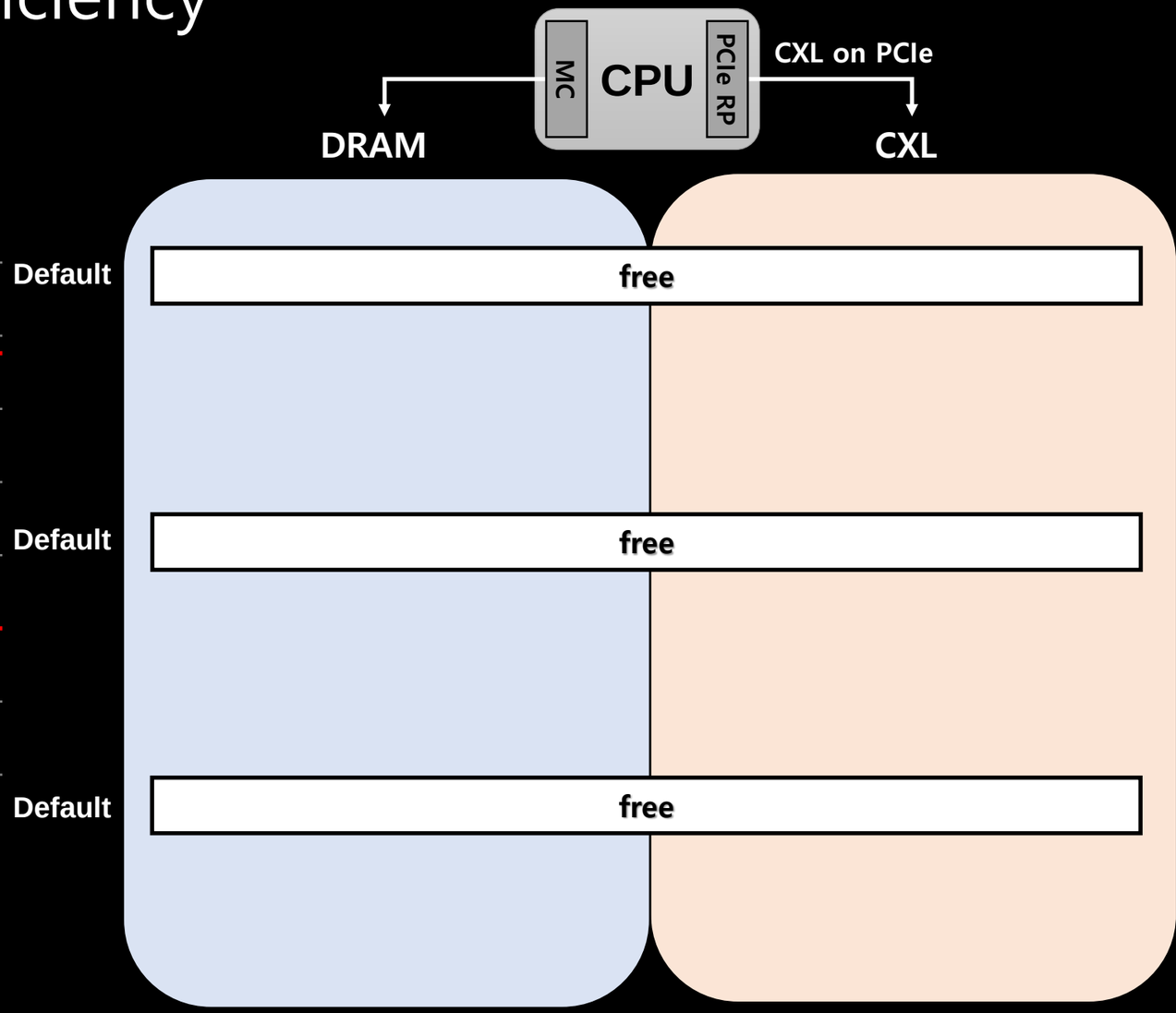
<Using swap space>

1. Pushing redis to swap device instead of CXL.
2. Not the scope of this evaluation!
   (**obviously much slower than CXL only**)

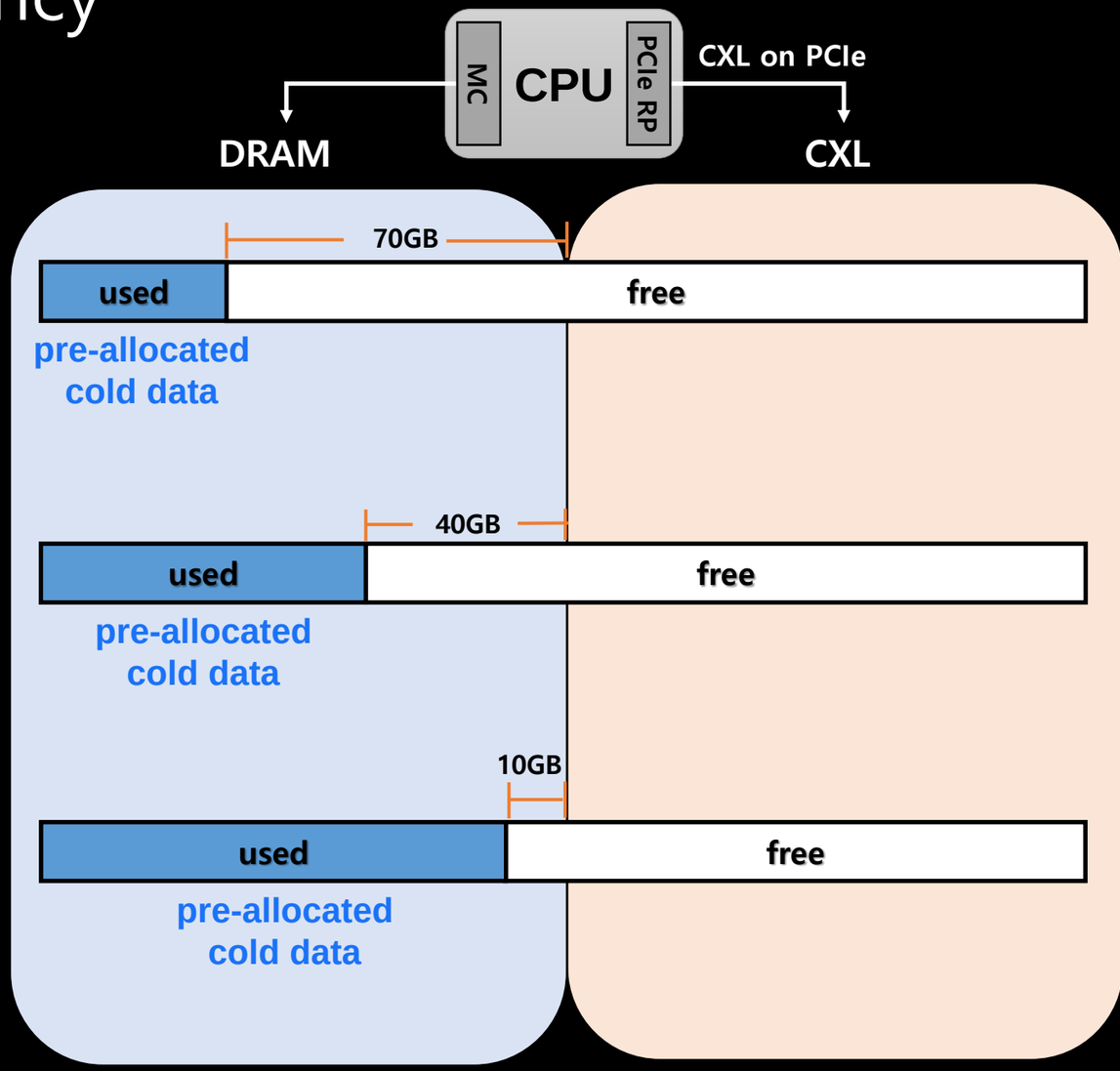
Evaluation based on various memory pressured cases

HMSDK: Enhancing CXL Memory Efficiency

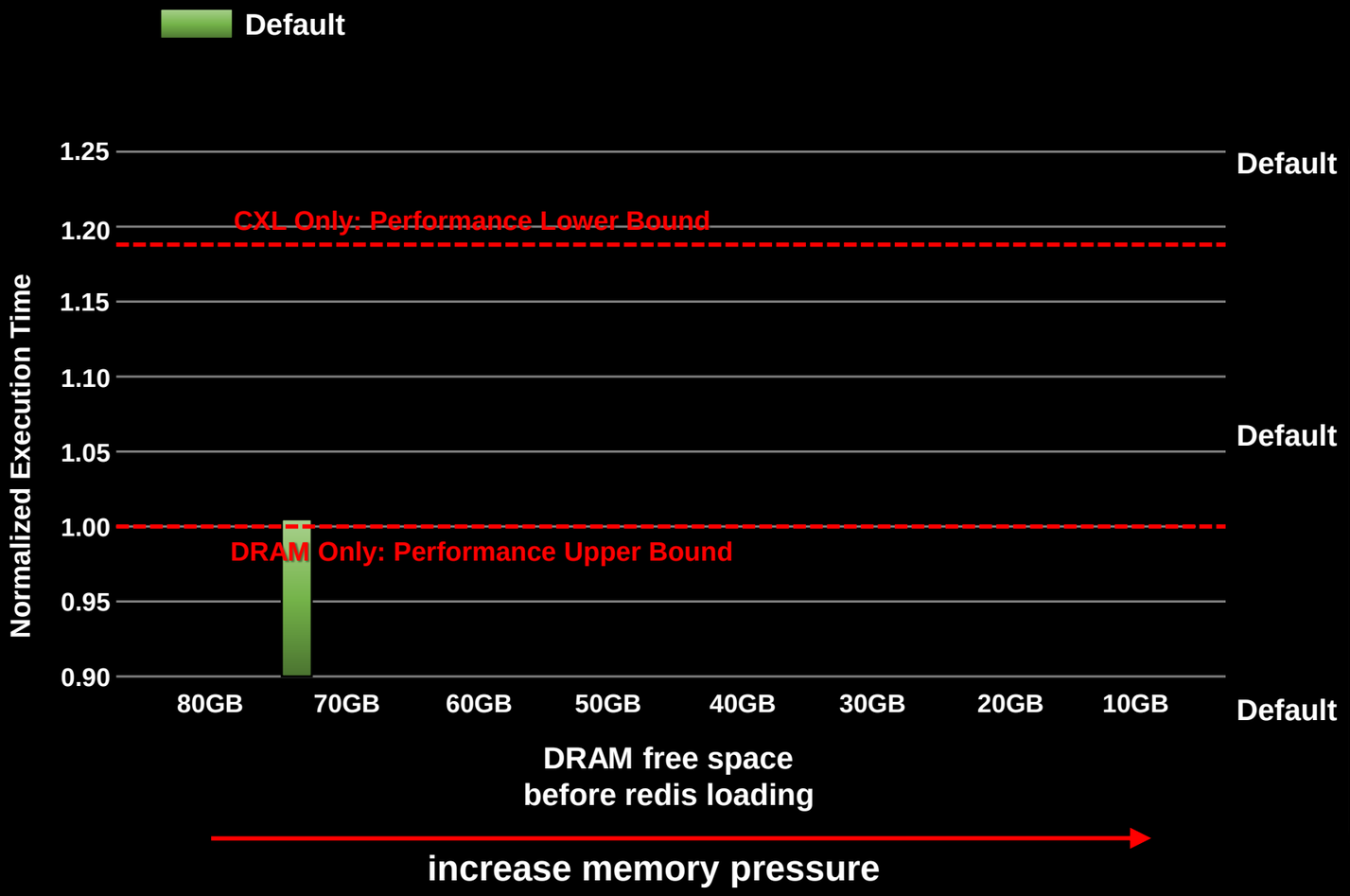


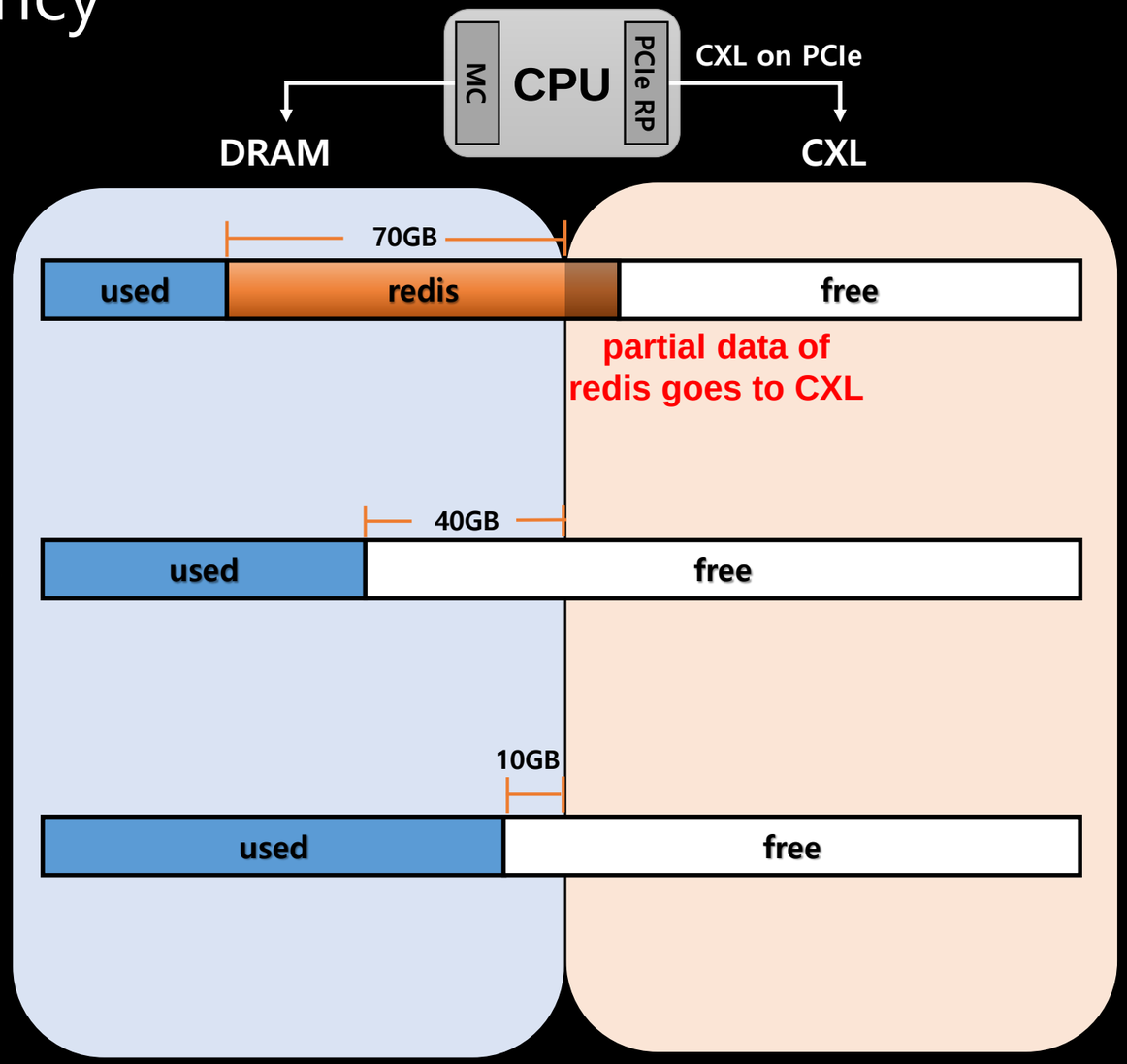


HMSDK: Enhancing CXL Memory Efficiency



HMSDK: Enhancing CXL Memory Efficiency

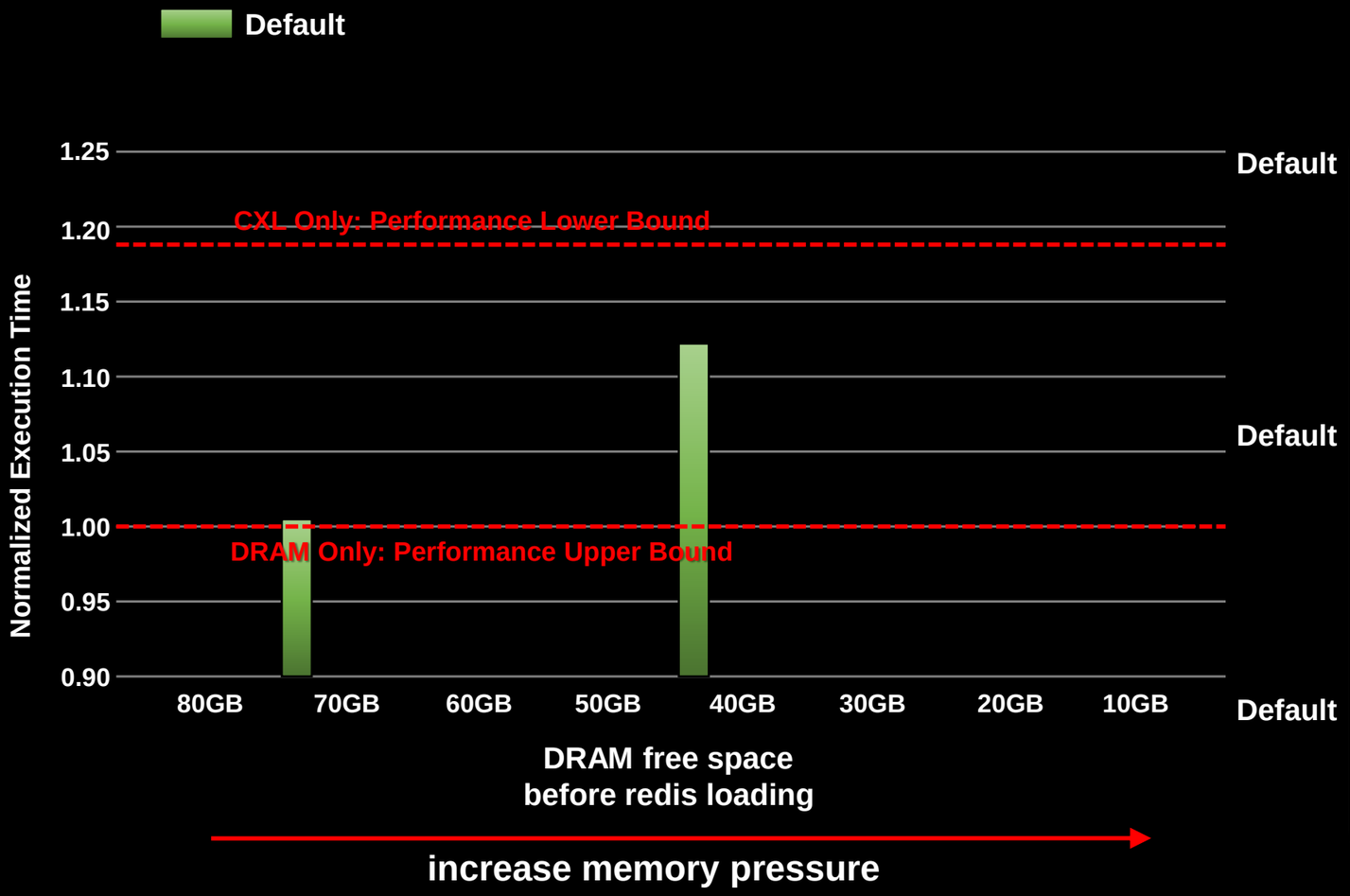


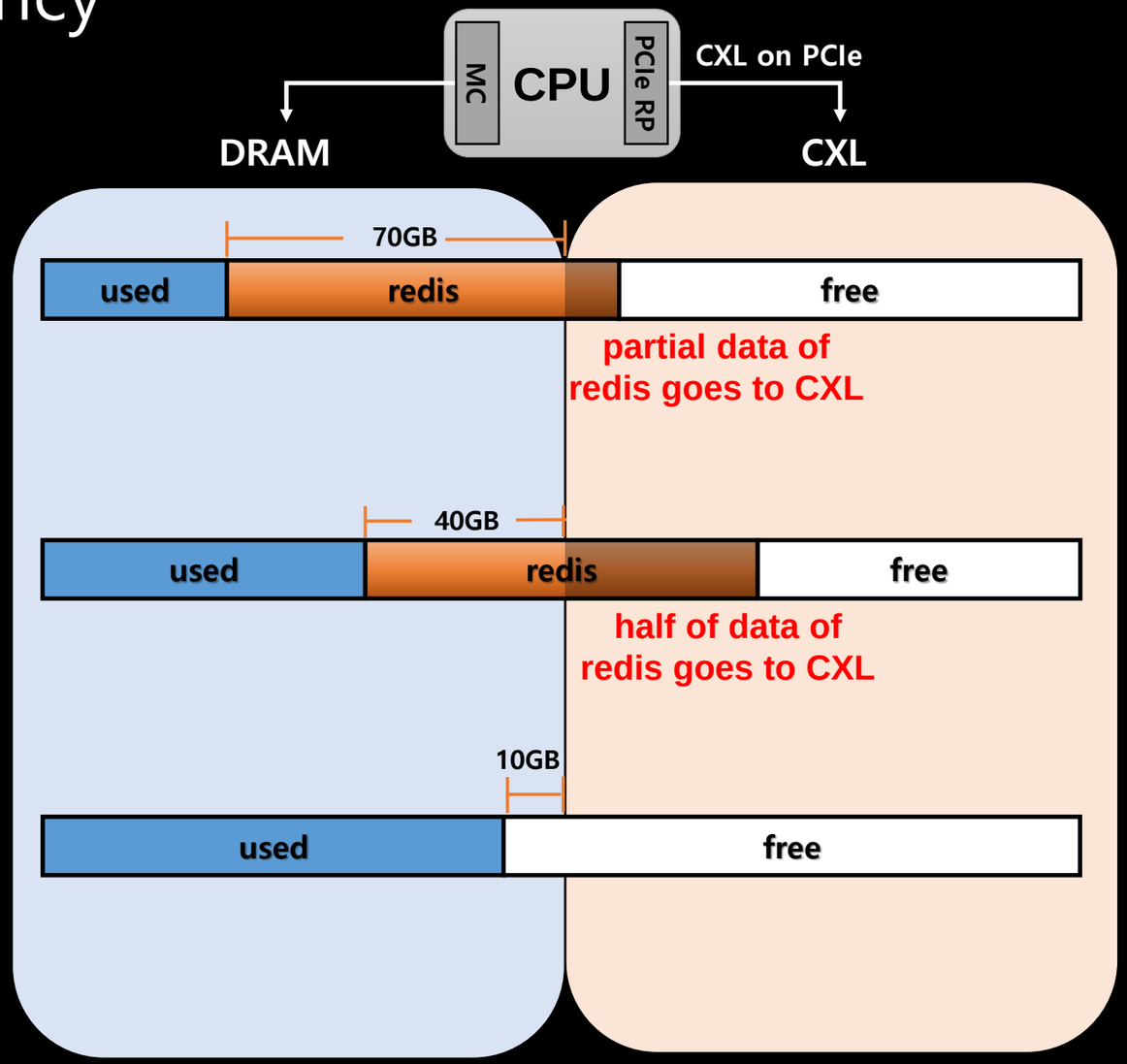


<Default>

1. DRAM is partially used by non-redis cold data.
2. Partial redis data is allocated on CXL memory.
   (due to insufficient space on DRAM)

HMSDK: Enhancing CXL Memory Efficiency

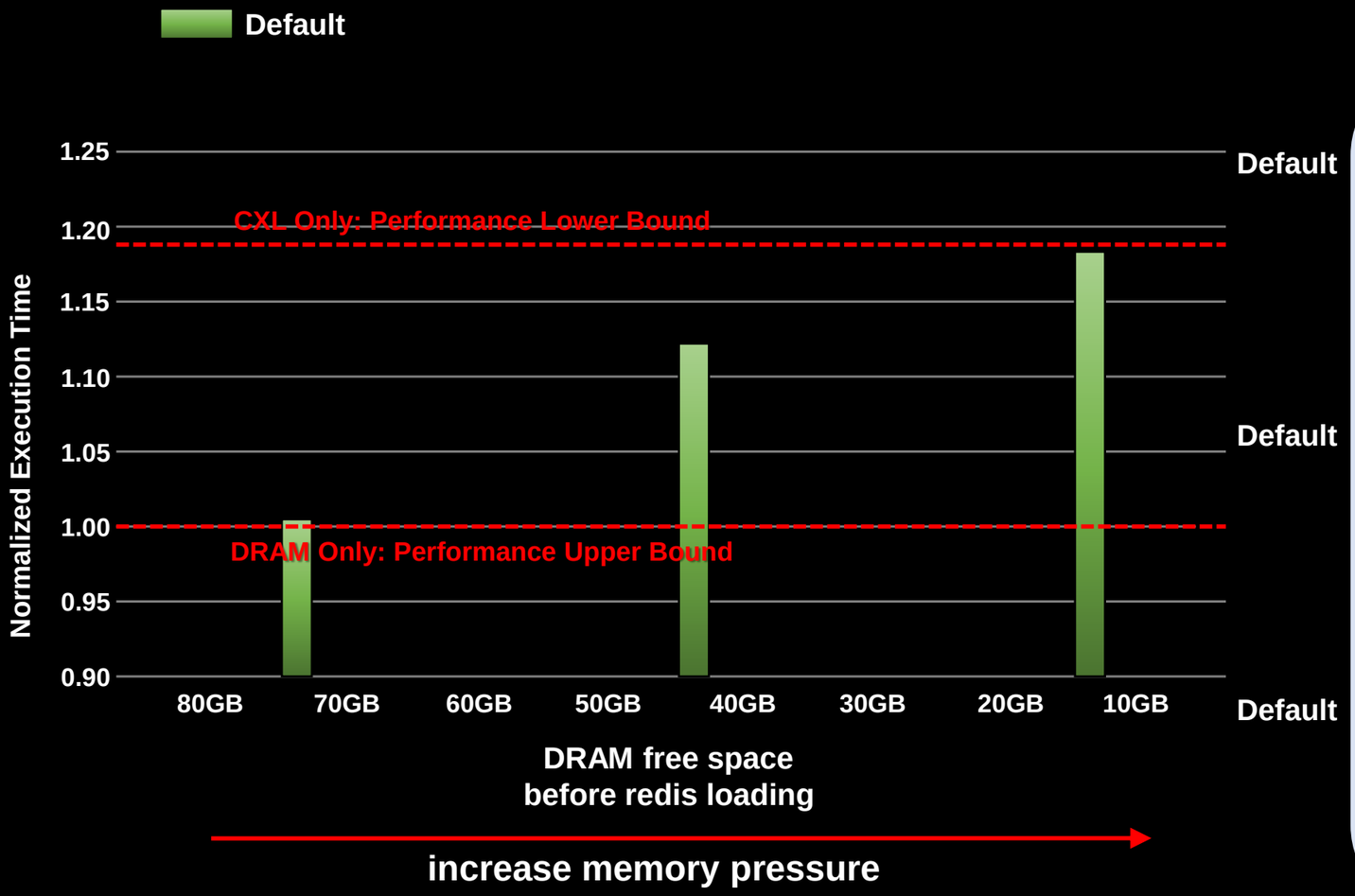


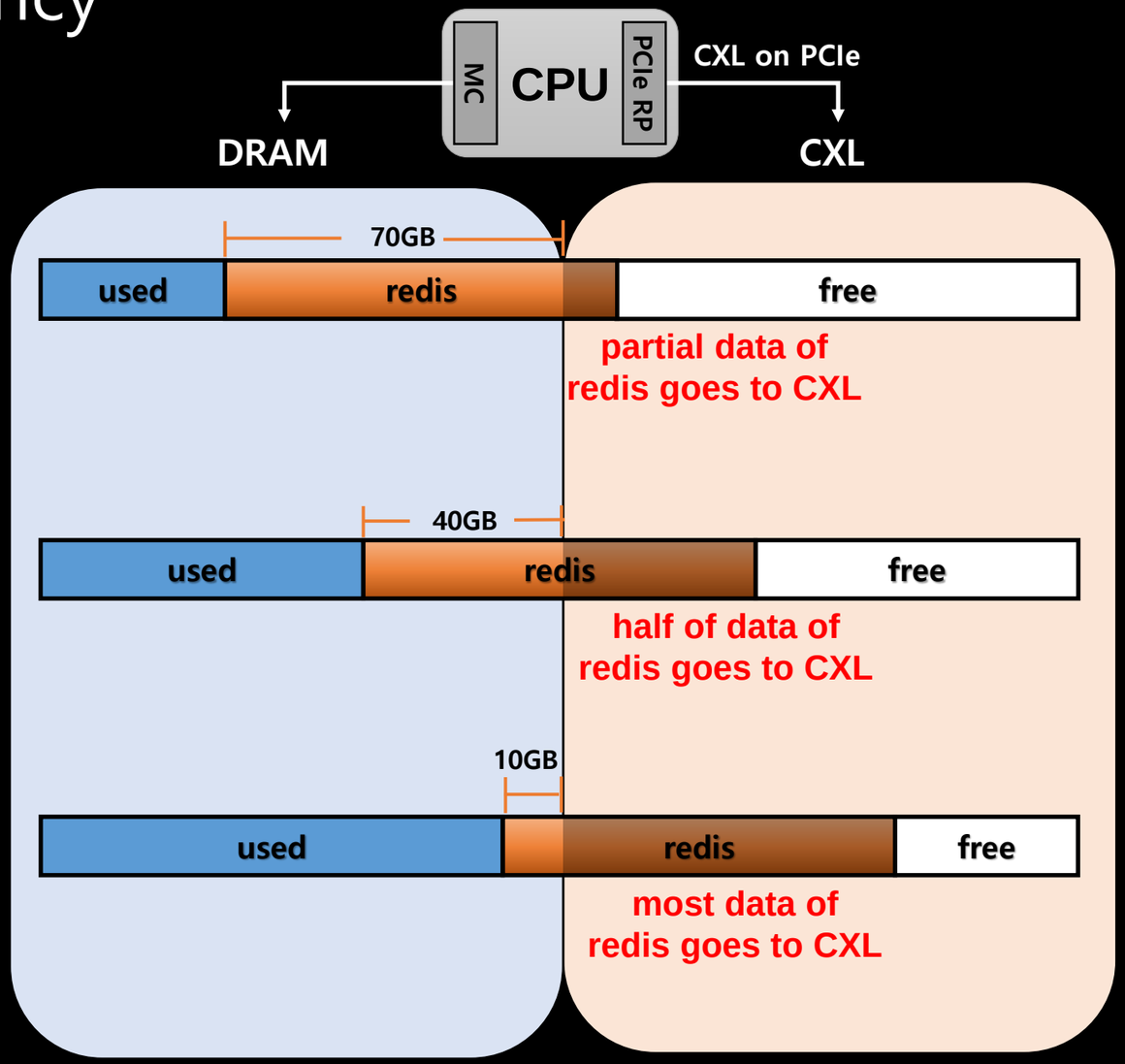


<Default>

1. DRAM is partially used by non-redis cold data.
2. Partial redis data is allocated on CXL memory.
   (due to insufficient space on DRAM)

HMSDK: Enhancing CXL Memory Efficiency

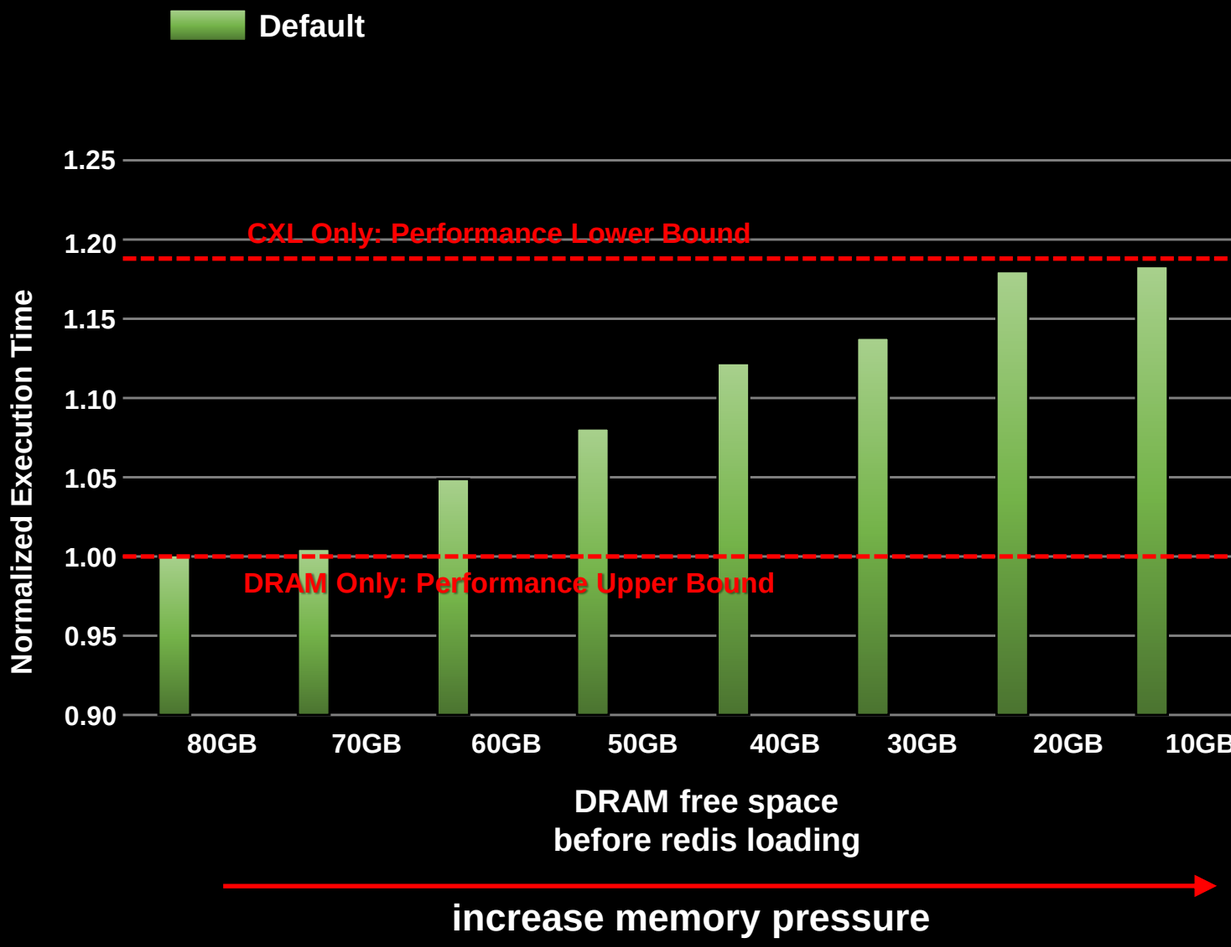


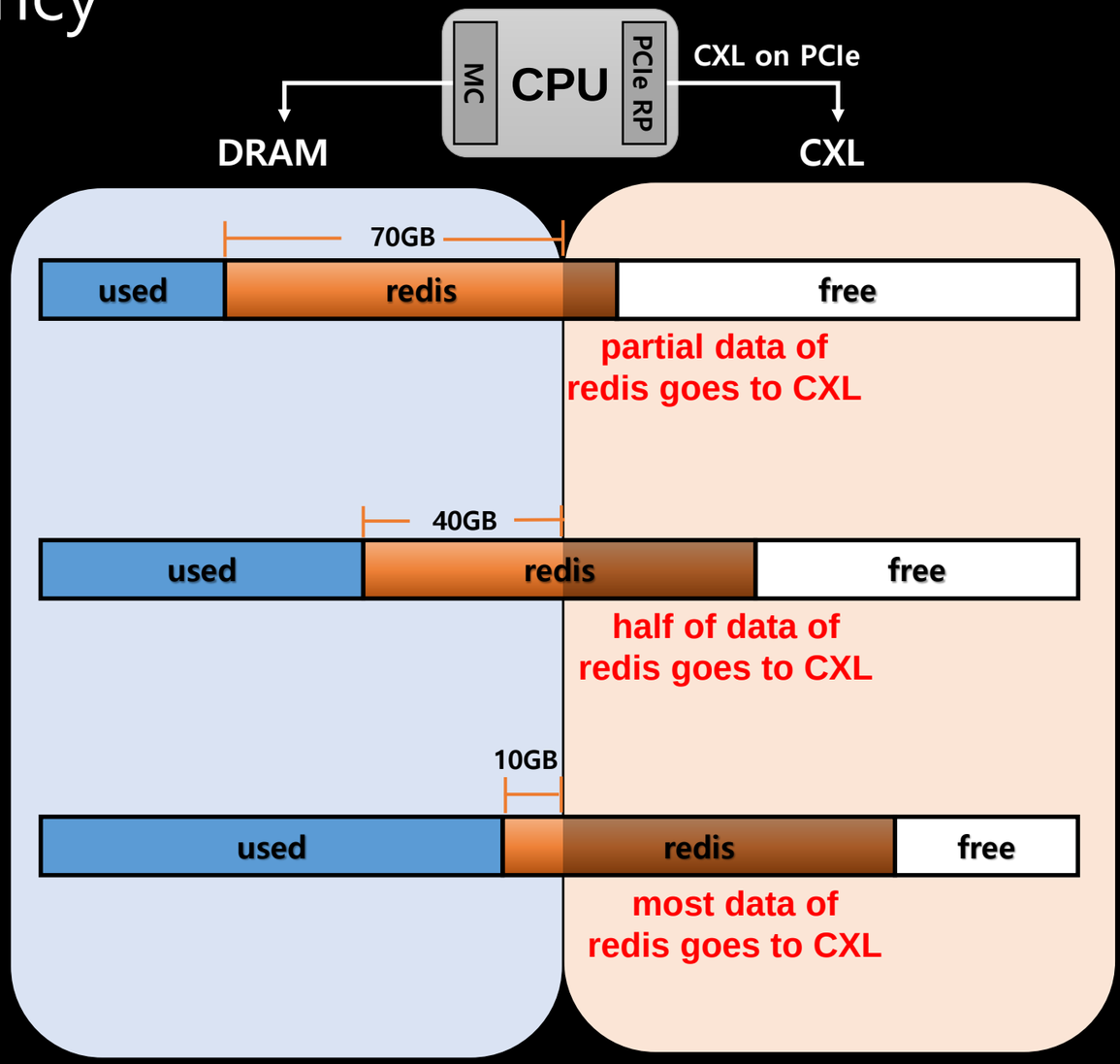


<Default>

1. DRAM is partially used by non-redis cold data.
2. Partial redis data is allocated on CXL memory.
   (due to insufficient space on DRAM)

HMSDK: Enhancing CXL Memory Efficiency

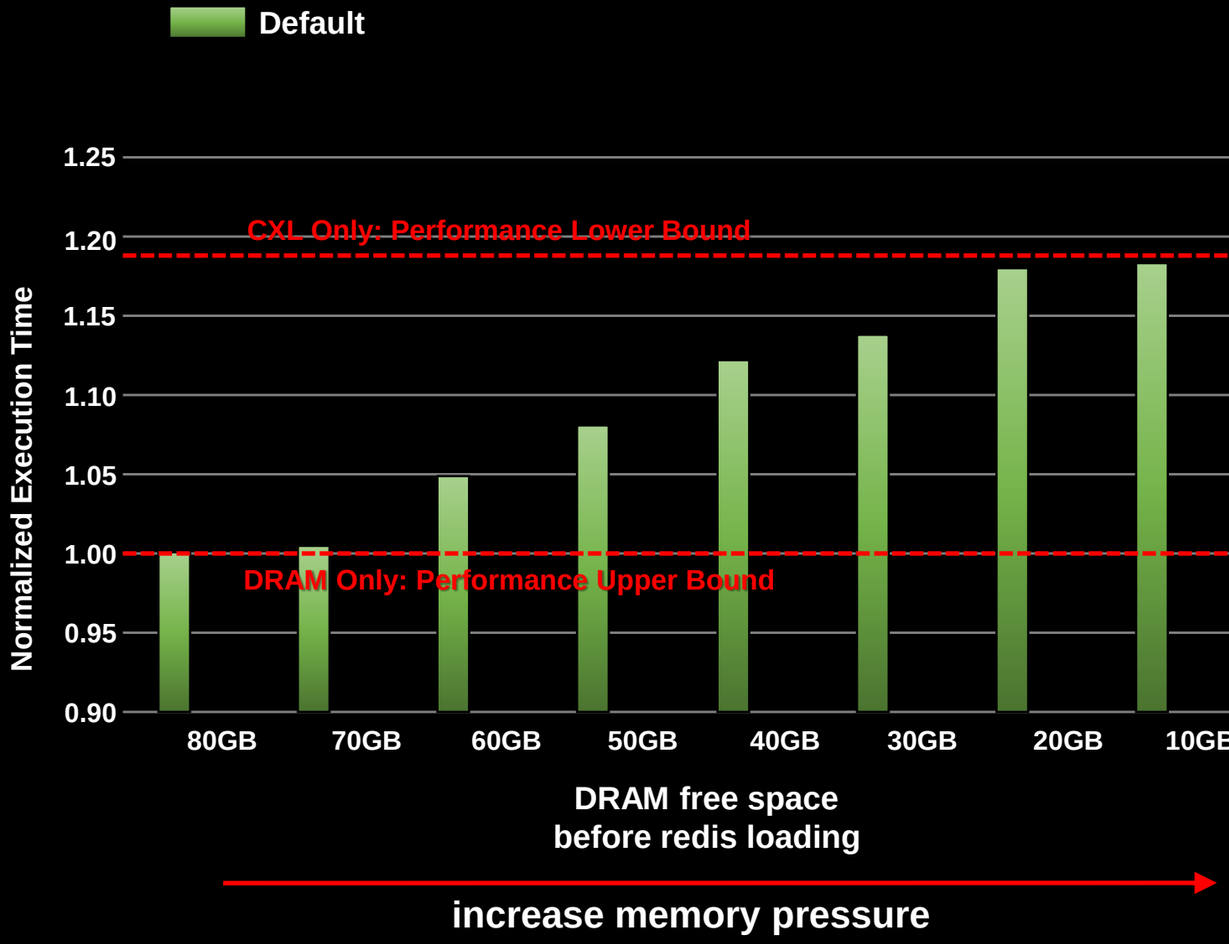


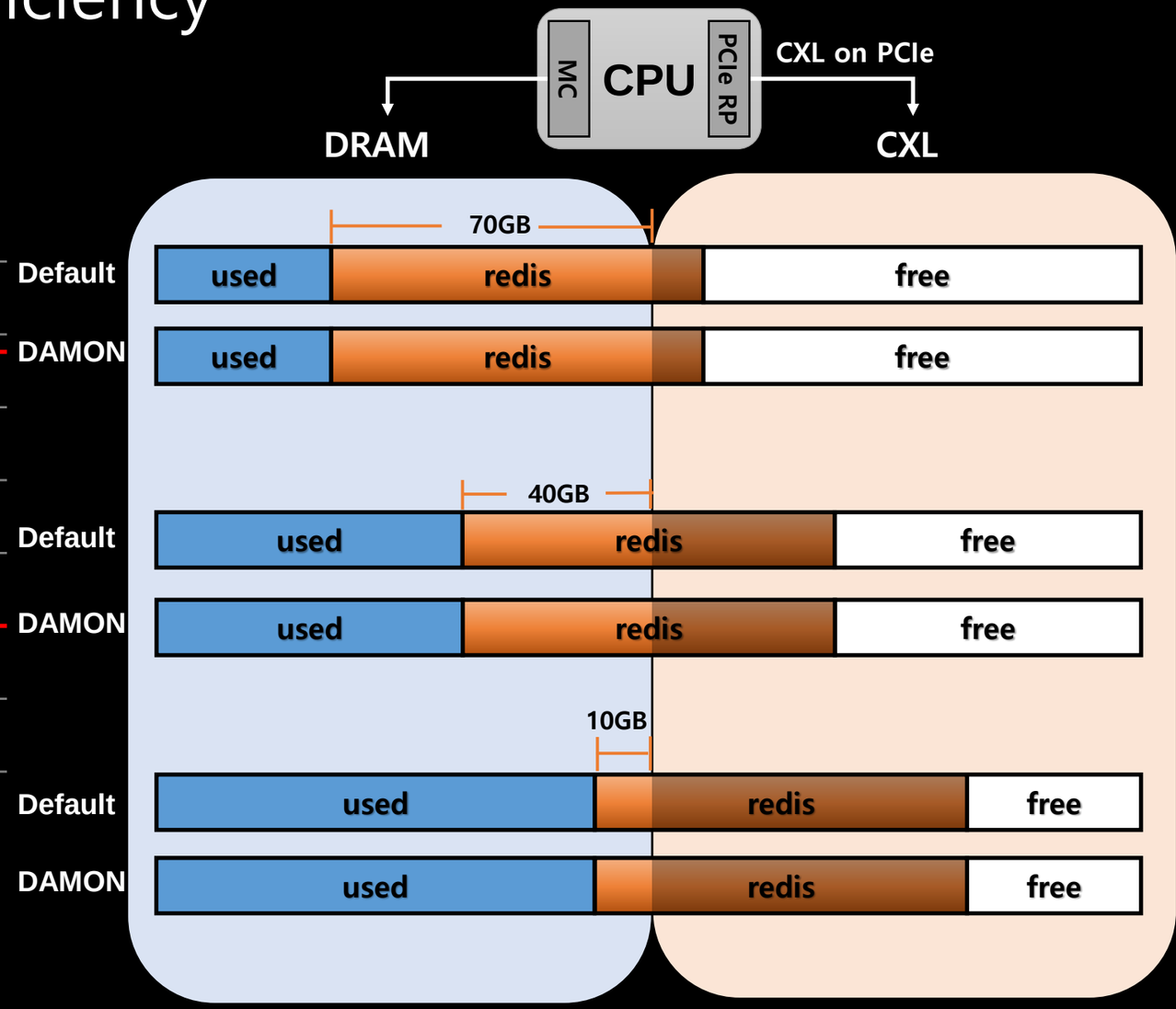


<Default>

1. DRAM is partially used by non-redis cold data.
2. Partial redis data is allocated on CXL memory.
   (due to insufficient space on DRAM)

HMSDK: Enhancing CXL Memory Efficiency



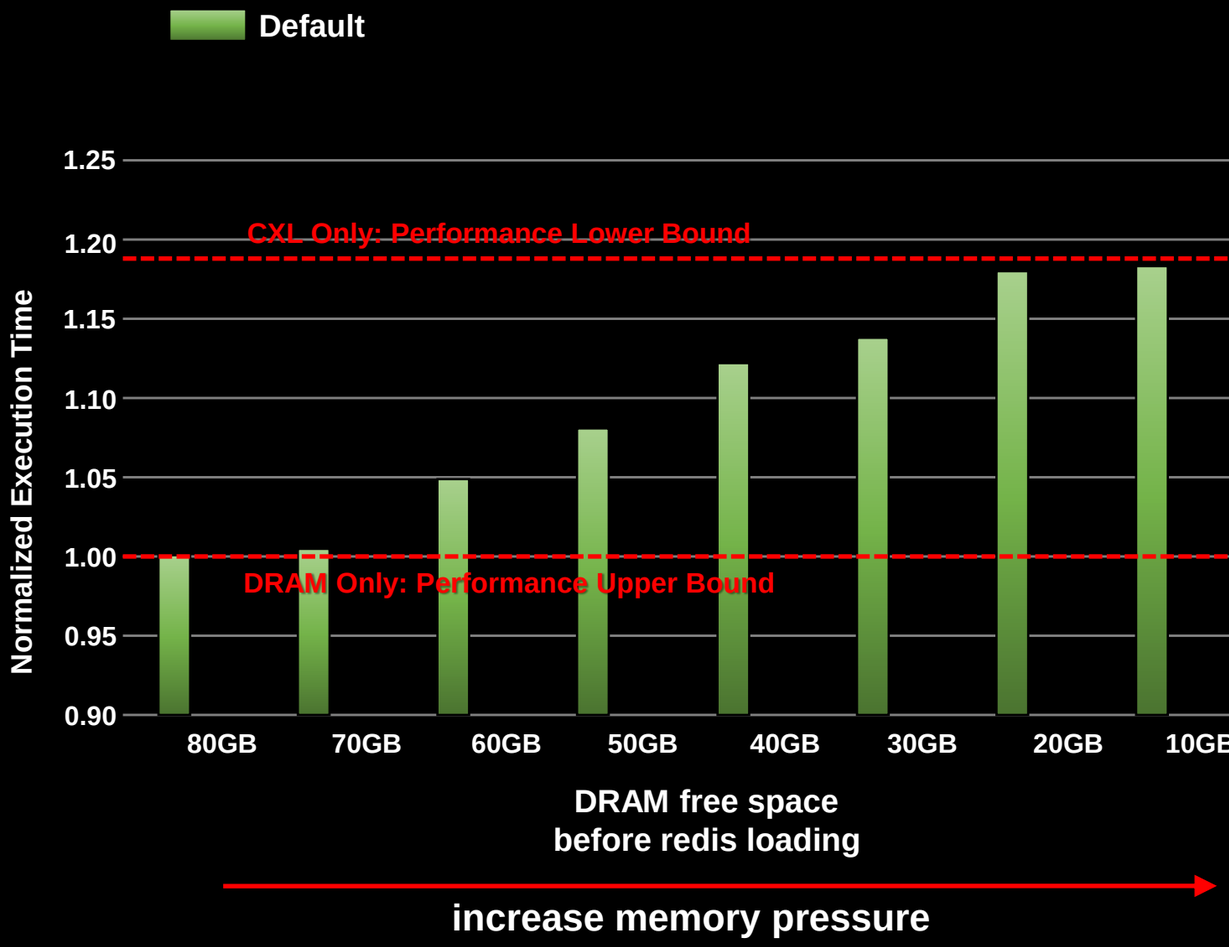


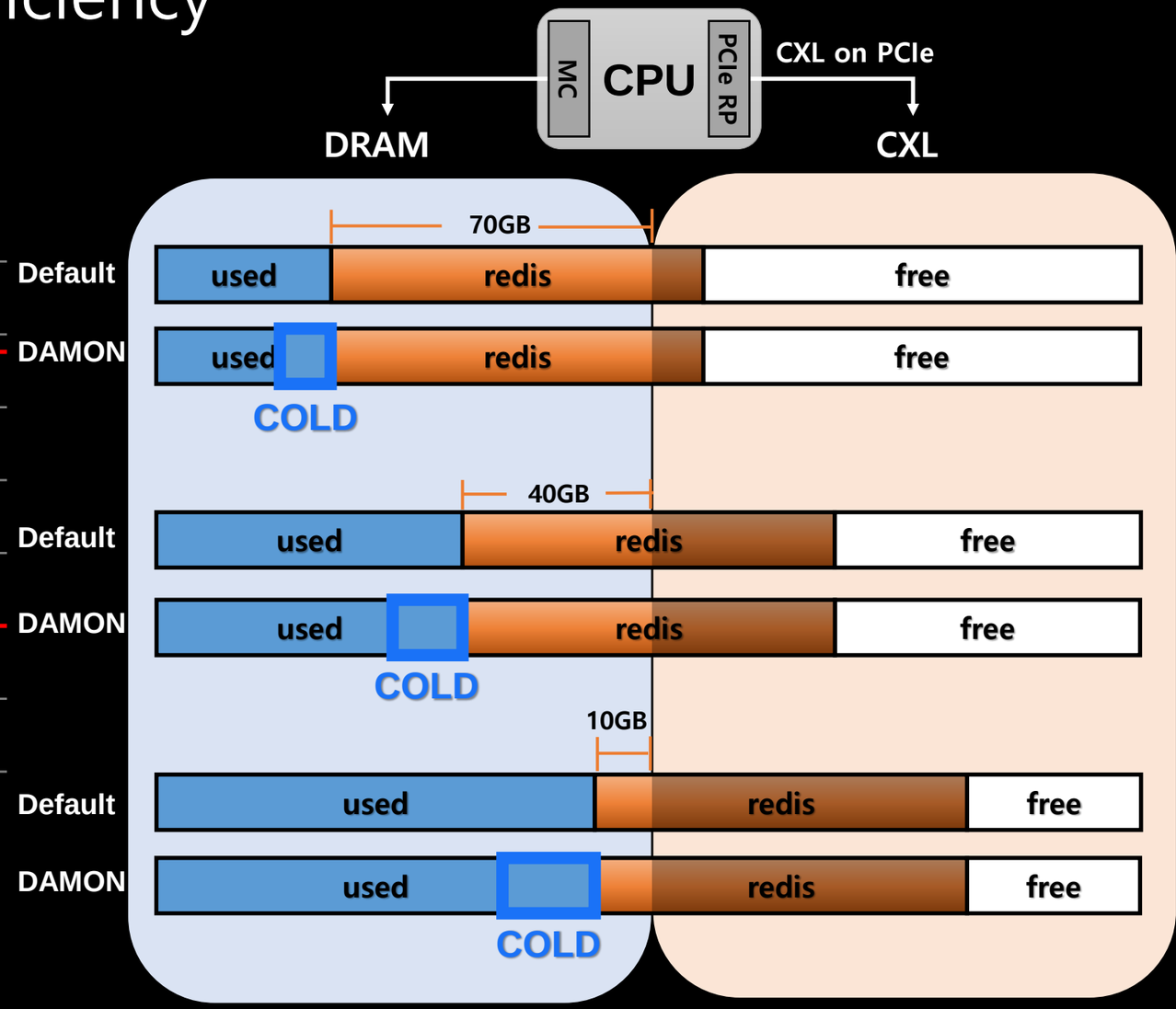
<Default>

1. DRAM is partially used by non-redis cold data.
2. Partial redis data is allocated on CXL memory.
   (due to insufficient space on DRAM)

<HMSDK>

HMSDK: Enhancing CXL Memory Efficiency





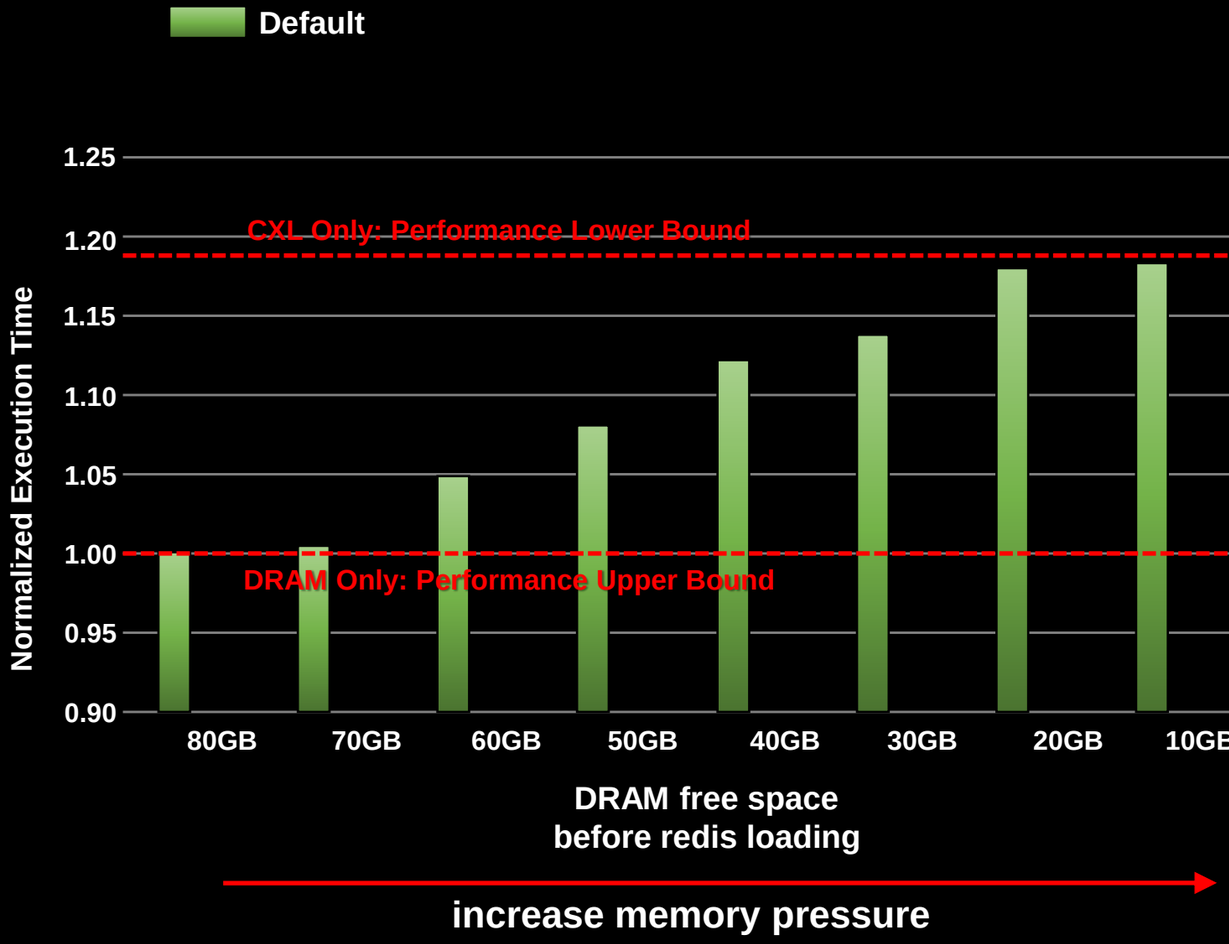
<Default>

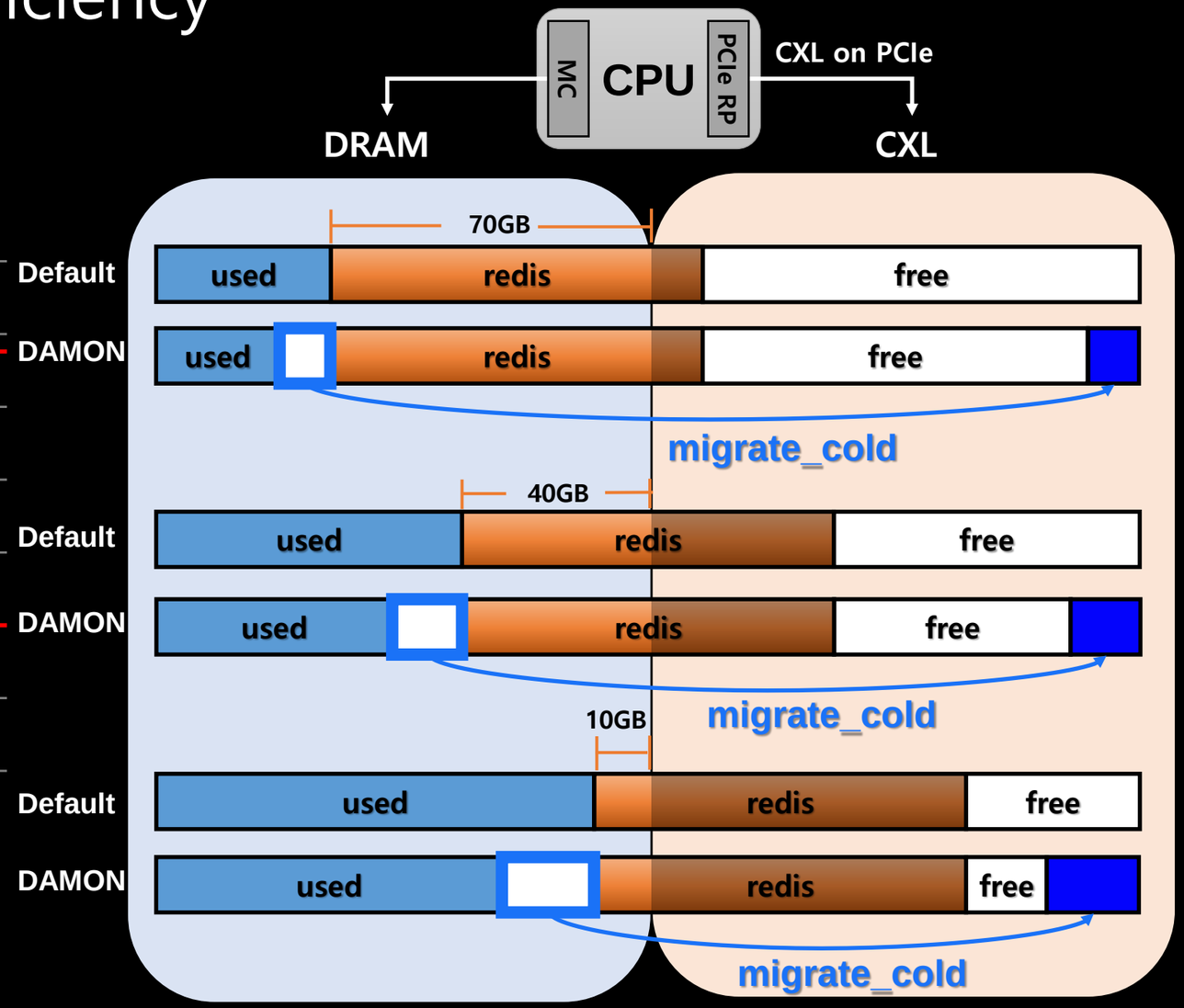
1. DRAM is partially used by non-redis cold data.
2. Partial redis data is allocated on CXL memory.
   (due to insufficient space on DRAM)

<HMSDK>

1. Demote cold data from DRAM to CXL memory.

HMSDK: Enhancing CXL Memory Efficiency





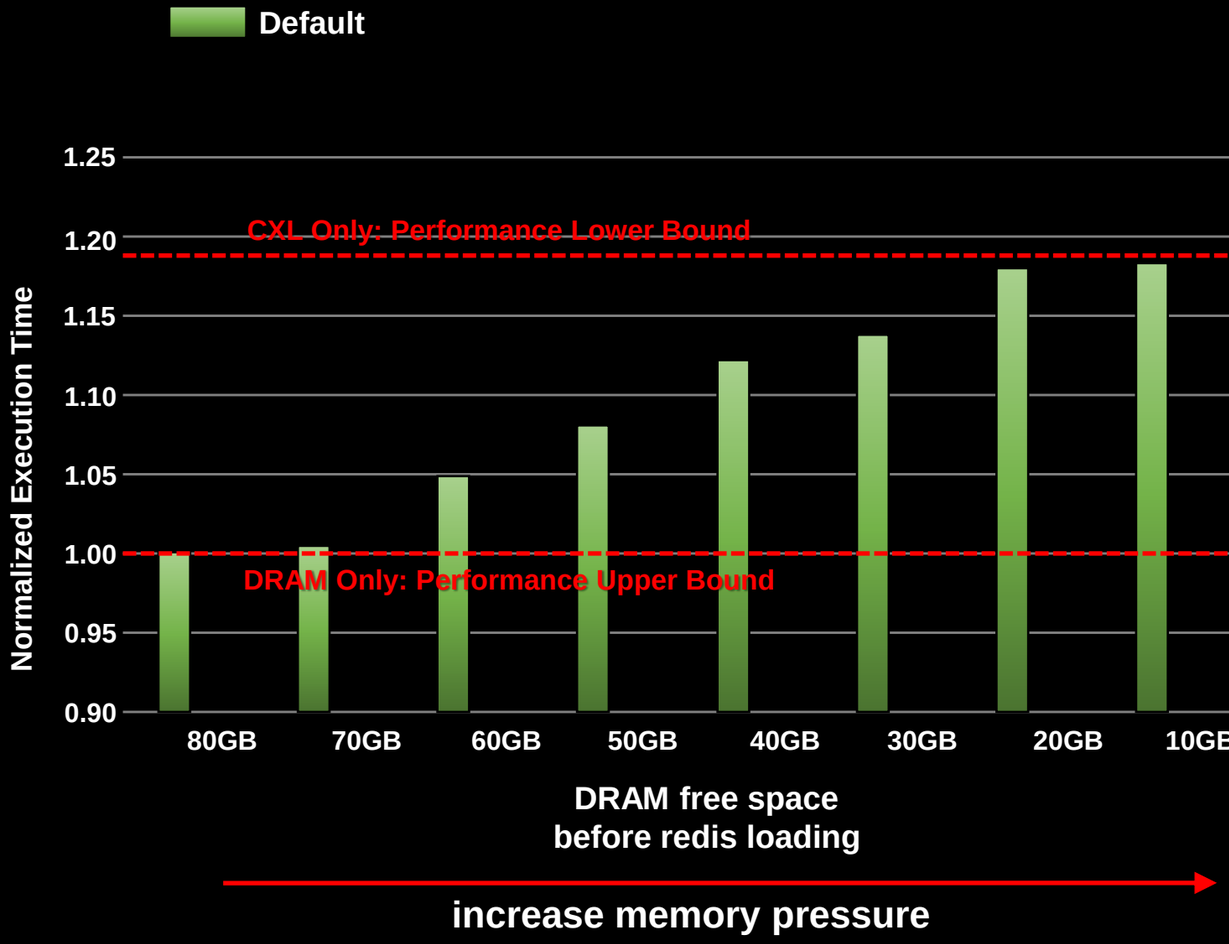
<Default>

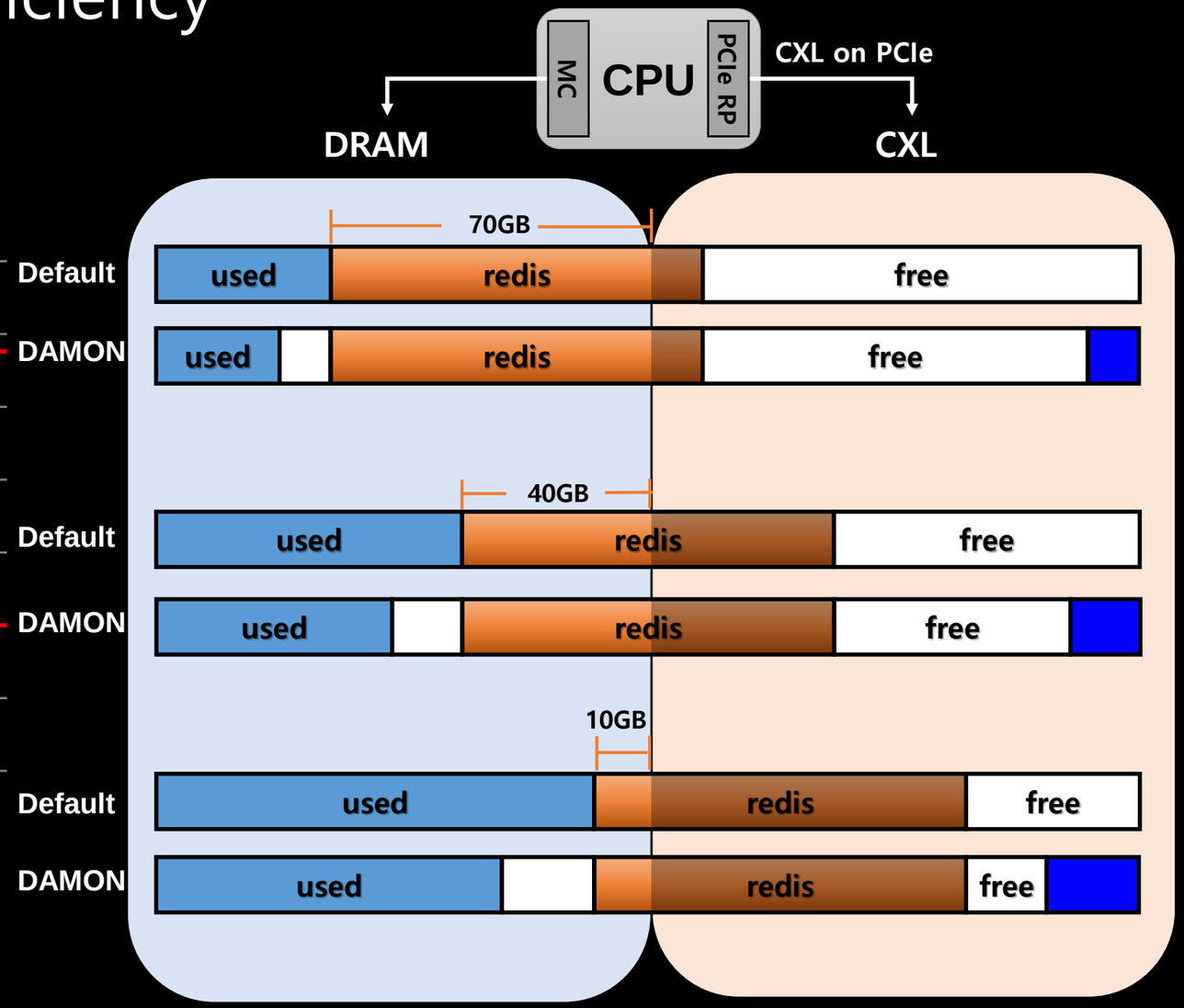
1. DRAM is partially used by non-redis cold data.
2. Partial redis data is allocated on CXL memory.
   (due to insufficient space on DRAM)

<HMSDK>

1. Demote cold data from DRAM to CXL memory.

HMSDK: Enhancing CXL Memory Efficiency





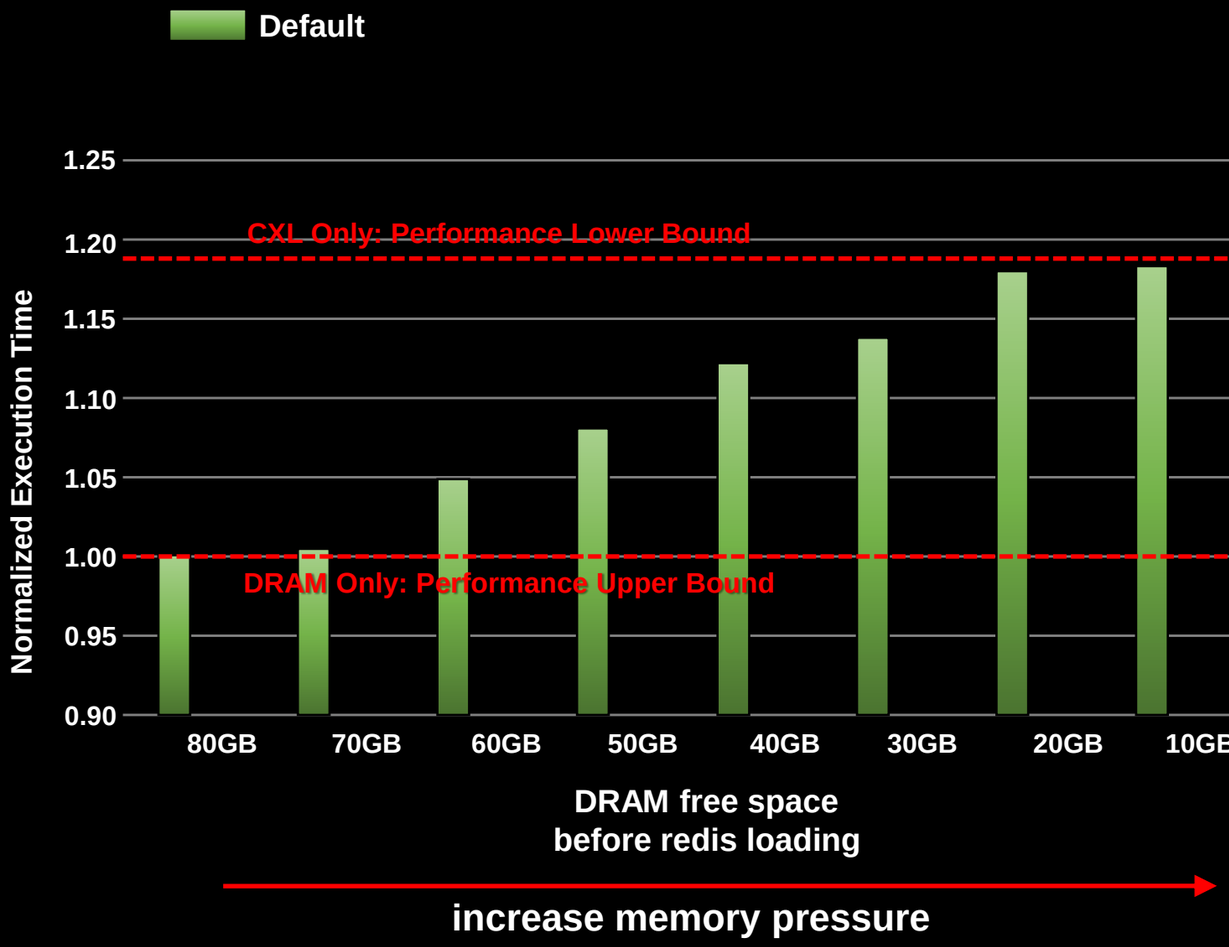
<Default>

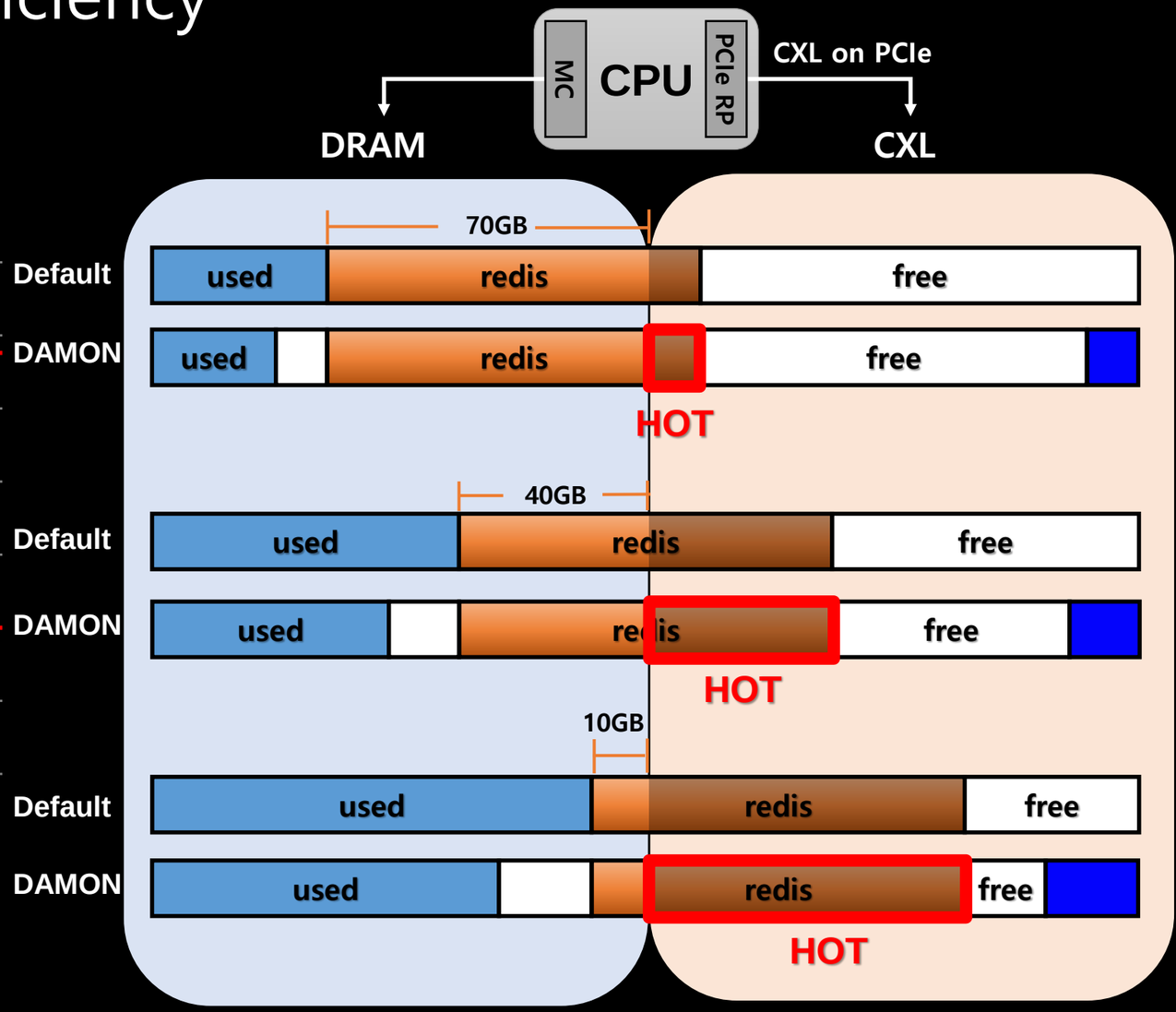
1. DRAM is partially used by non-redis cold data.
2. Partial redis data is allocated on CXL memory.
   (due to insufficient space on DRAM)

<HMSDK>

1. Demote cold data from DRAM to CXL memory.

HMSDK: Enhancing CXL Memory Efficiency





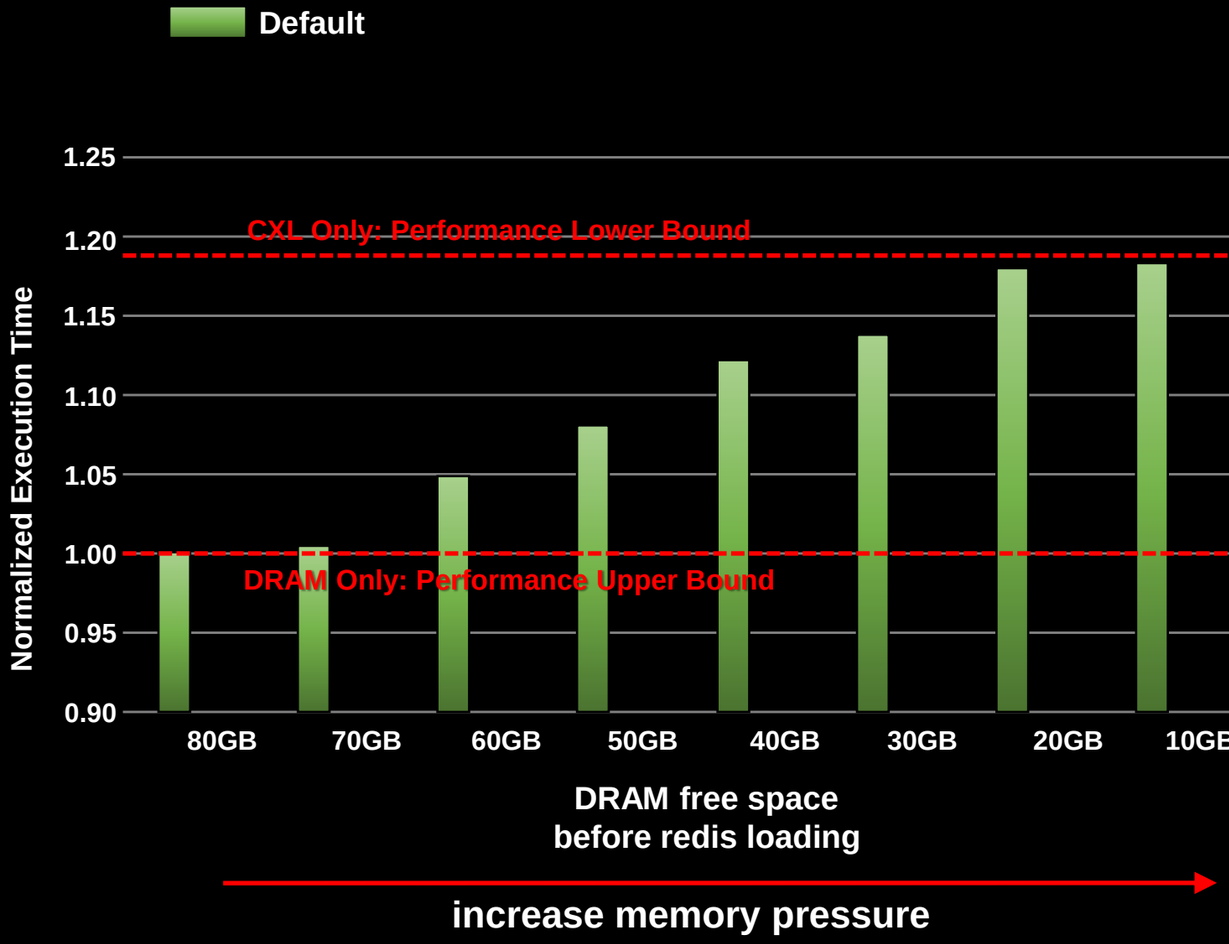
<Default>

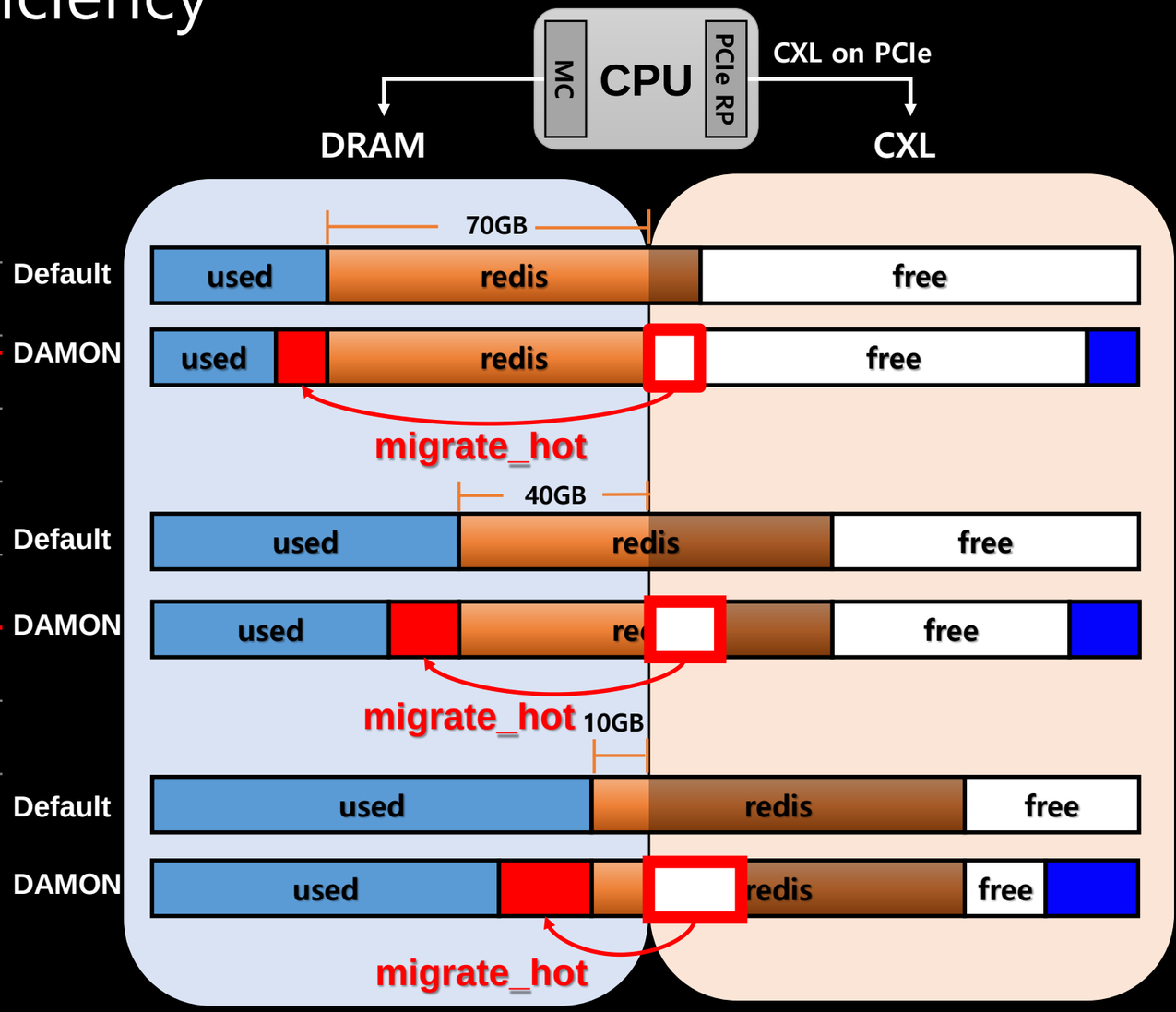
1. DRAM is partially used by non-redis cold data.
2. Partial redis data is allocated on CXL memory.
   (due to insufficient space on DRAM)

<HMSDK>

1. Demote cold data from DRAM to CXL memory.
2. Promote more redis data to fast DRAM.
   (while keeping cold data on CXL memory)

HMSDK: Enhancing CXL Memory Efficiency

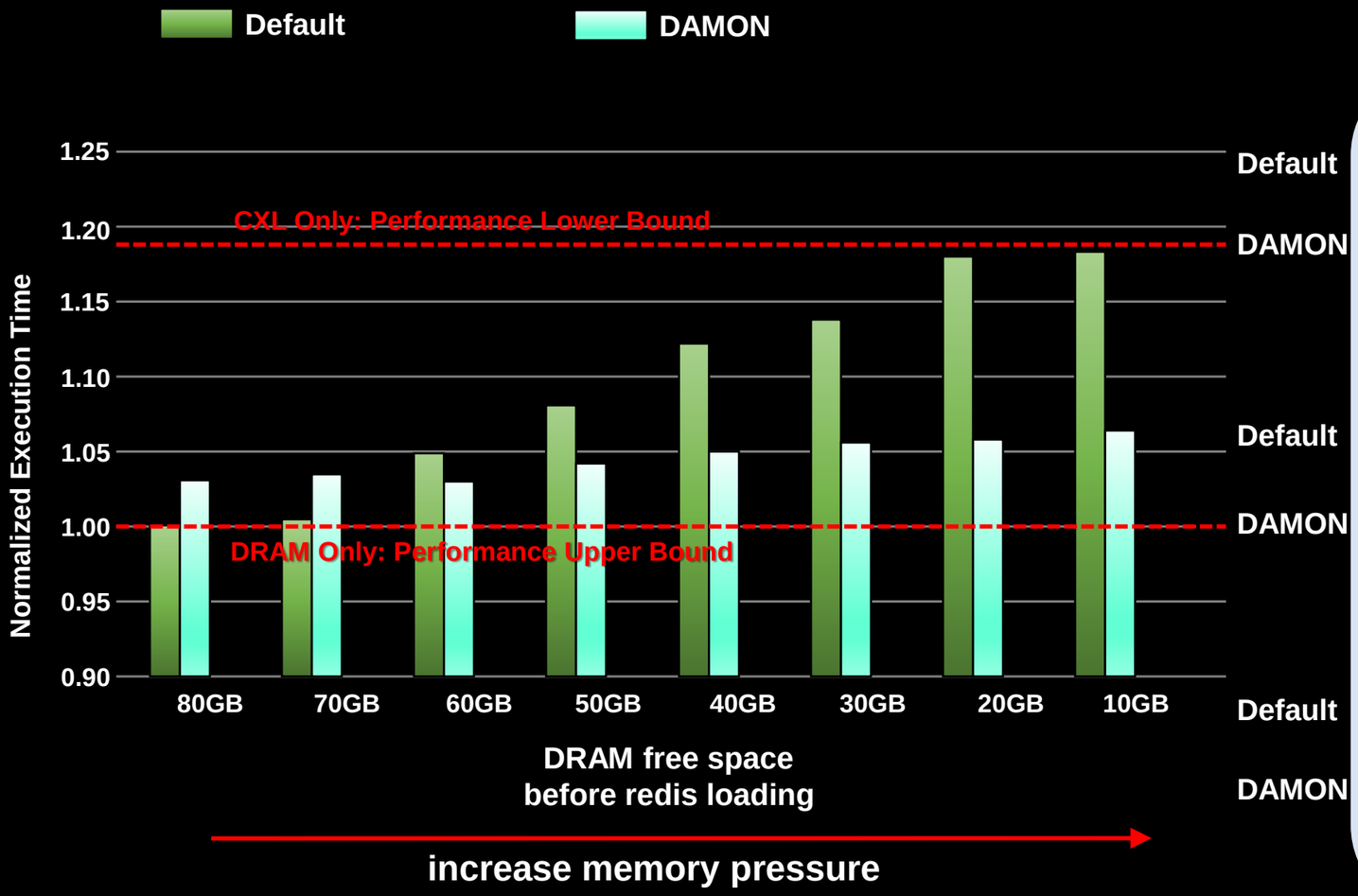


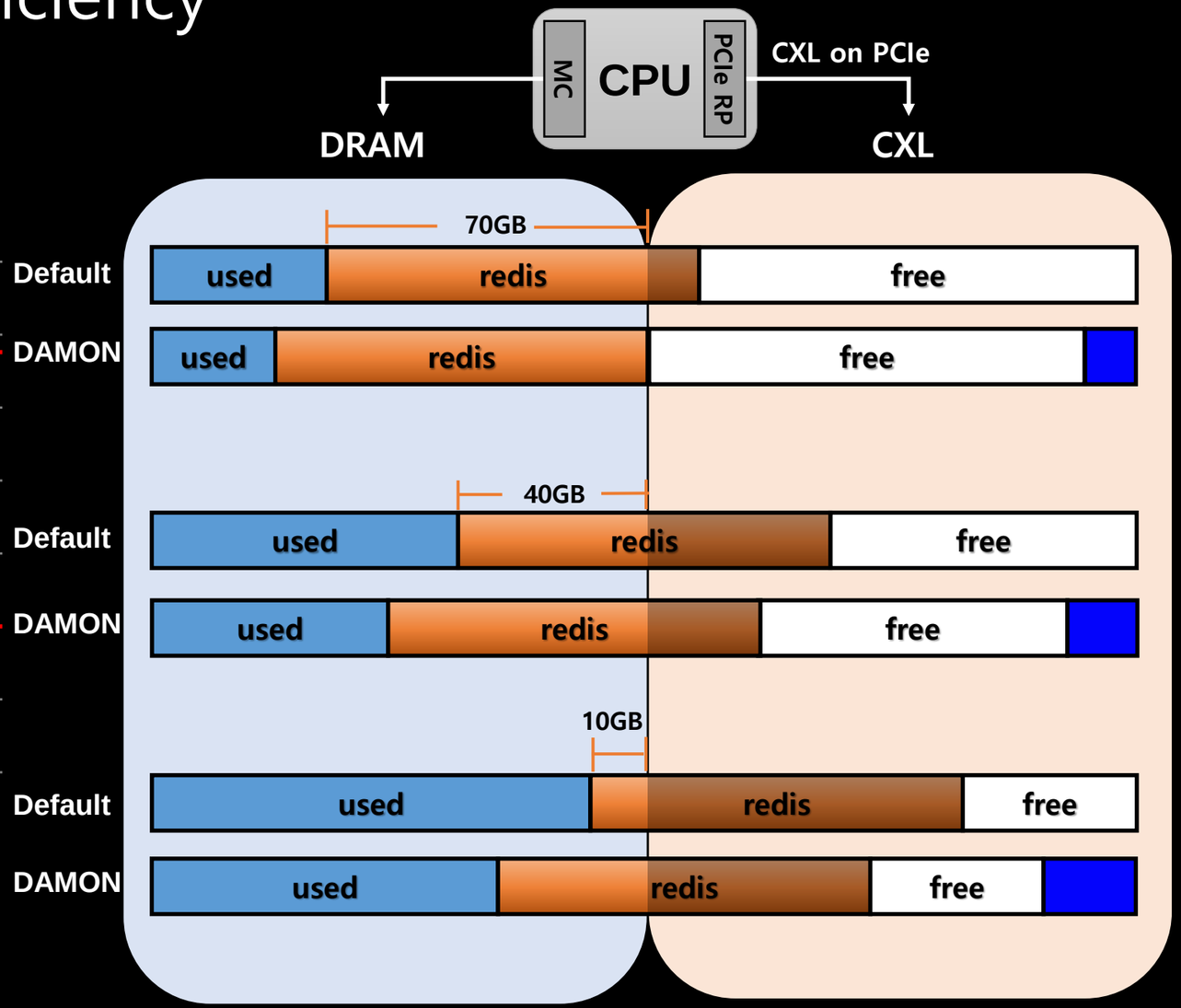


<Default>
1. DRAM is partially used by non-redis cold data.
2. Partial redis data is allocated on CXL memory.
   (due to insufficient space on DRAM)

<HMSDK>
1. Demote cold data from DRAM to CXL memory.
2. Promote more redis data to fast DRAM.
   (while keeping cold data on CXL memory)

HMSDK: Enhancing CXL Memory Efficiency





<Default>

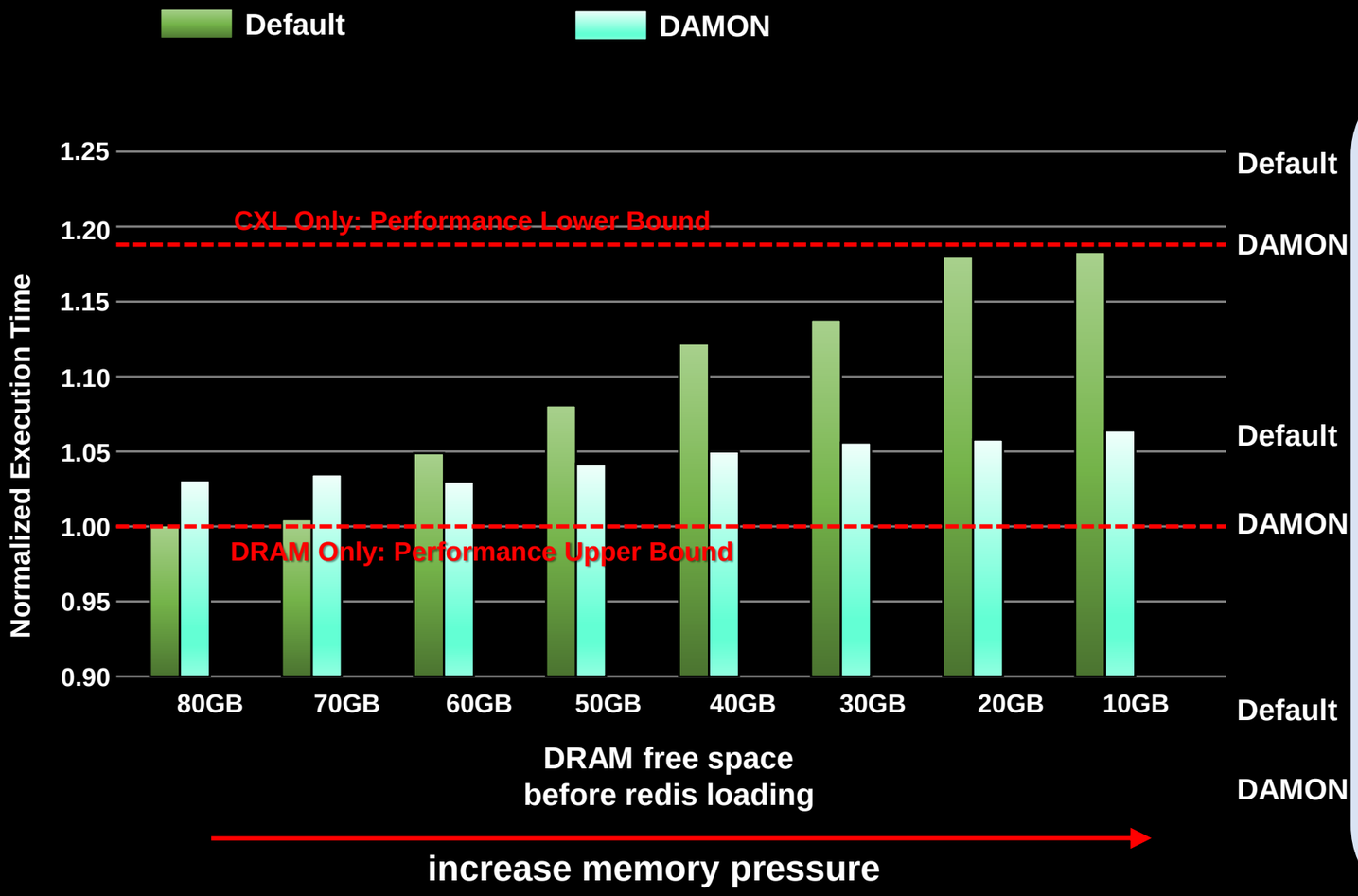
1. DRAM is partially used by non-redis cold data.
2. Partial redis data is allocated on CXL memory.
   (due to insufficient space on DRAM)

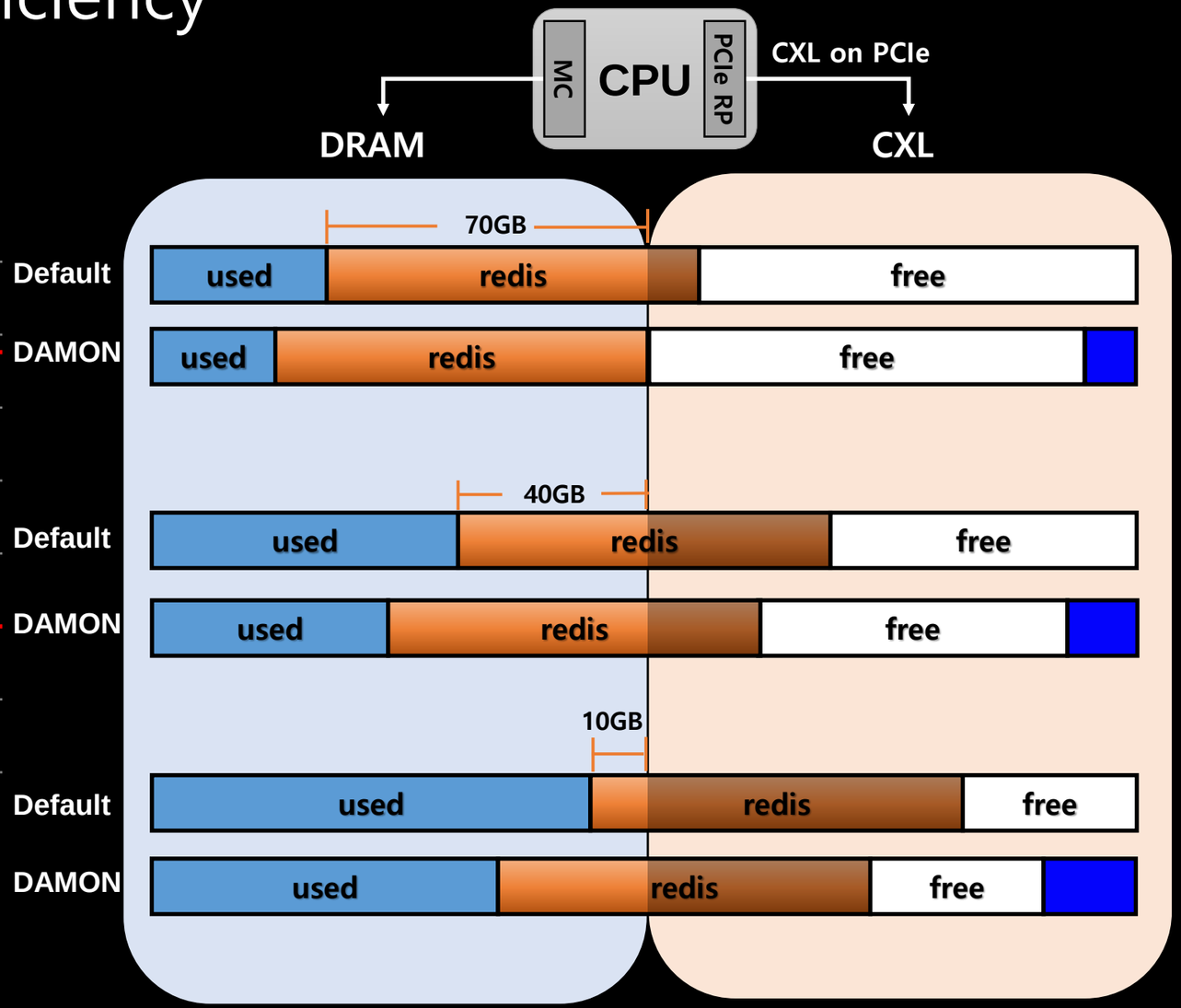
<HMSDK>

1. Demote cold data from DRAM to CXL memory.
2. Promote more redis data to fast DRAM.
   (while keeping cold data on CXL memory)

If demotion_enabled is on (kswapd helps demotion instead of swap out)

HMSDK: Enhancing CXL Memory Efficiency





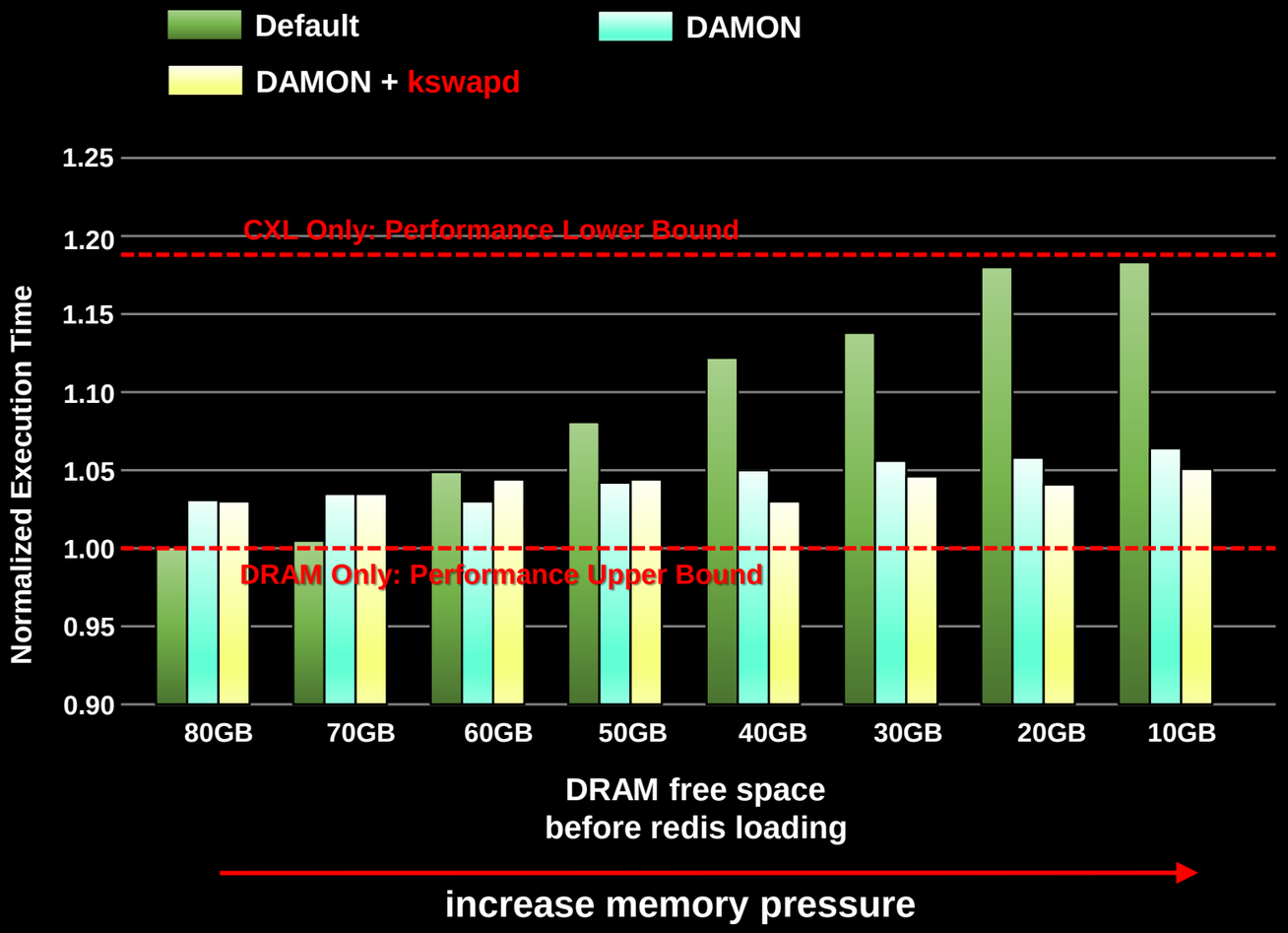
<Default>

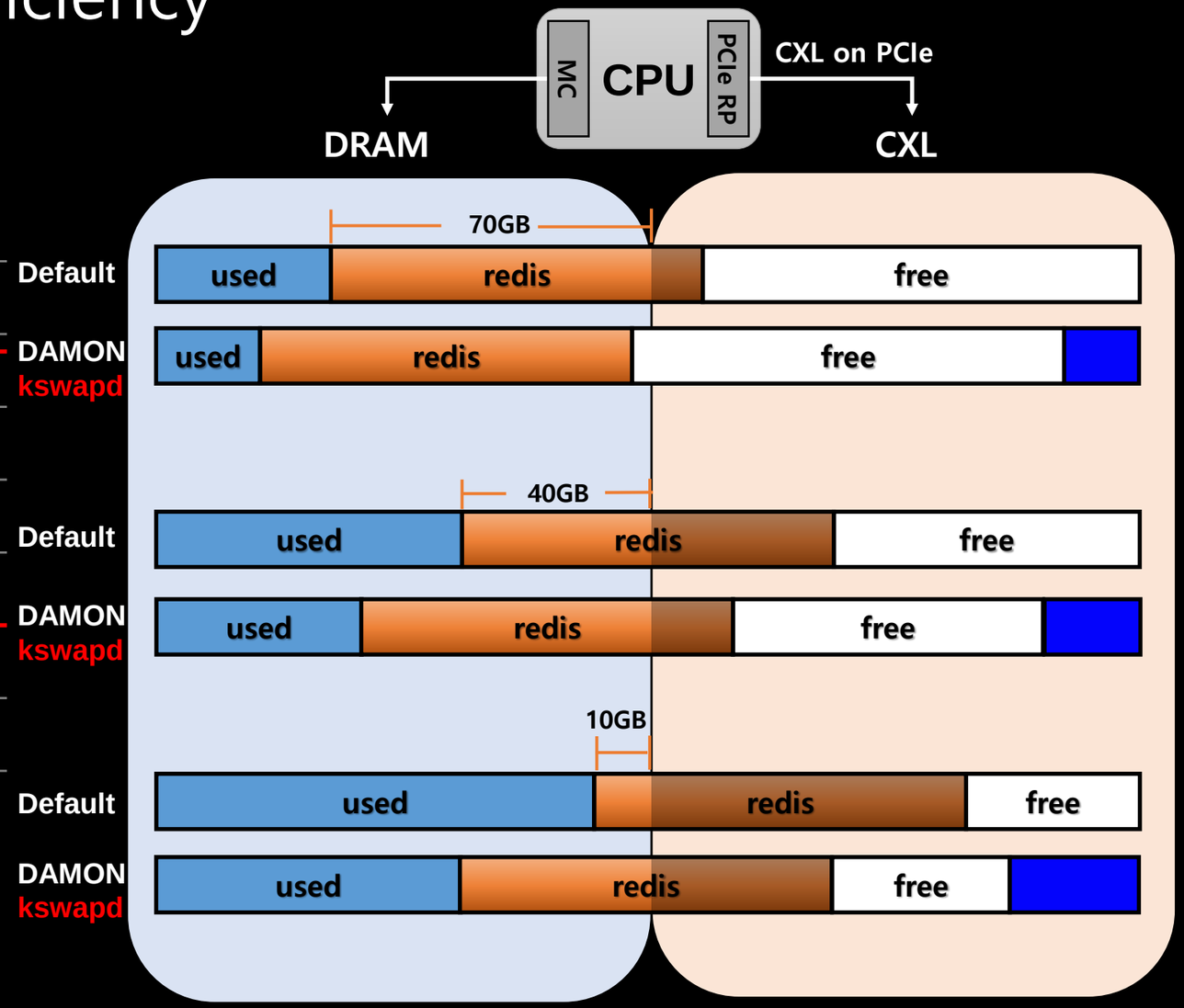
1. DRAM is partially used by non-redis cold data.
2. Partial redis data is allocated on CXL memory.
   (due to insufficient space on DRAM)

<HMSDK>

1. Demote cold data from DRAM to CXL memory.
2. Promote more redis data to fast DRAM.
   (while keeping cold data on CXL memory)

HMSDK: Enhancing CXL Memory Efficiency





<Default>

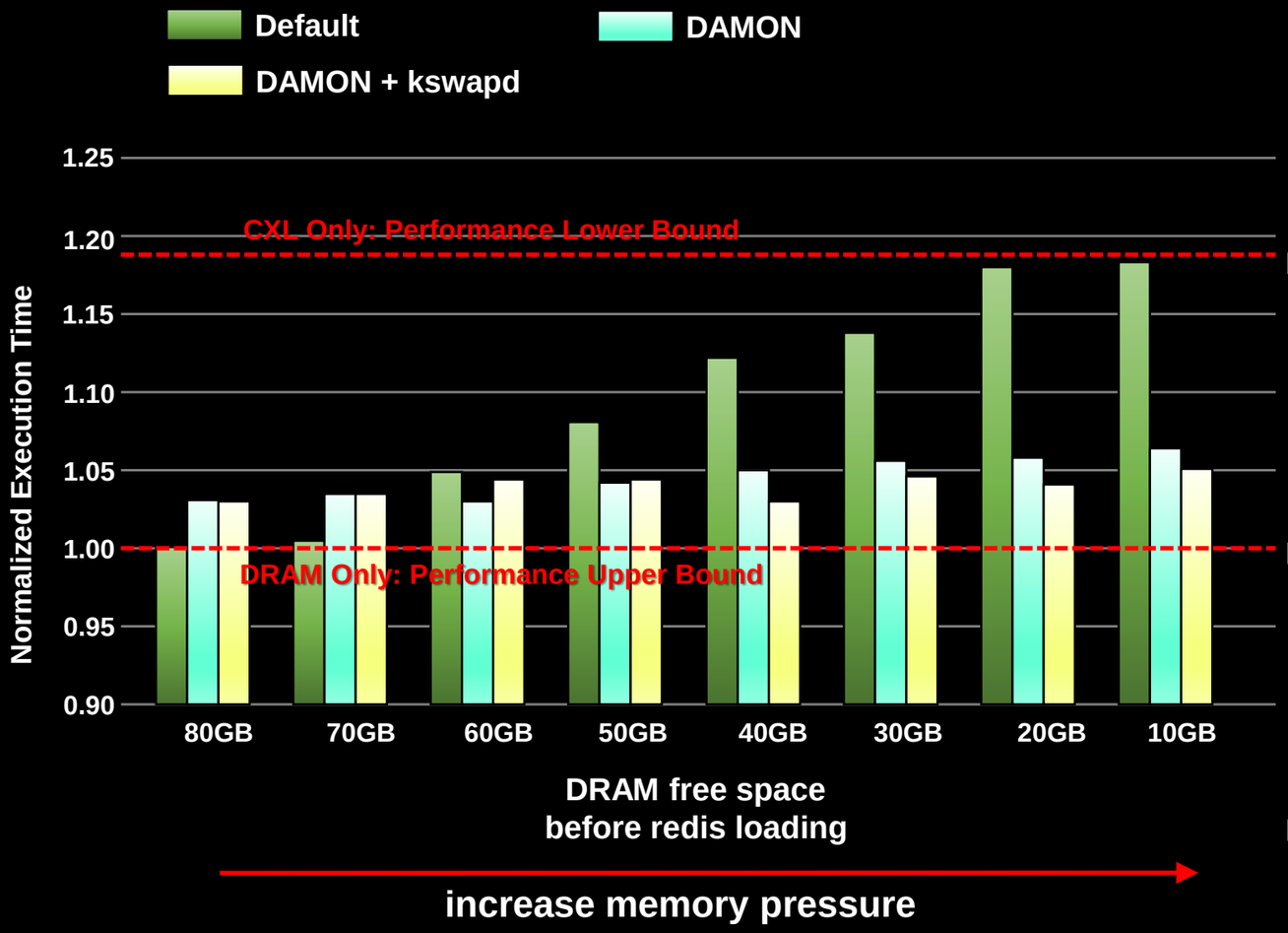
1. DRAM is partially used by non-redis cold data.
2. Partial redis data is allocated on CXL memory.
   (due to insufficient space on DRAM)

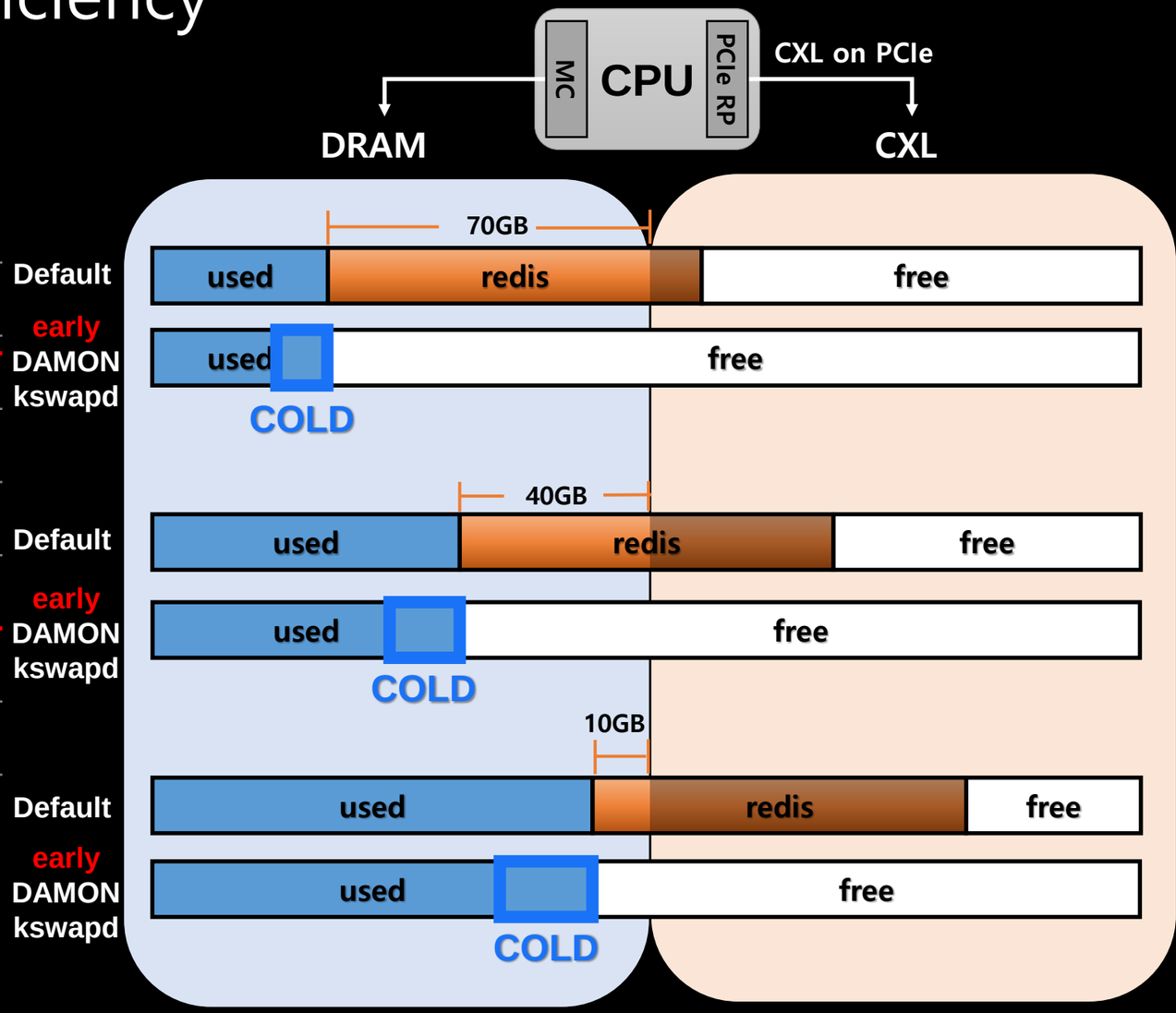
<HMSDK>

1. Demote cold data from DRAM to CXL memory.
2. Promote more redis data to fast DRAM.
   (while keeping cold data on CXL memory)

If DAMON is always on
(early DAMON result)

HMSDK: Enhancing CXL Memory Efficiency





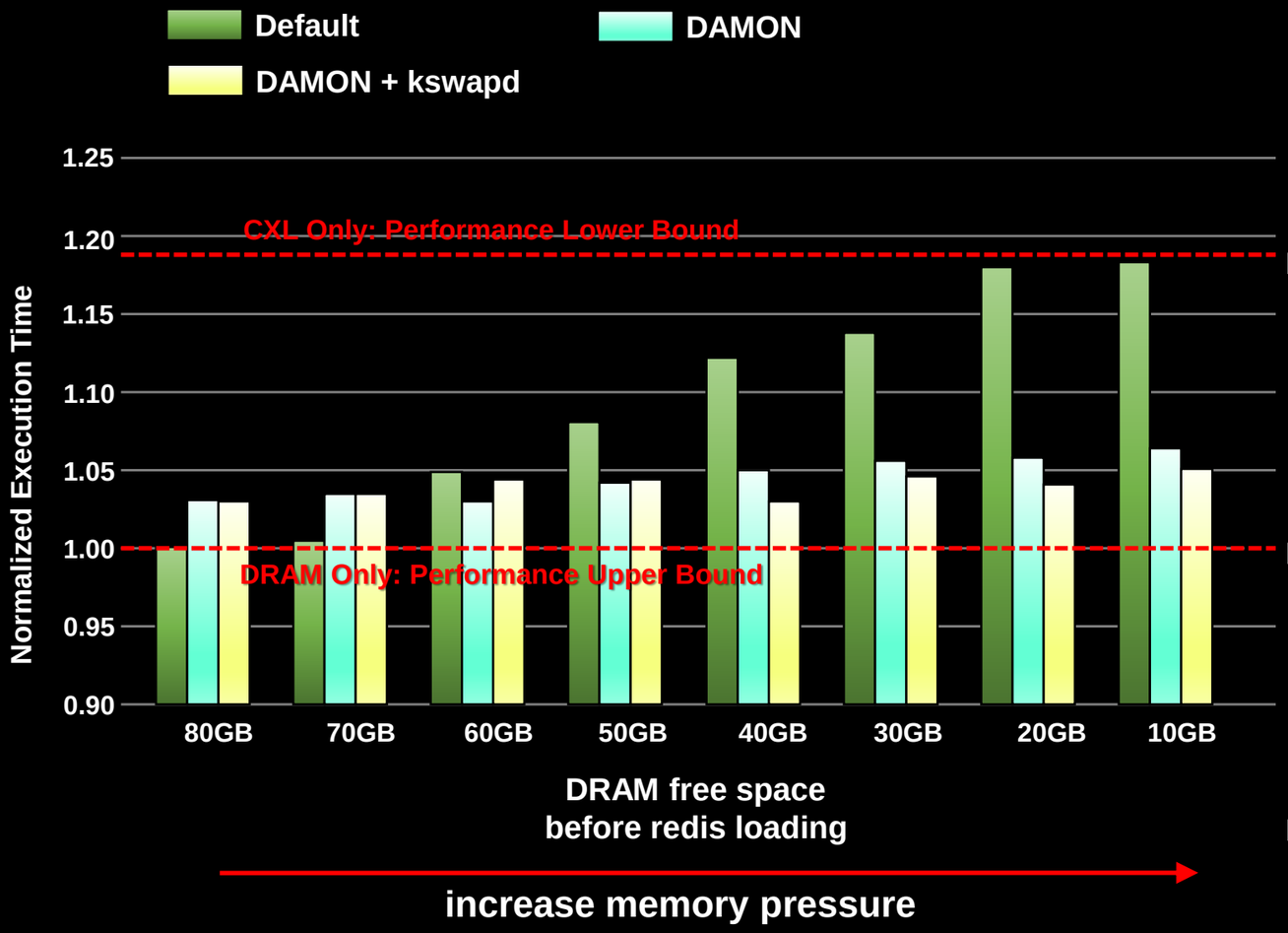
<Default>

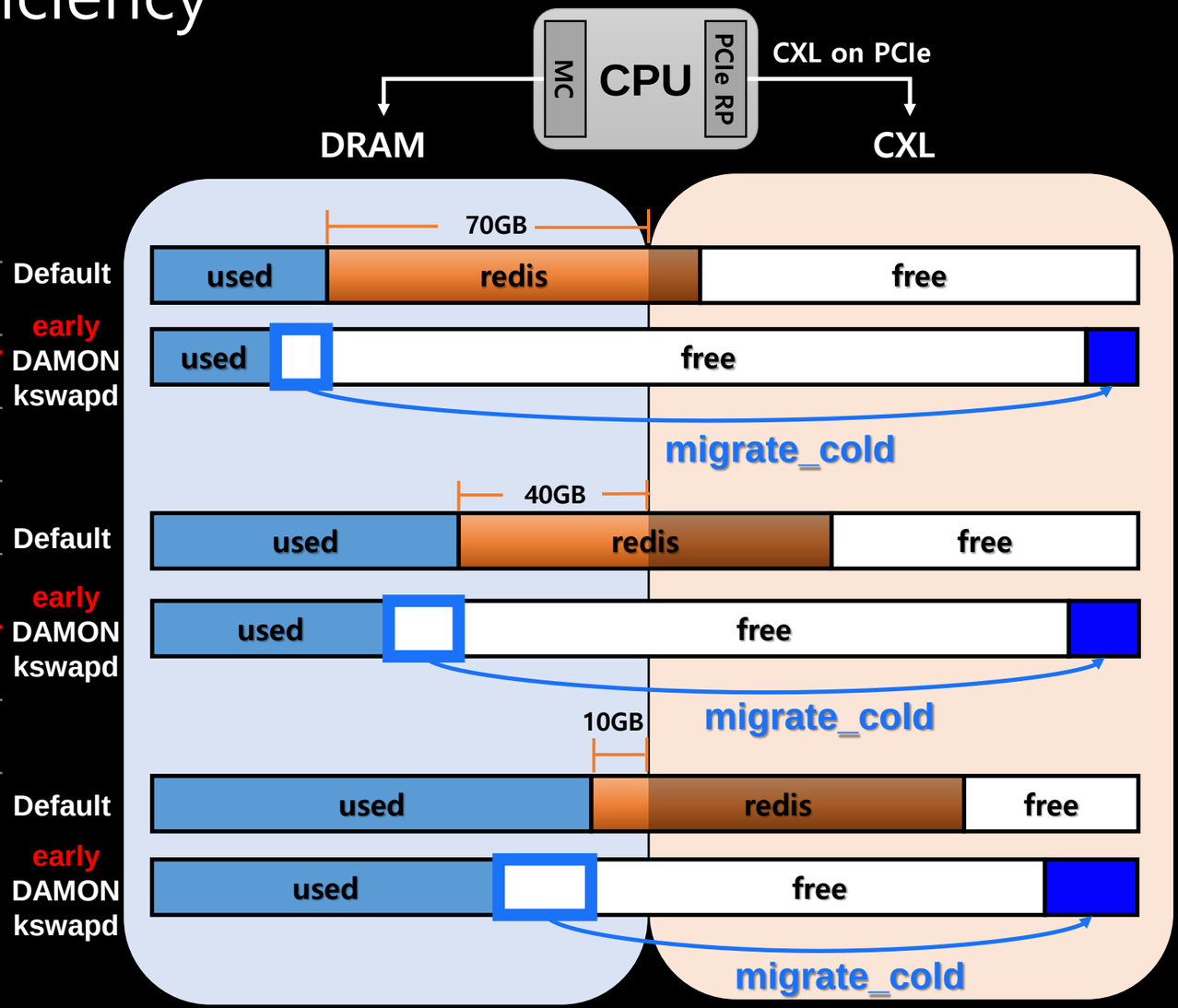
1. DRAM is partially used by non-redis cold data.
2. Partial redis data is allocated on CXL memory.
   (due to insufficient space on DRAM)

<HMSDK>

1. Demote cold data from DRAM to CXL memory.

HMSDK: Enhancing CXL Memory Efficiency





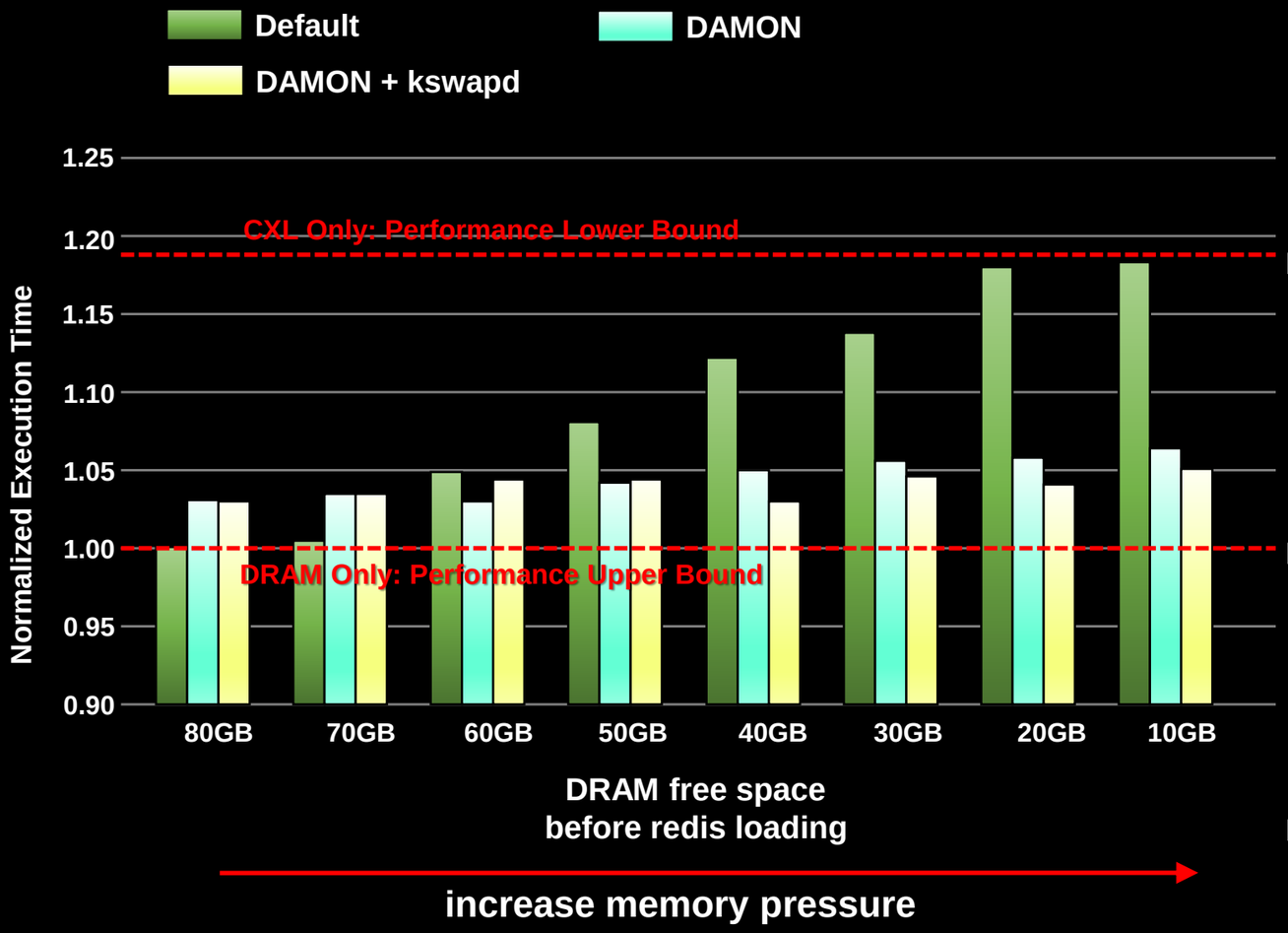
<Default>

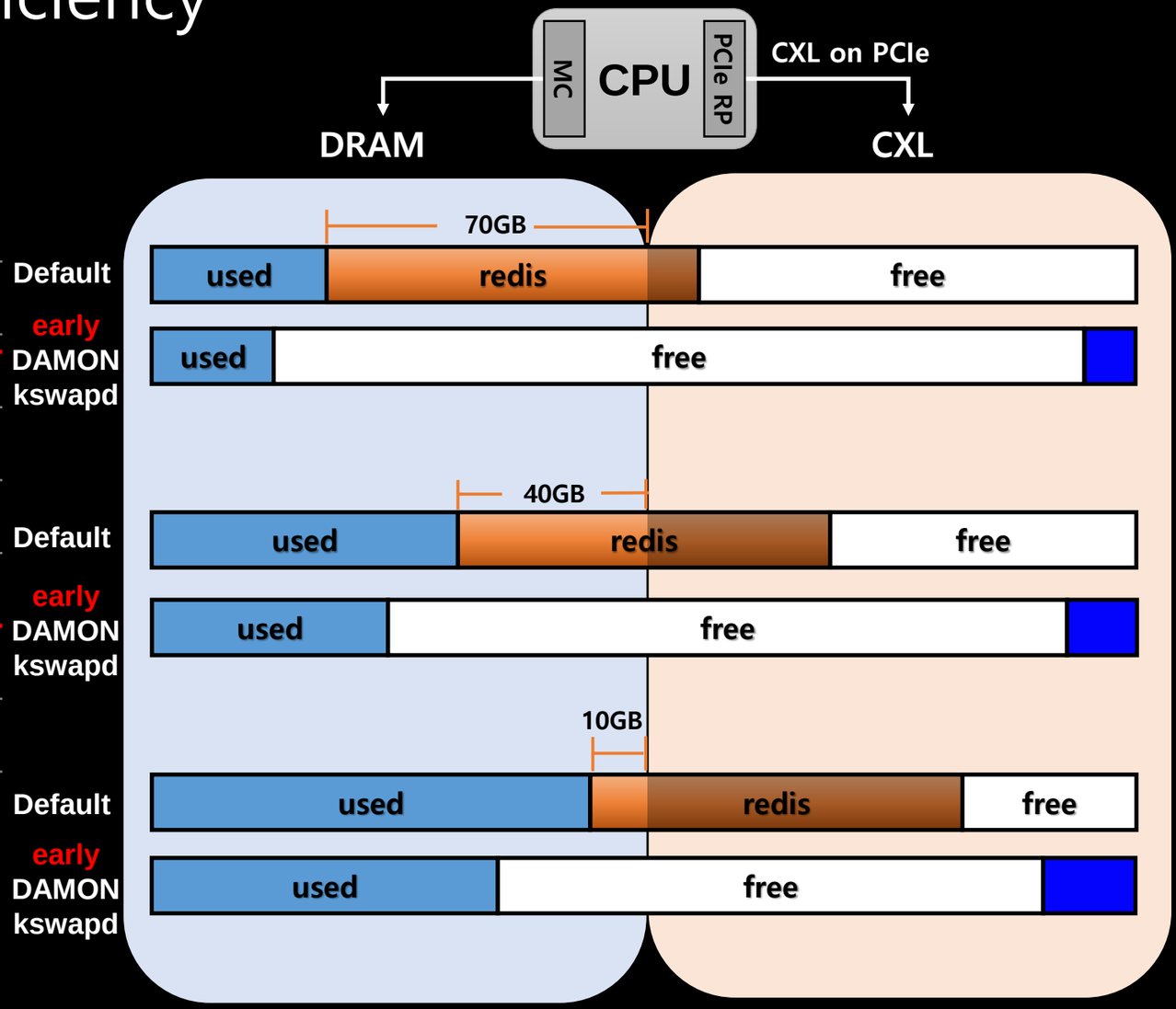
1. DRAM is partially used by non-redis cold data.
2. Partial redis data is allocated on CXL memory.
   (due to insufficient space on DRAM)

<HMSDK>

1. Demote cold data from DRAM to CXL memory.

HMSDK: Enhancing CXL Memory Efficiency





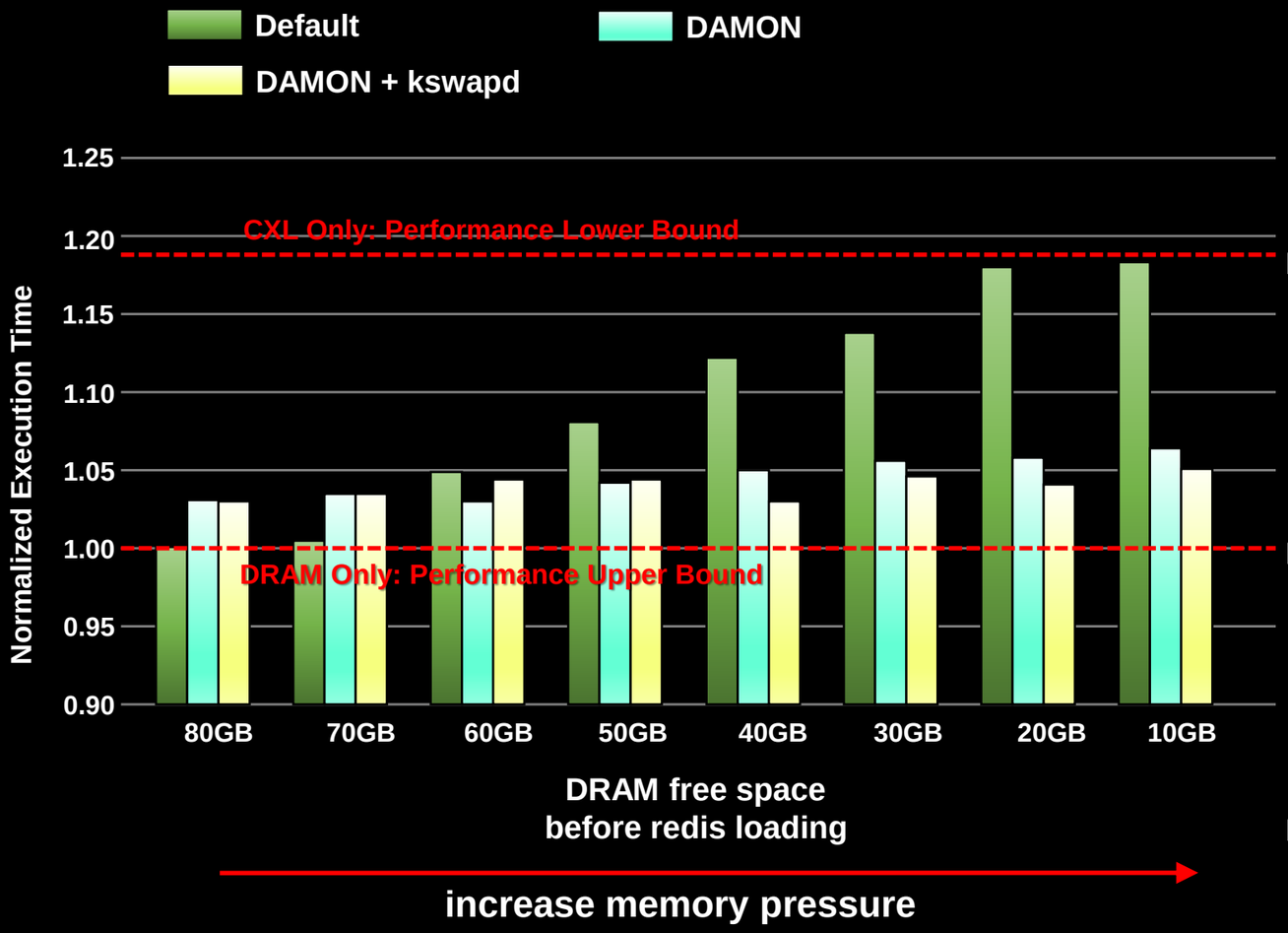
<Default>

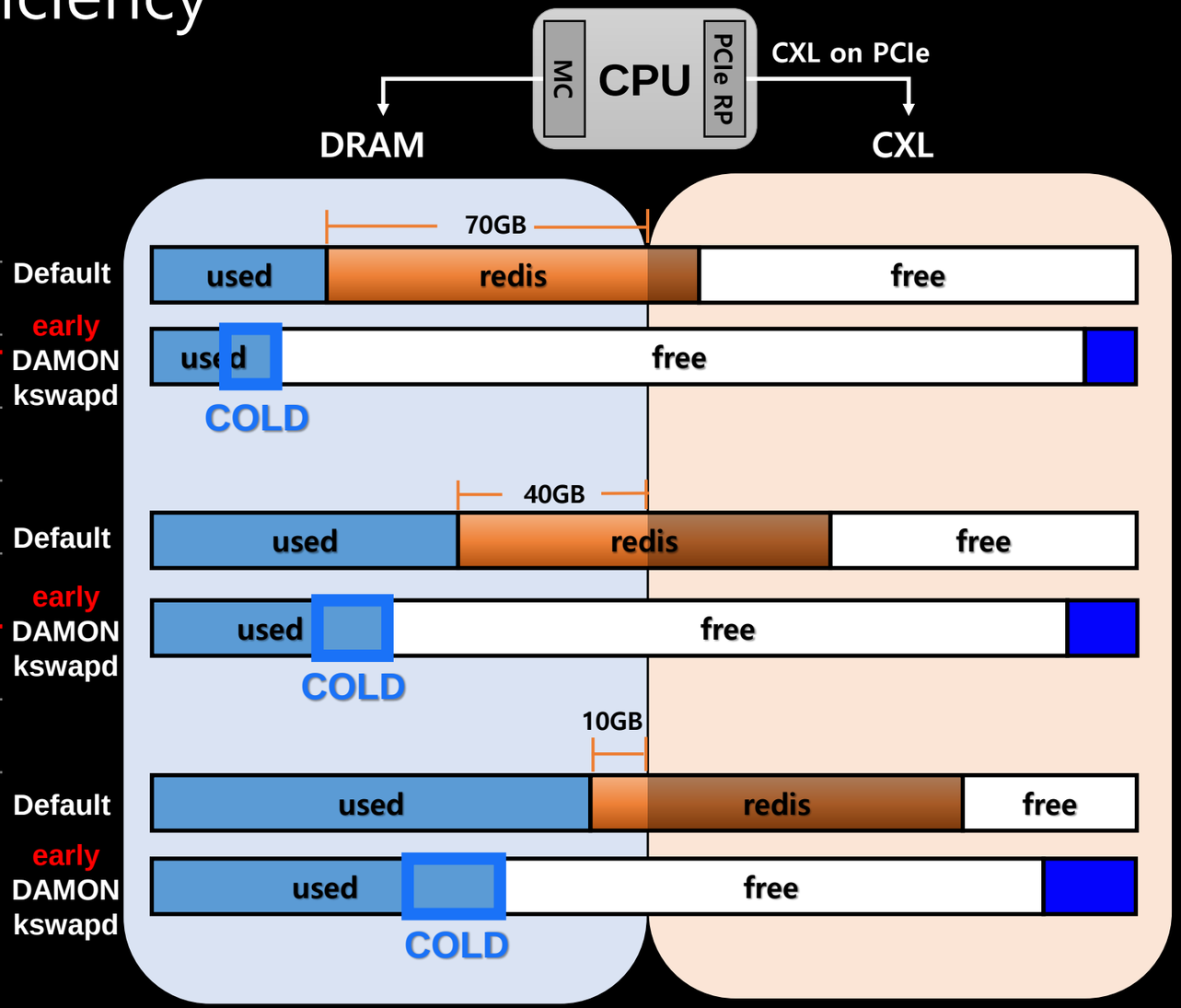
1. DRAM is partially used by non-redis cold data.
2. Partial redis data is allocated on CXL memory.
   (due to insufficient space on DRAM)

<HMSDK>

1. Demote cold data from DRAM to CXL memory.

HMSDK: Enhancing CXL Memory Efficiency





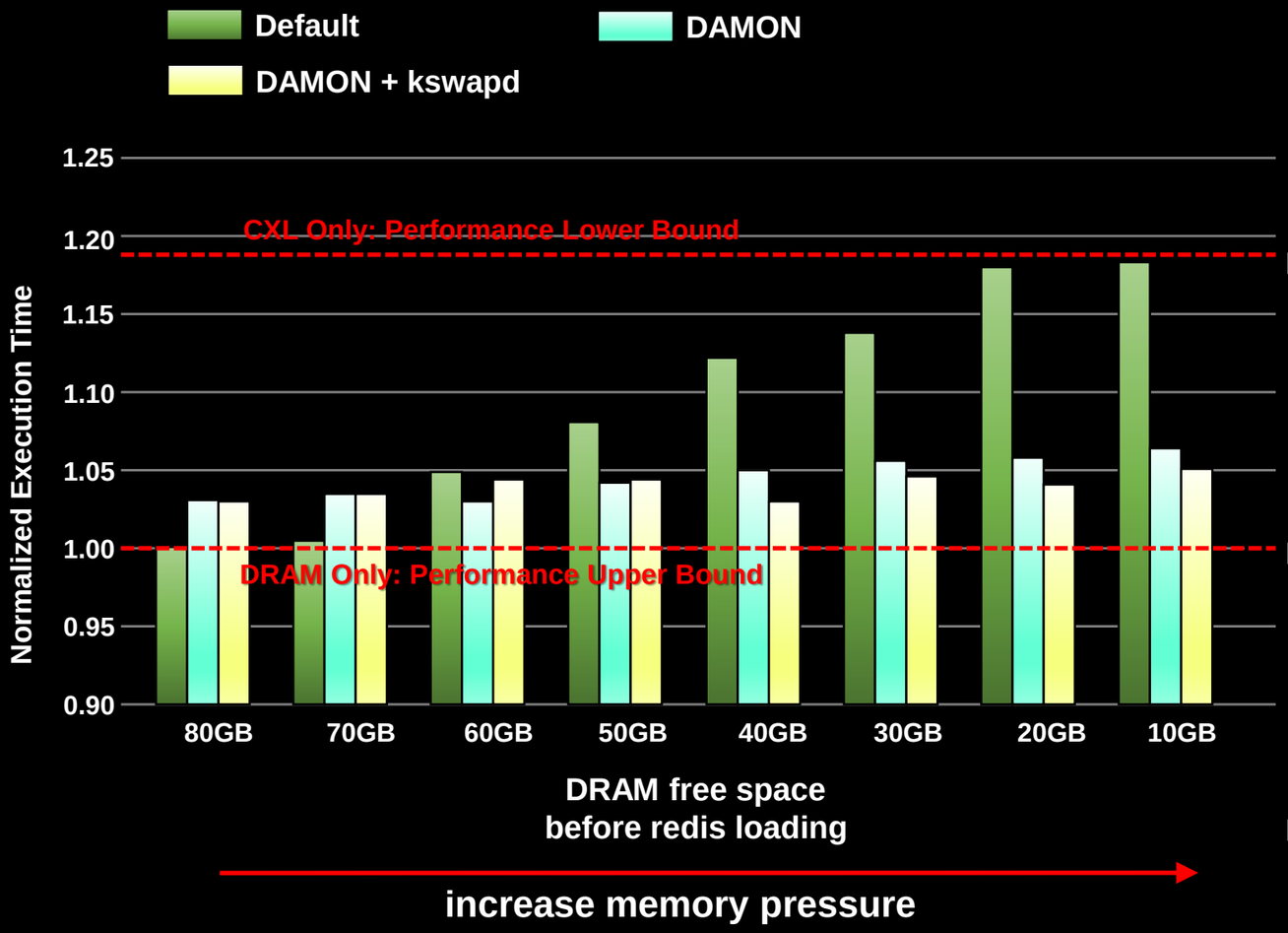
<Default>

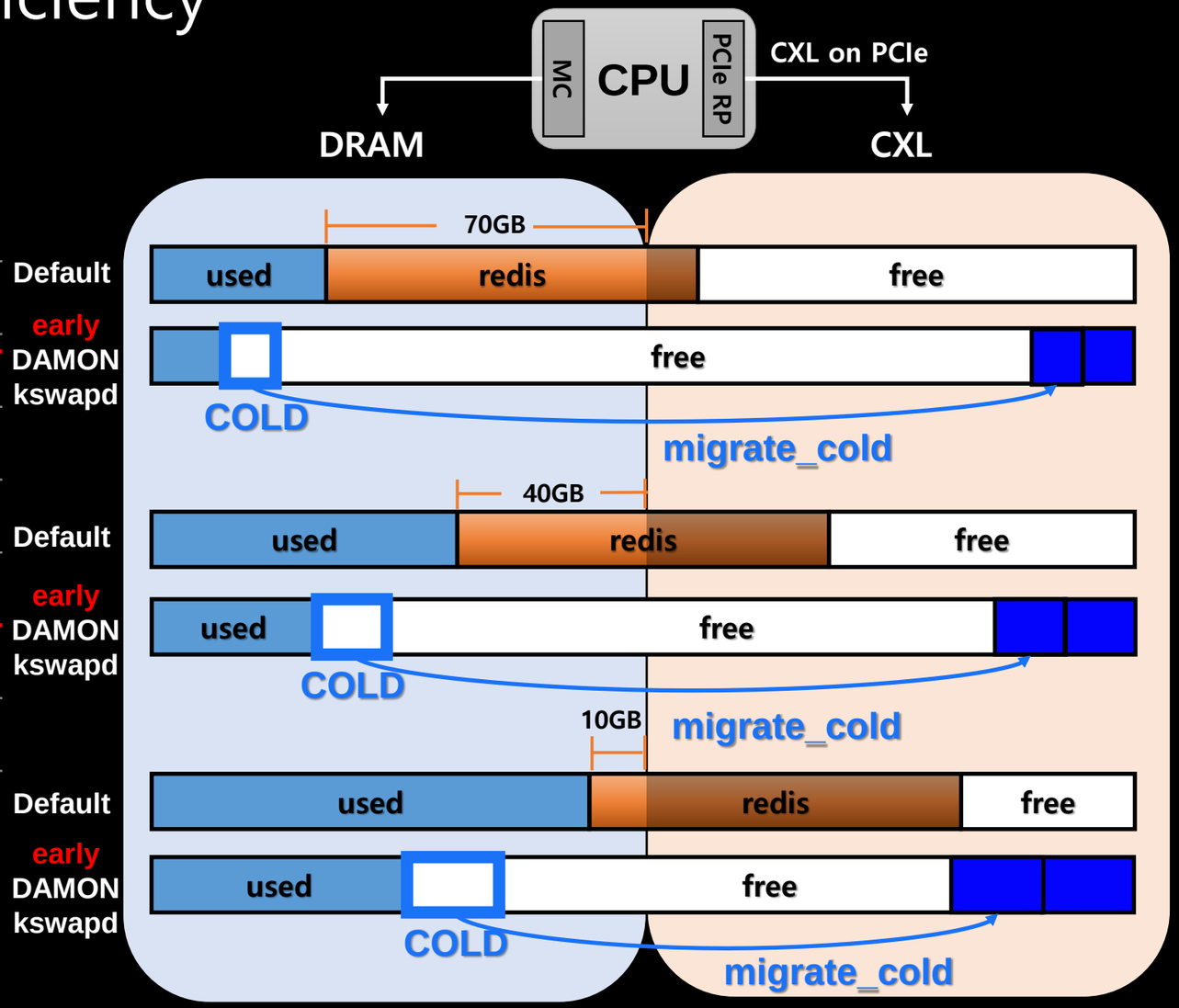
1. DRAM is partially used by non-redis cold data.
2. Partial redis data is allocated on CXL memory.
   (due to insufficient space on DRAM)

<HMSDK>

1. Demote cold data from DRAM to CXL memory.
2. Keep demote cold data

HMSDK: Enhancing CXL Memory Efficiency





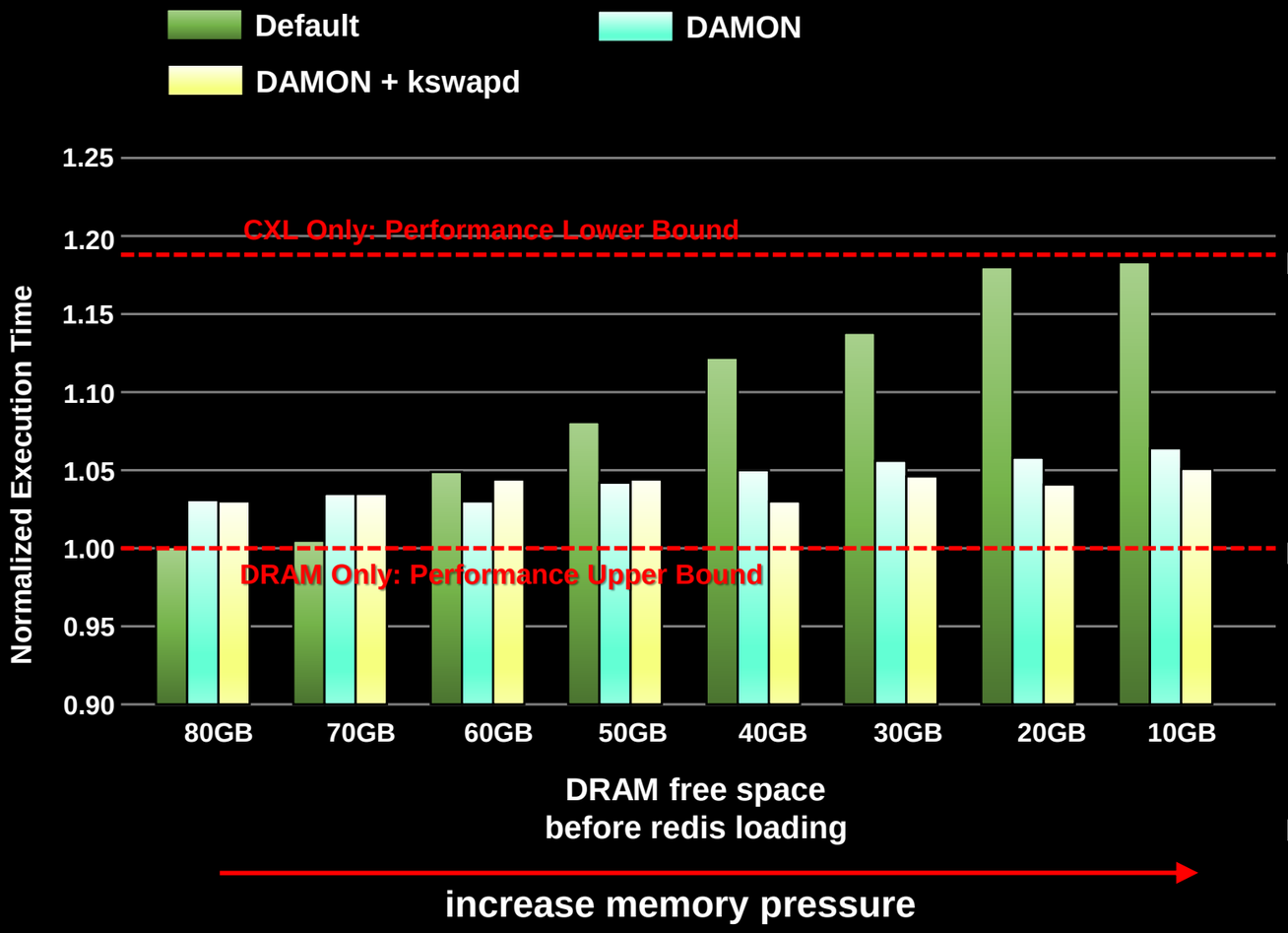
<Default>

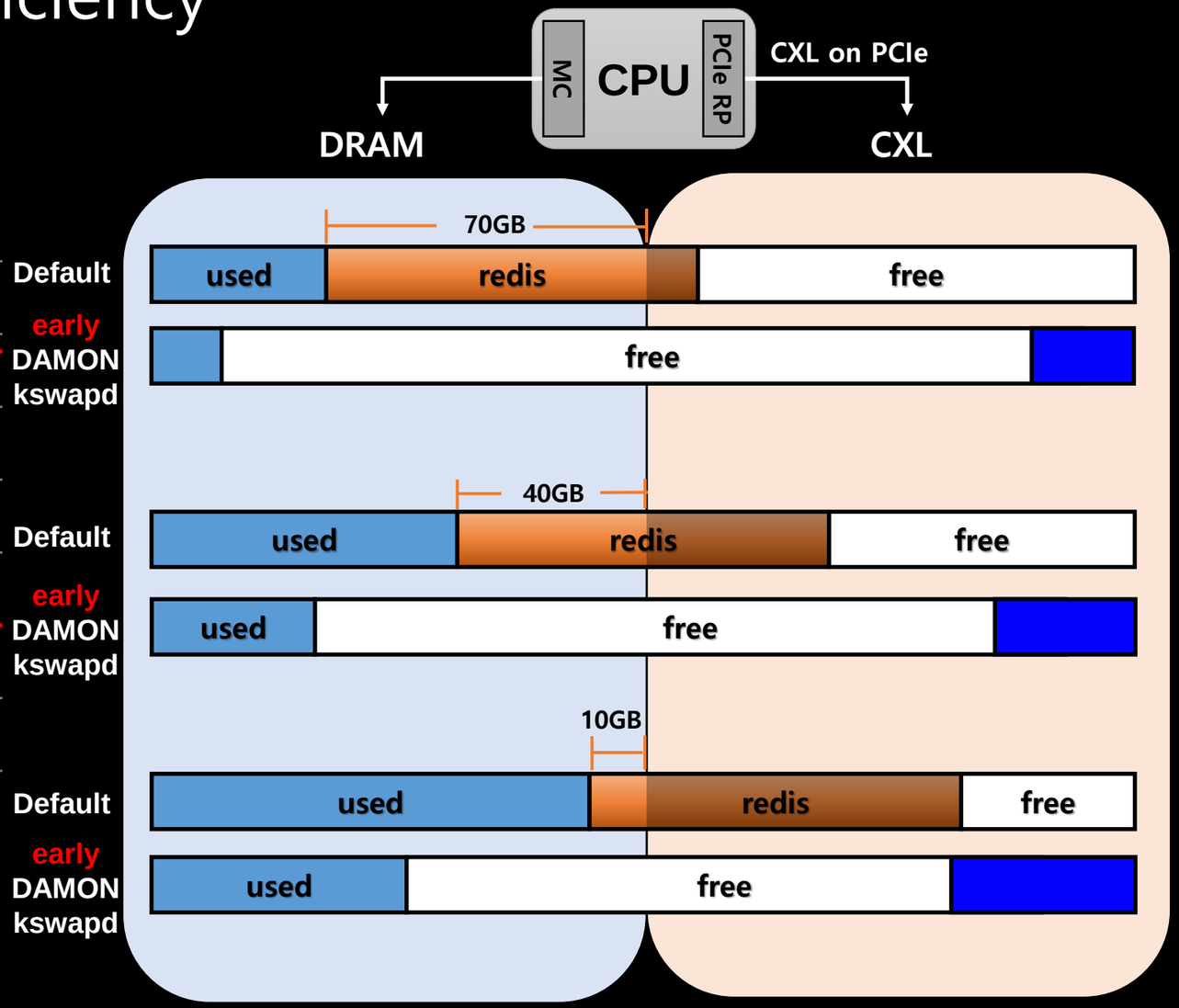
1. DRAM is partially used by non-redis cold data.
2. Partial redis data is allocated on CXL memory.
   (due to insufficient space on DRAM)

<HMSDK>

1. Demote cold data from DRAM to CXL memory.
2. Keep demote cold data

HMSDK: Enhancing CXL Memory Efficiency





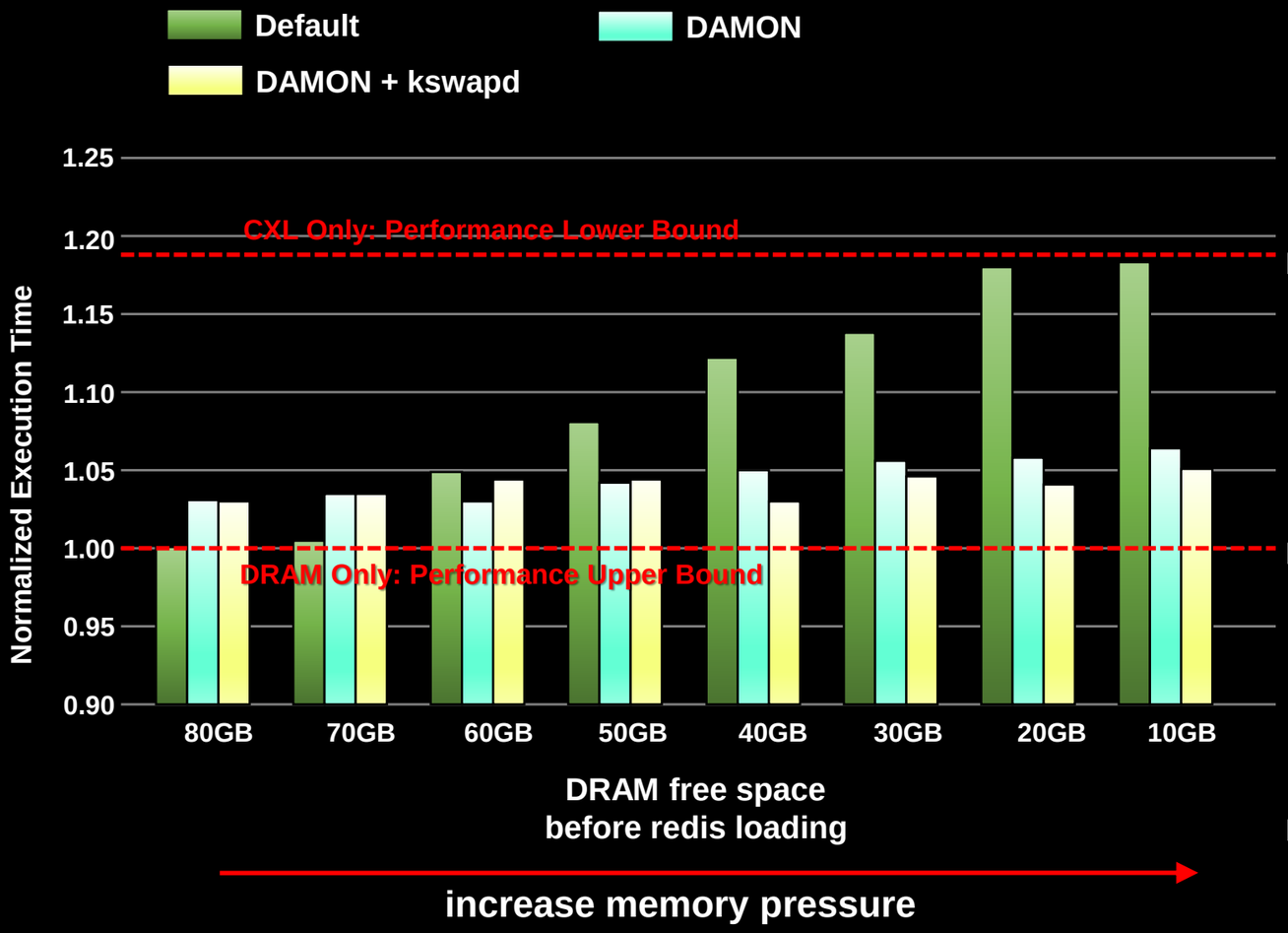
<Default>

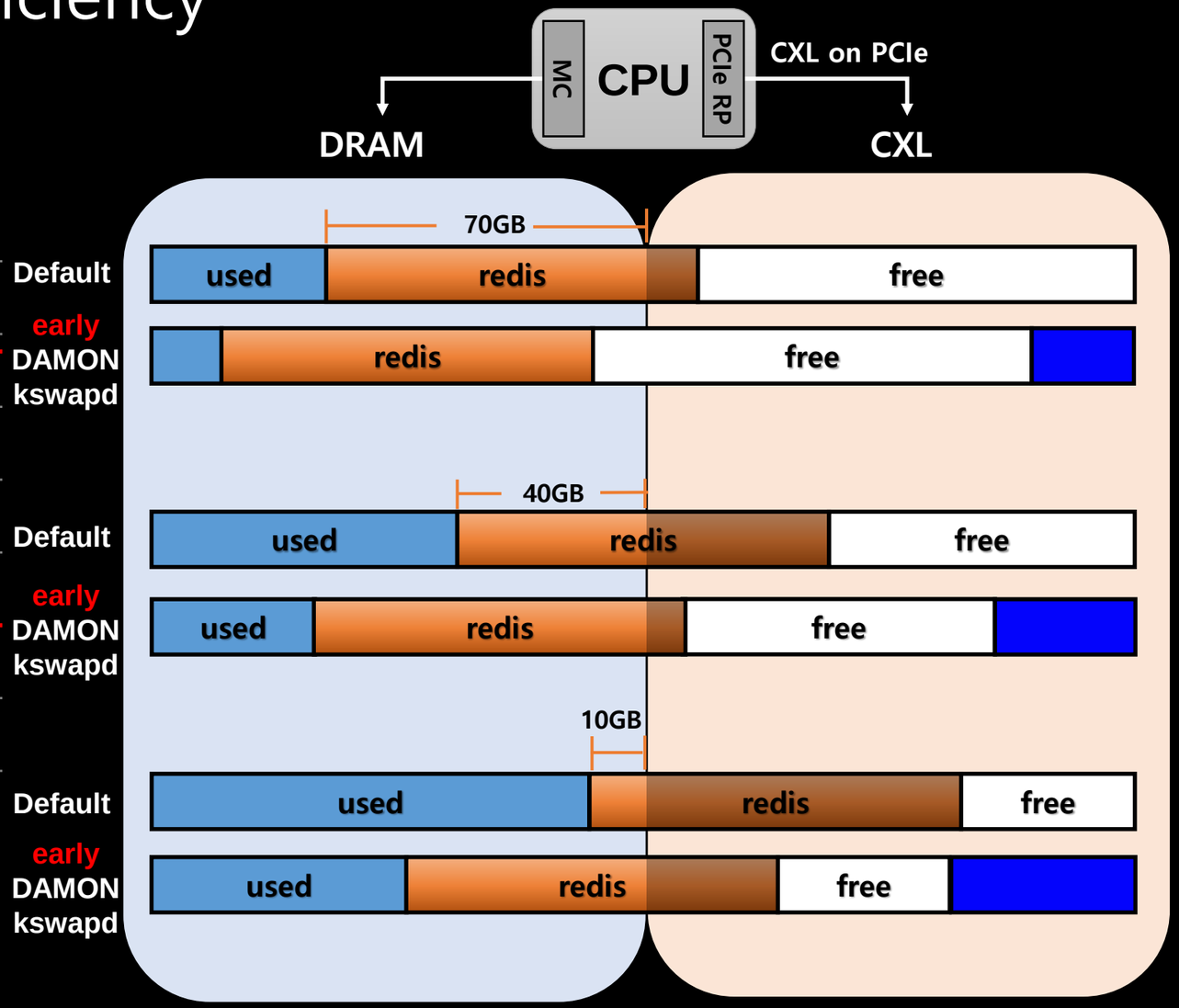
1. DRAM is partially used by non-redis cold data.
2. Partial redis data is allocated on CXL memory.
   (due to insufficient space on DRAM)

<HMSDK>

1. Demote cold data from DRAM to CXL memory.
2. Keep demote cold data

HMSDK: Enhancing CXL Memory Efficiency





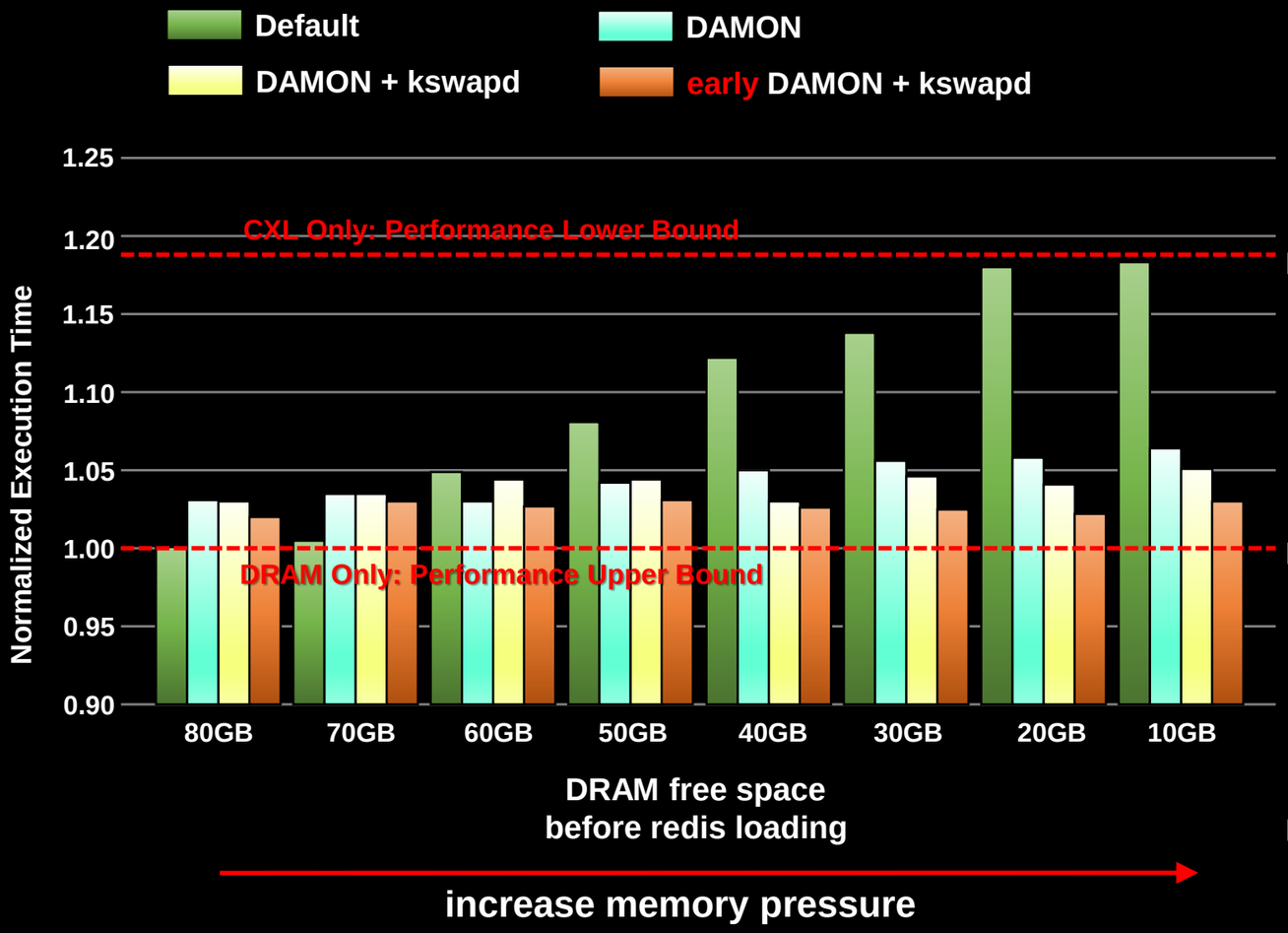
<Default>

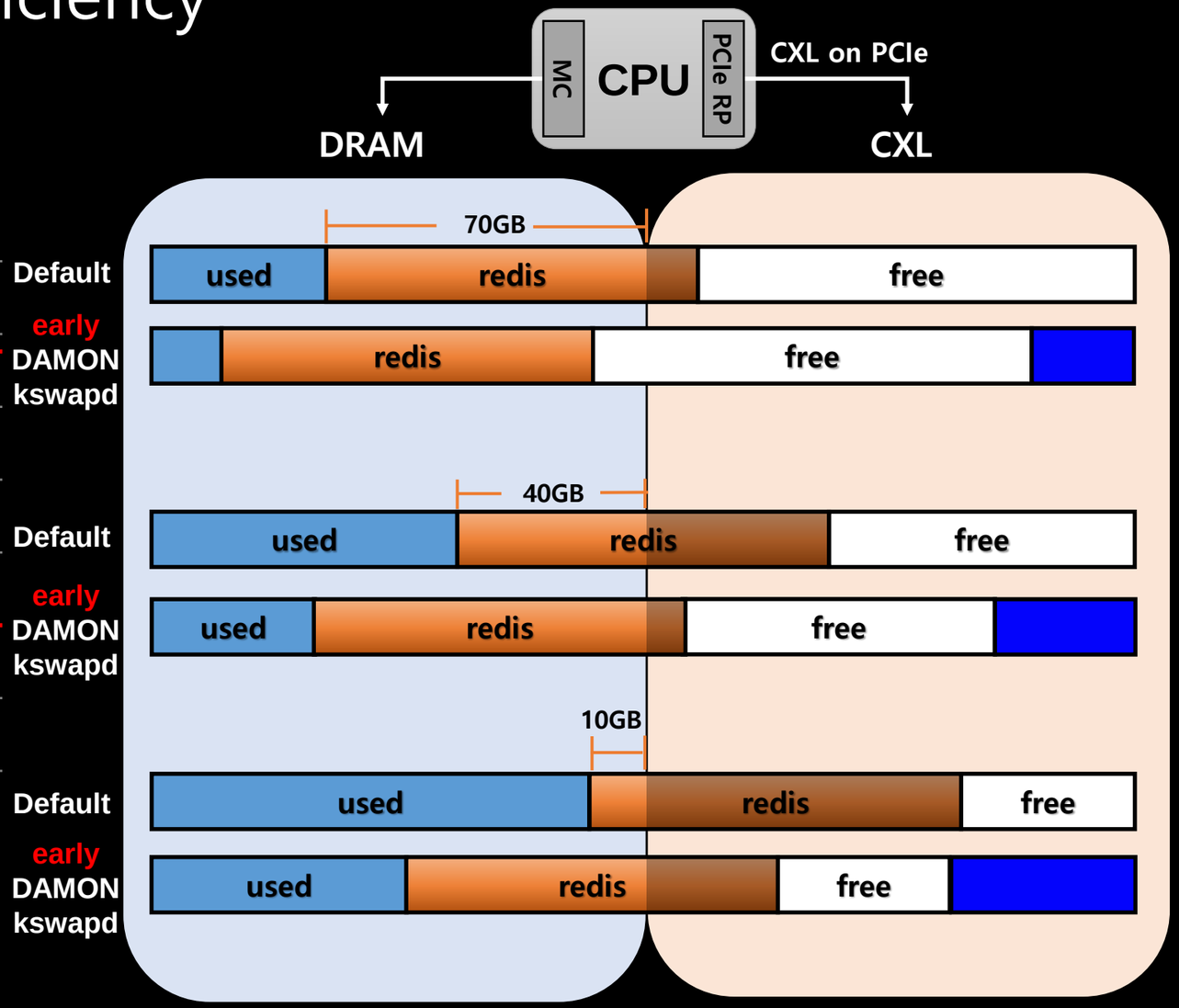
1. DRAM is partially used by non-redis cold data.
2. Partial redis data is allocated on CXL memory.
   (due to insufficient space on DRAM)

<HMSDK>

1. Demote cold data from DRAM to CXL memory.
2. Keep demote cold data
3. More redis data can be allocated on DRAM.

HMSDK: Enhancing CXL Memory Efficiency

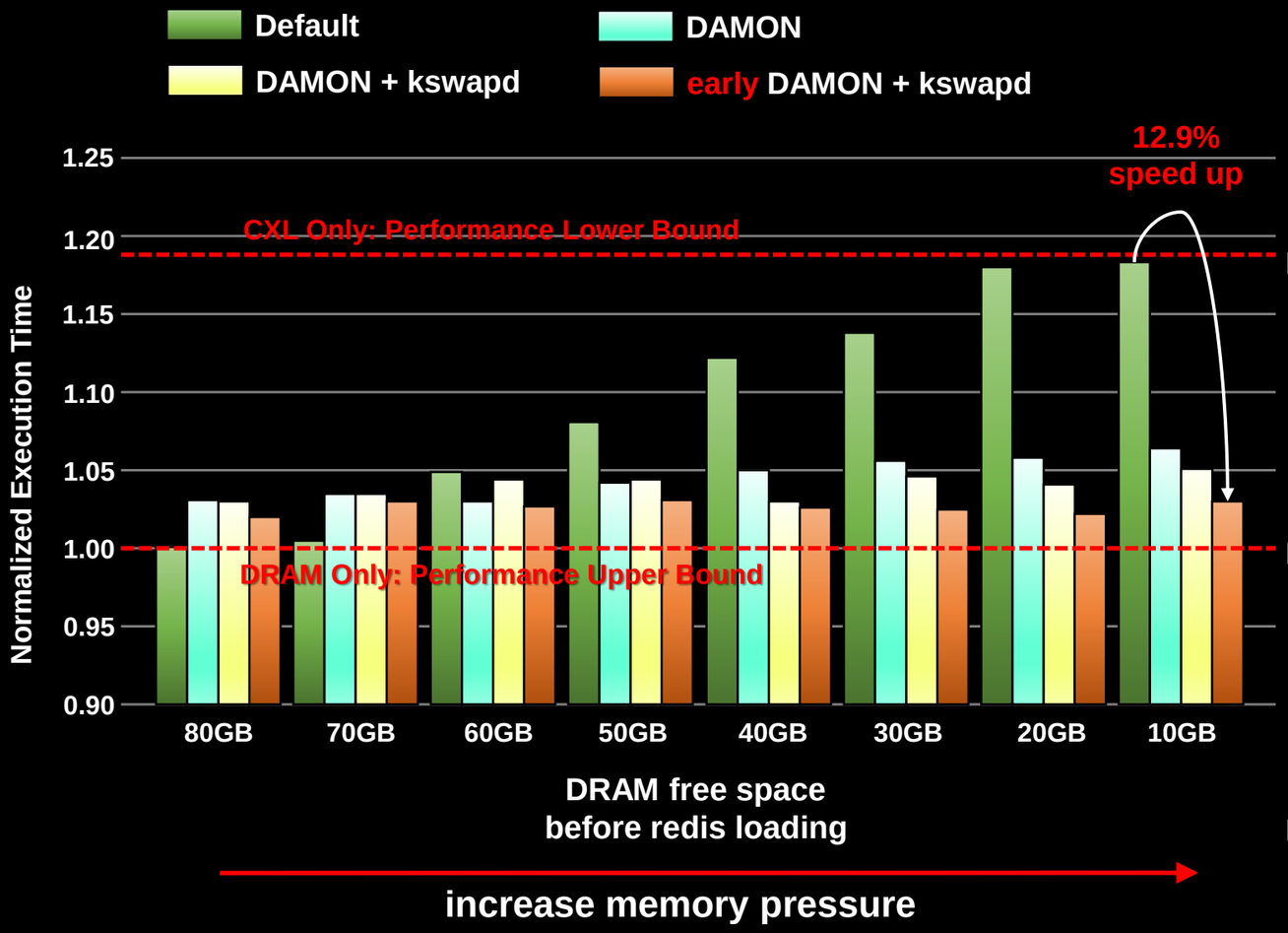


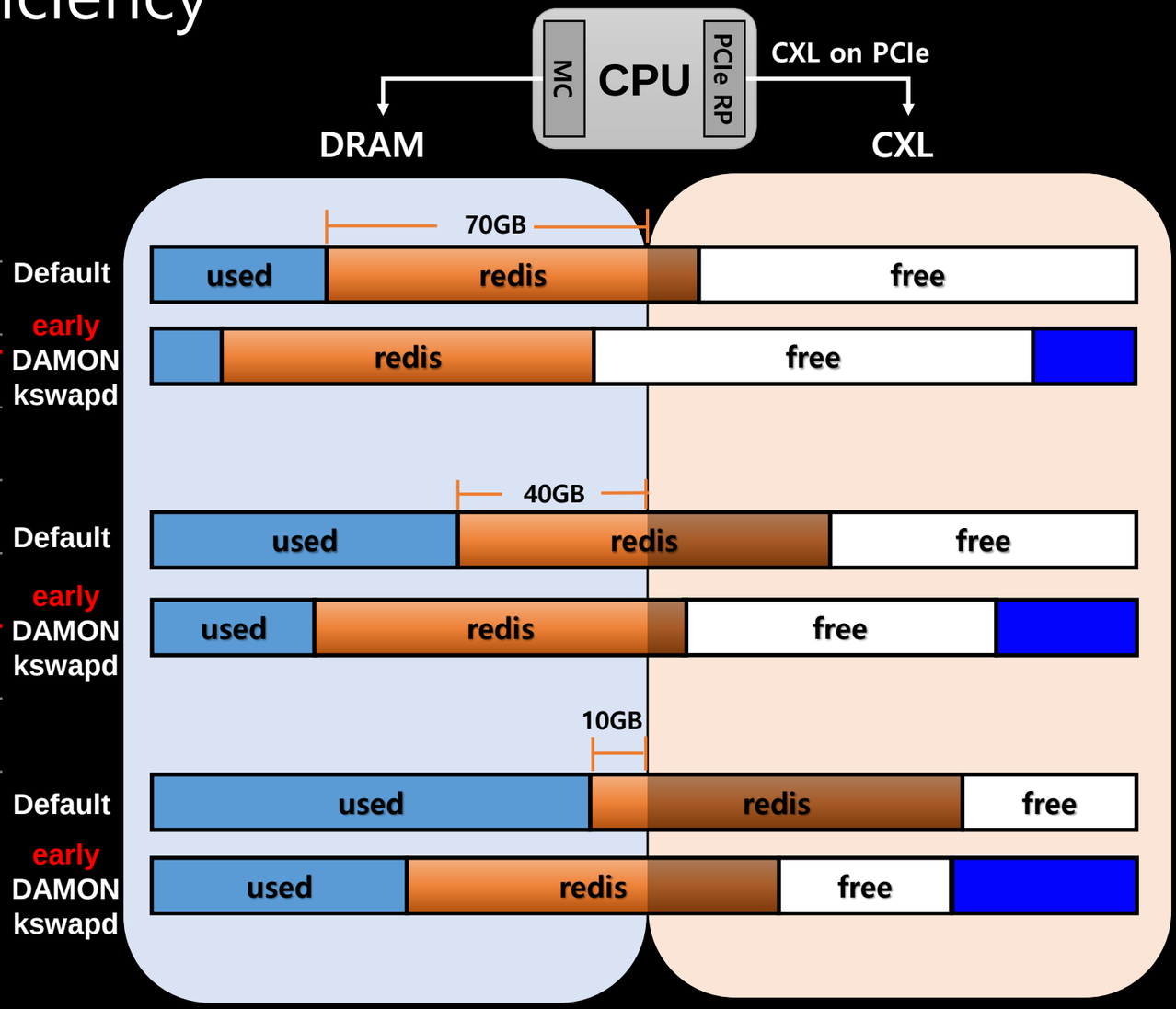


<Default>
1. DRAM is partially used by non-redis cold data.
2. Partial redis data is allocated on CXL memory.
   (due to insufficient space on DRAM)

<HMSDK>
1. Demote cold data from DRAM to CXL memory.
2. Keep demote cold data
3. More redis data can be allocated on DRAM.

HMSDK: Enhancing CXL Memory Efficiency





<Default>

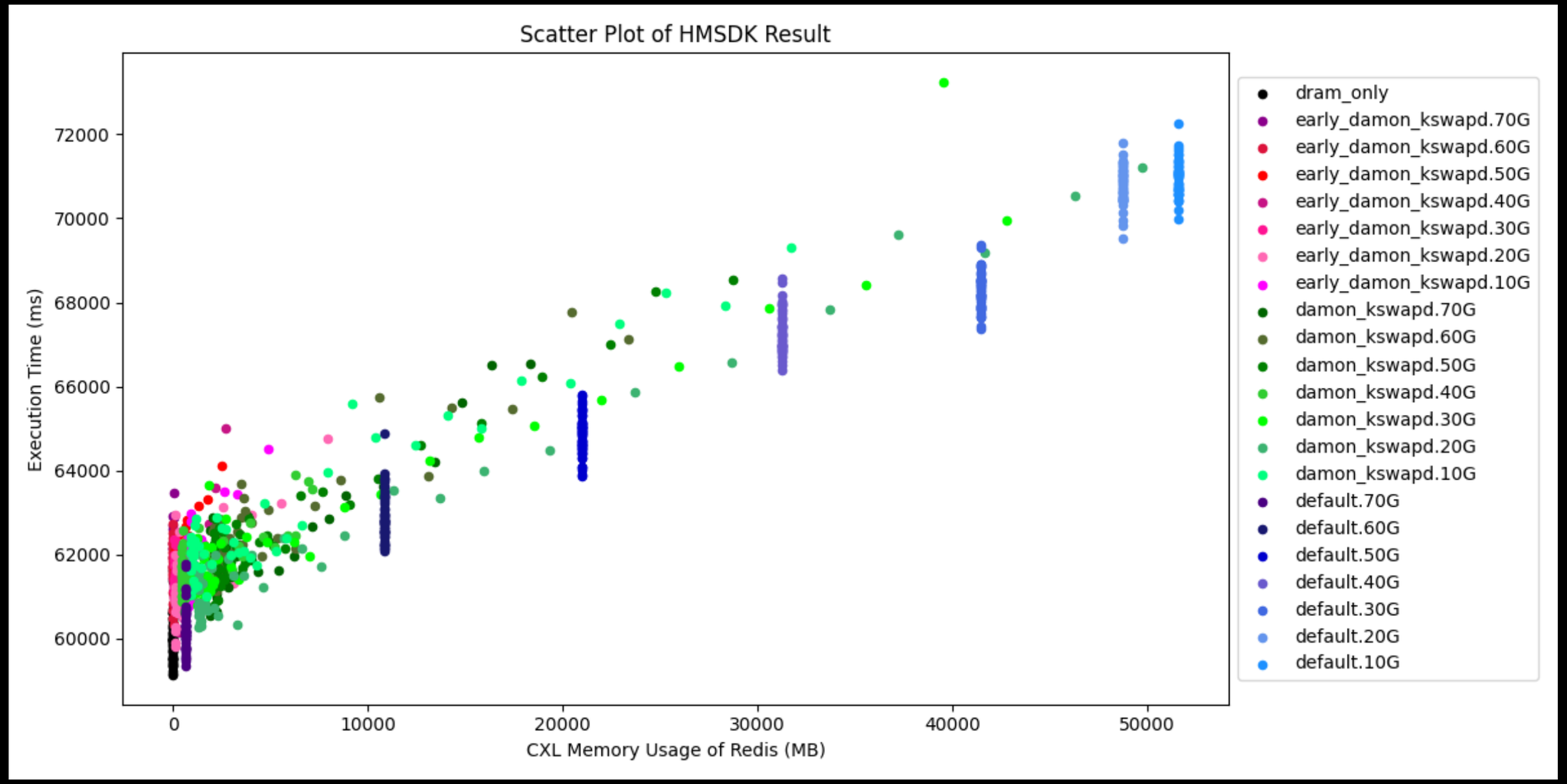
1. DRAM is partially used by non-redis cold data.
2. Partial redis data is allocated on CXL memory.
   (due to insufficient space on DRAM)

<HMSDK>

1. Demote cold data from DRAM to CXL memory.
2. Keep demote cold data
3. More redis data can be allocated on DRAM.

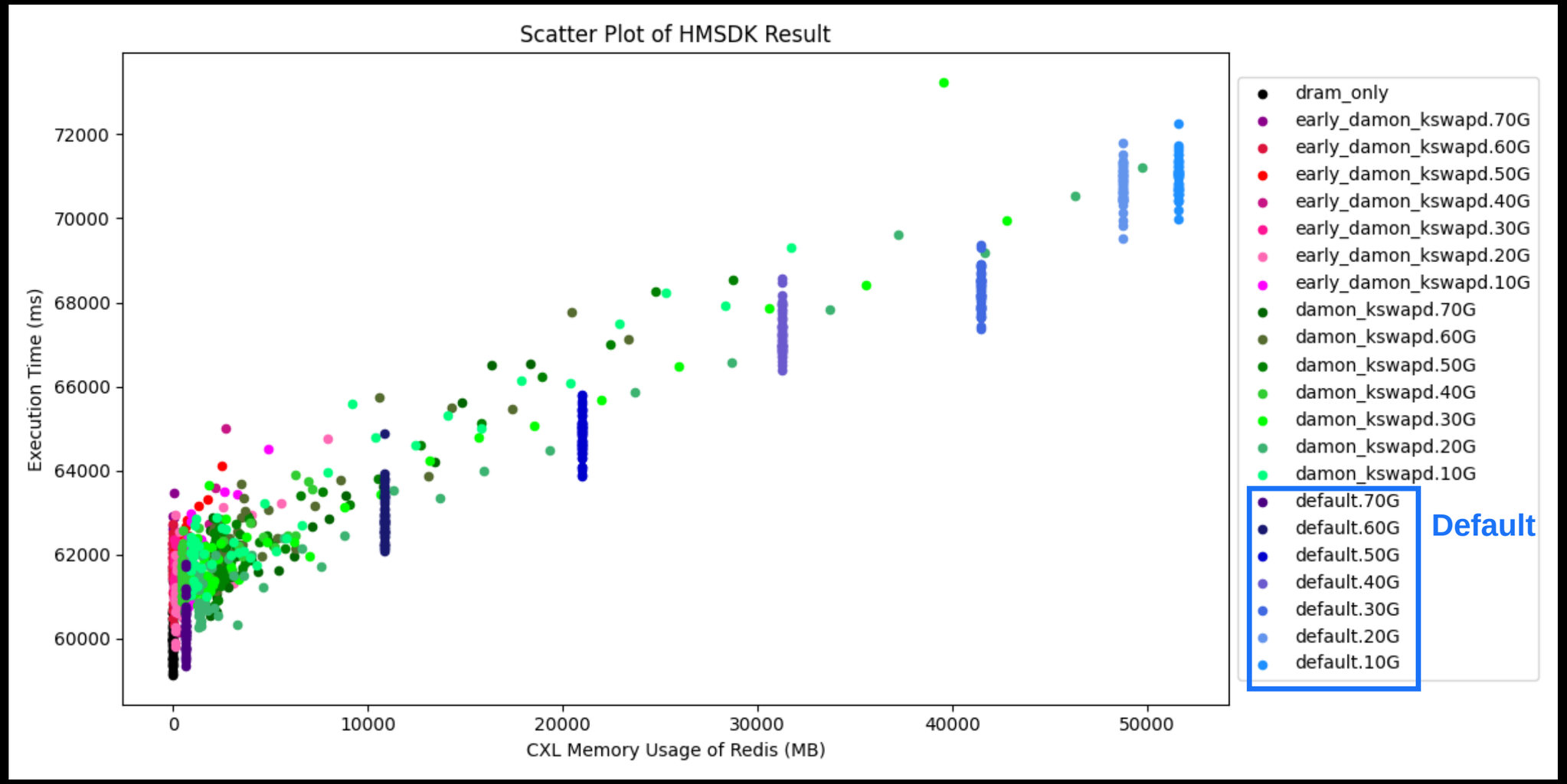
Evaluation (Redis + YCSB zipfian distribution)

- X-axis: CXL Memory Usage of Redis (MB)
- Y-axis: Execution Time (ms)



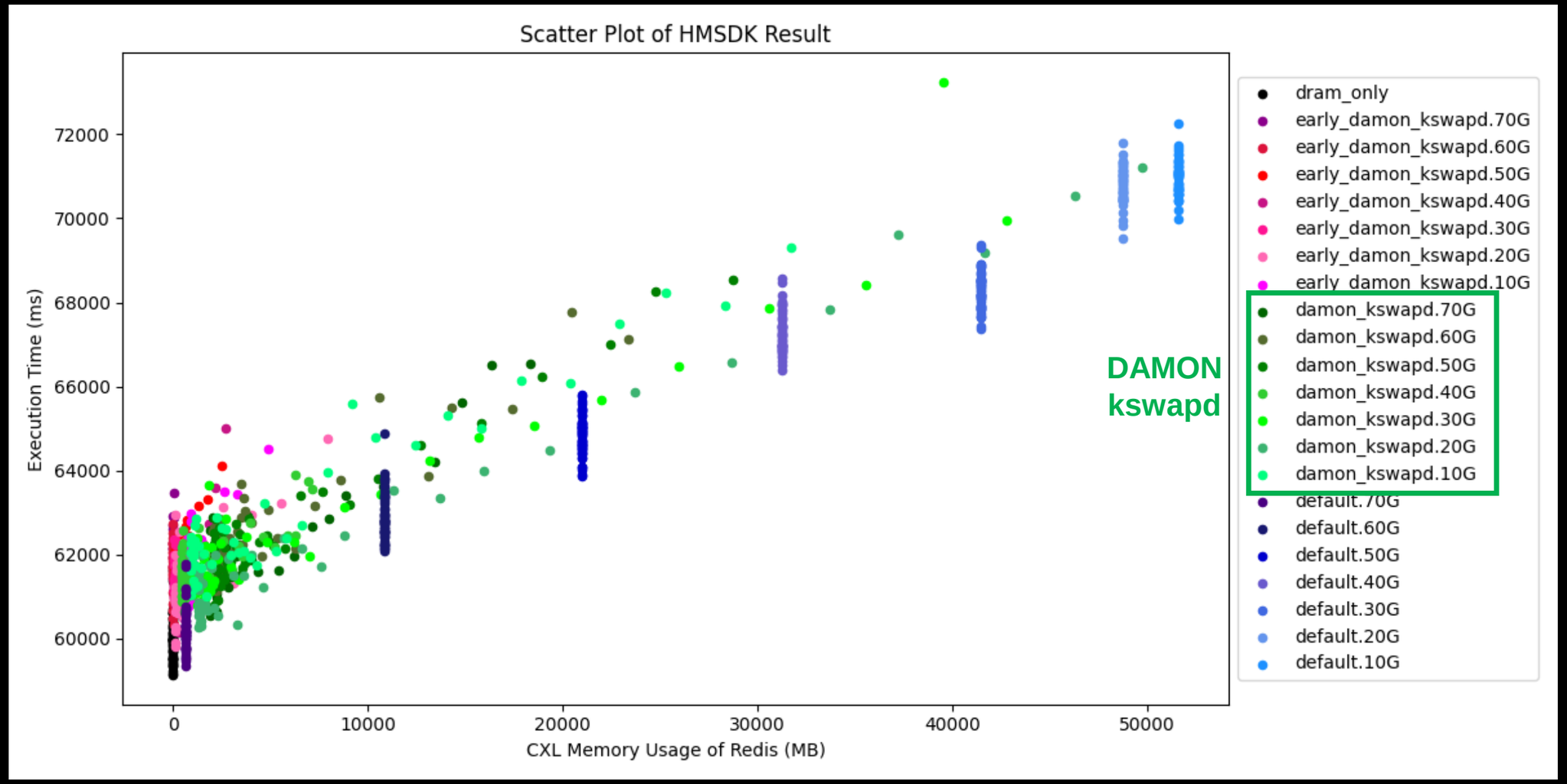
Evaluation (Redis + YCSB zipfian distribution)

- **As remaining DRAM free space insufficient, use more CXL memory**
  - The execution time of "Default" linearly grows while DAMON minimizes slowdown.



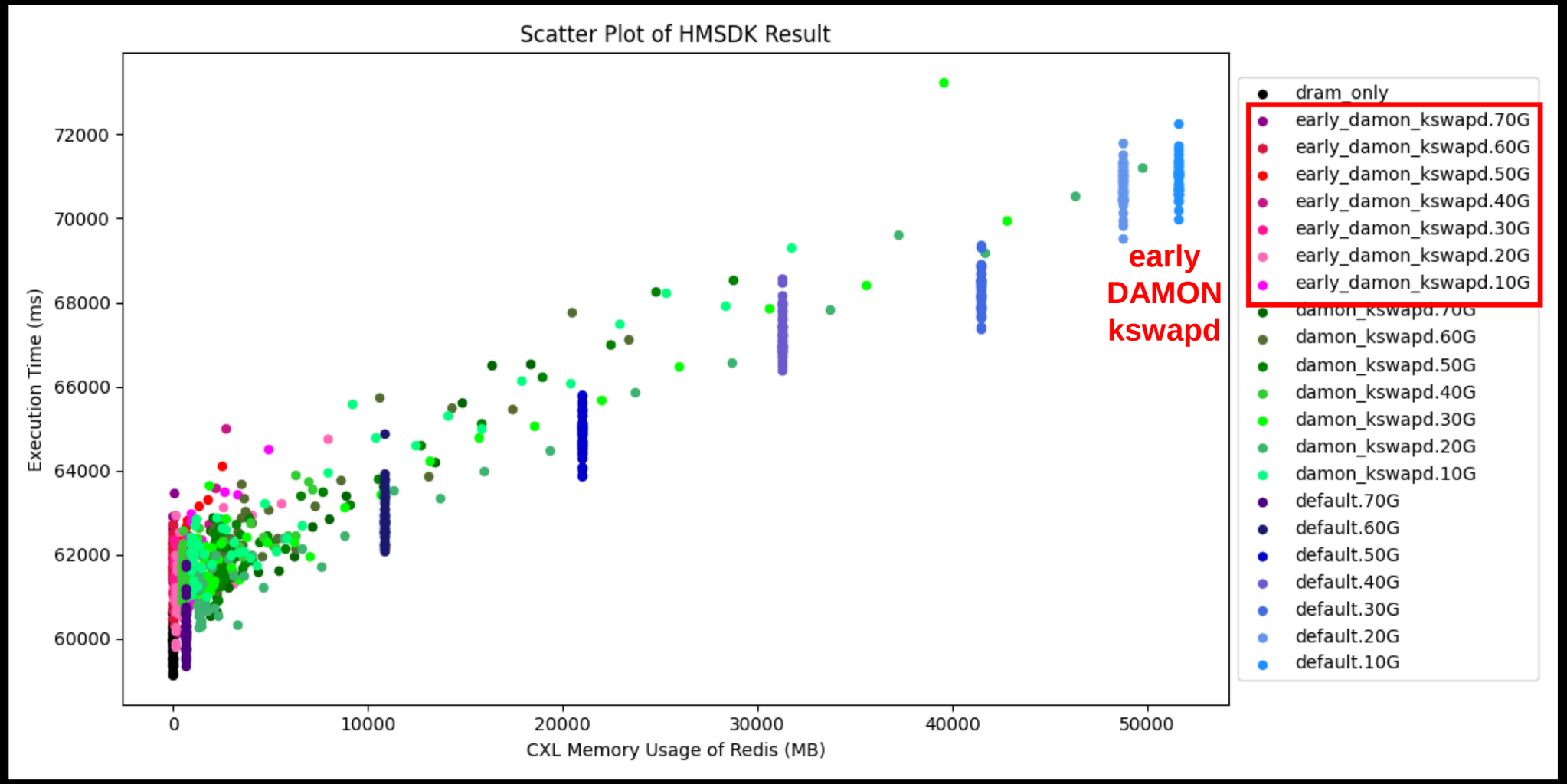
Evaluation (Redis + YCSB zipfian distribution)

- **As remaining DRAM free space insufficient, use more CXL memory**
  - The execution time of "Default" linearly grows while DAMON minimizes slowdown.



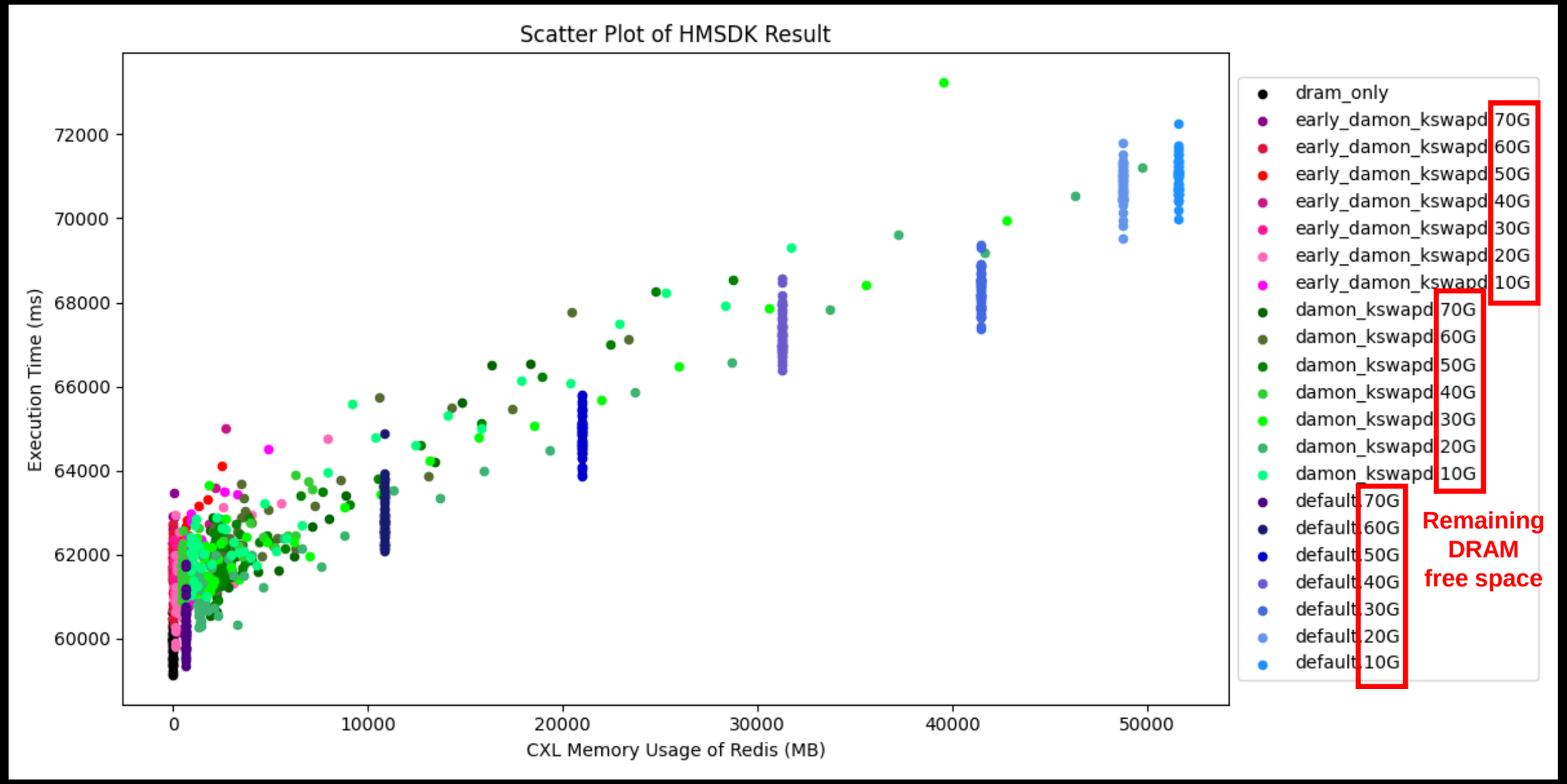
Evaluation (Redis + YCSB zipfian distribution)

- **As remaining DRAM free space insufficient, use more CXL memory**
  - The execution time of "Default" linearly grows while DAMON minimizes slowdown.



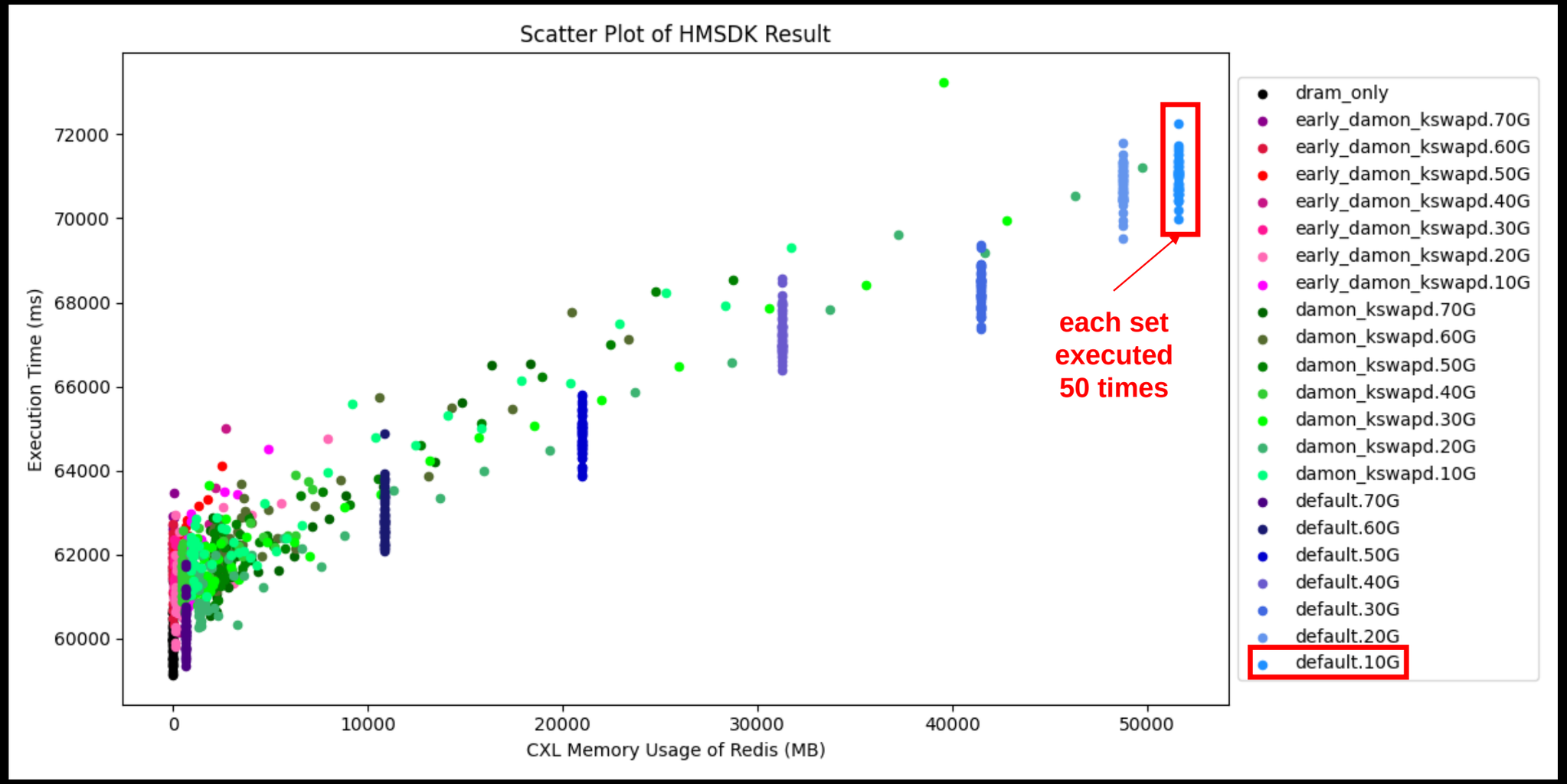
Evaluation (Redis + YCSB zipfian distribution)

- **As remaining DRAM free space insufficient, use more CXL memory**
  - The execution time of "Default" linearly grows while DAMON minimizes slowdown.



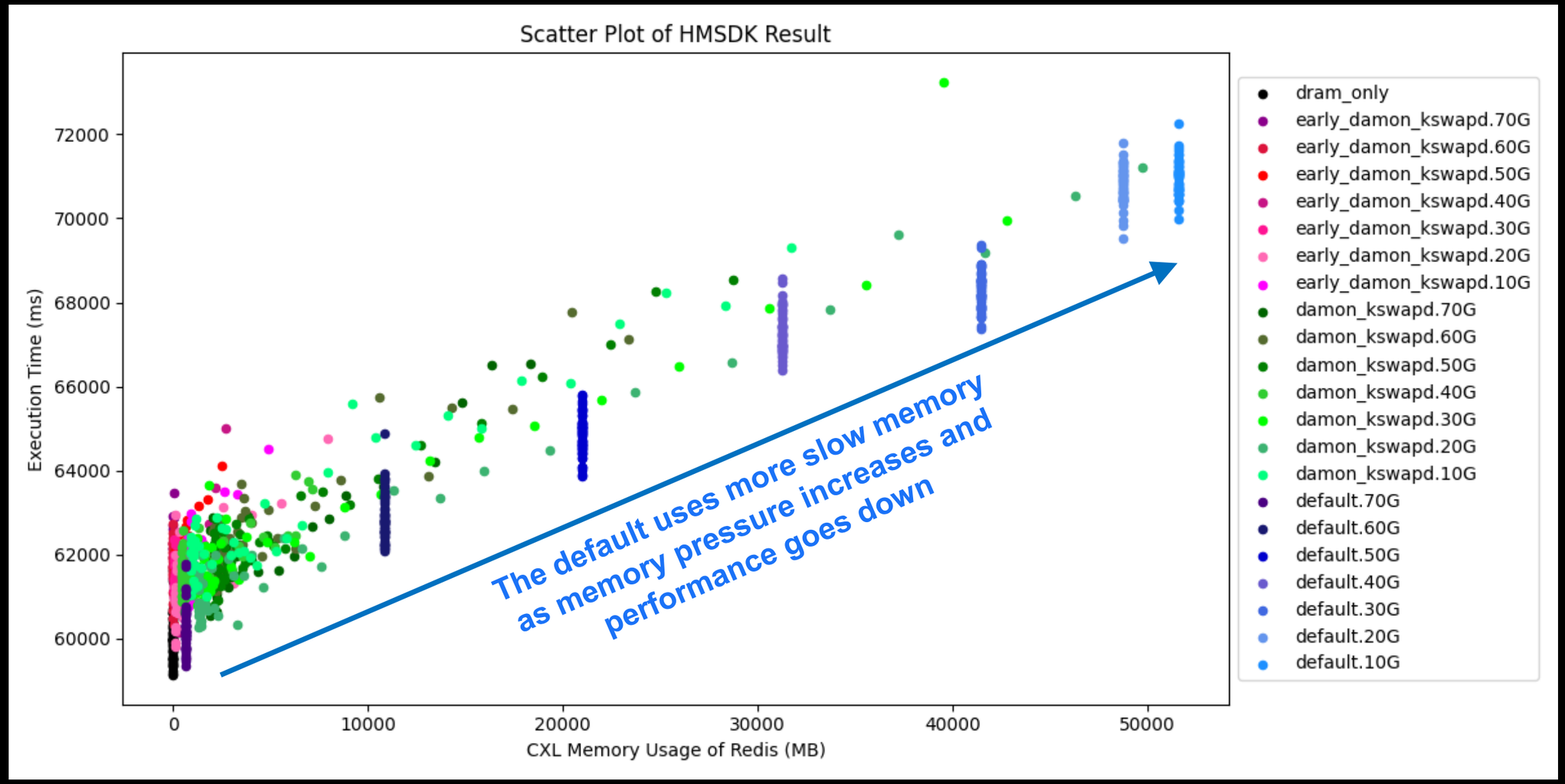
Evaluation (Redis + YCSB zipfian distribution)

- **As remaining DRAM free space insufficient, use more CXL memory**
  - The execution time of "Default" linearly grows while DAMON minimizes slowdown.



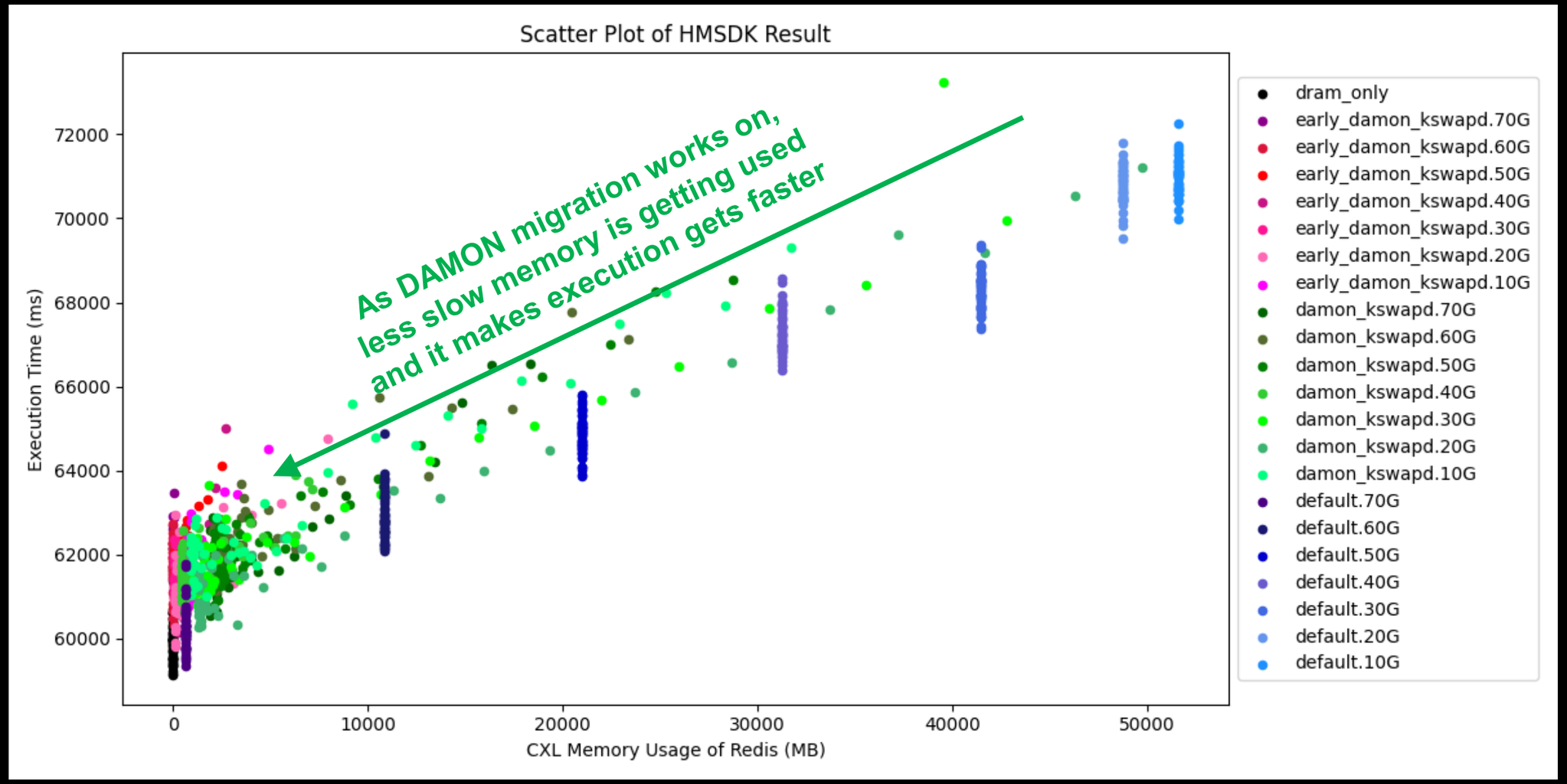
Evaluation (Redis + YCSB zipfian distribution)

- **As remaining DRAM free space insufficient, use more CXL memory**
  - The execution time of "Default" linearly grows while DAMON minimizes slowdown.



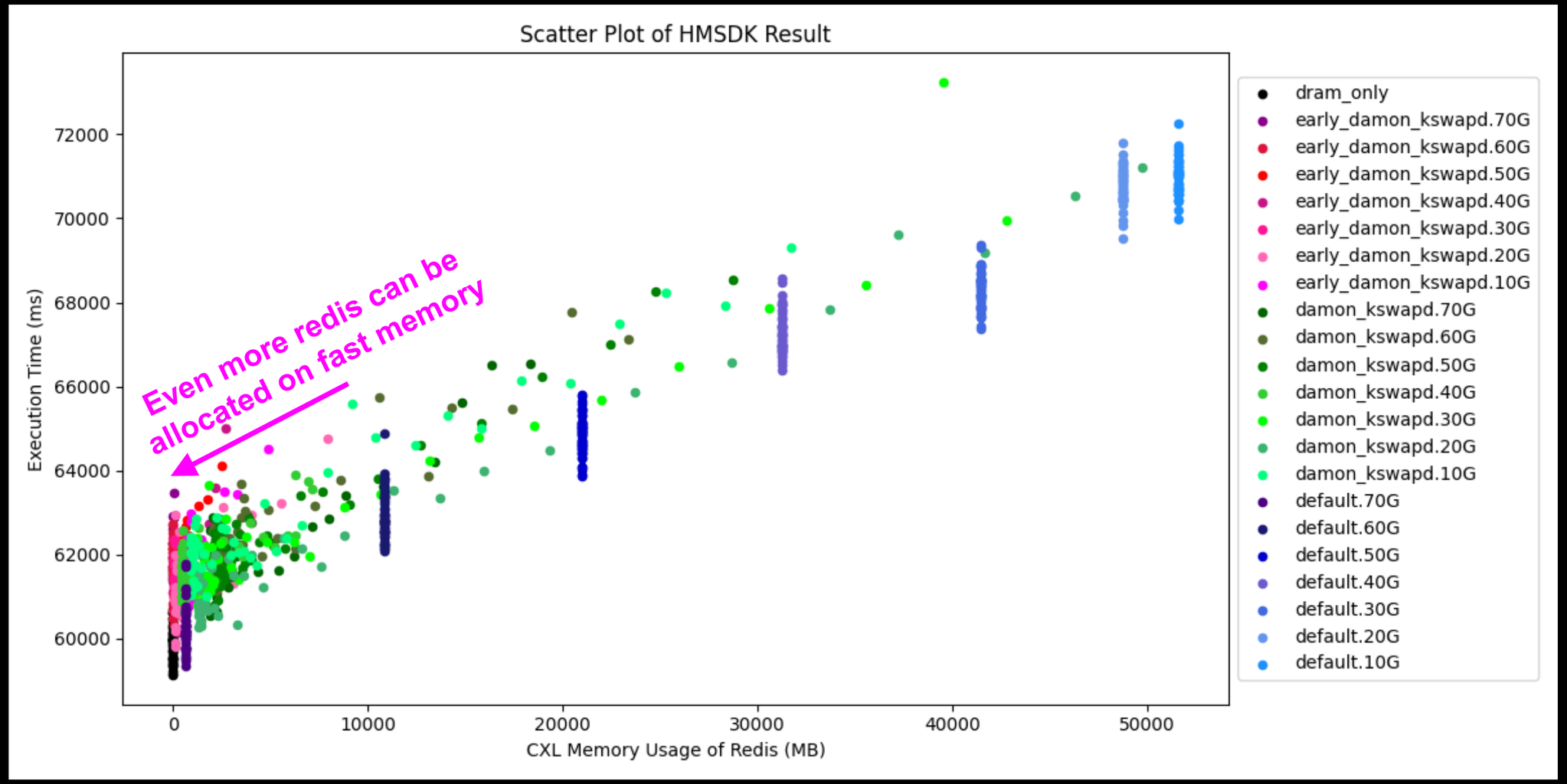
Evaluation (Redis + YCSB zipfian distribution)

- **As remaining DRAM free space insufficient, use more CXL memory**
  - The execution time of "Default" linearly grows while DAMON minimizes slowdown.



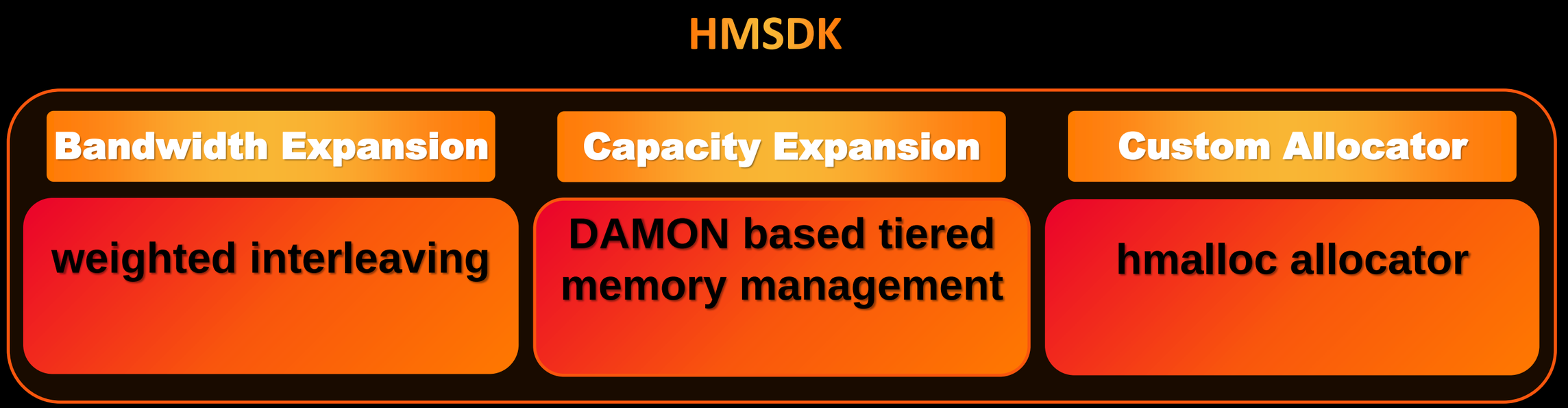
Evaluation (Redis + YCSB zipfian distribution)

- **As remaining DRAM free space insufficient, use more CXL memory**
  - The execution time of "Default" linearly grows while DAMON minimizes slowdown.



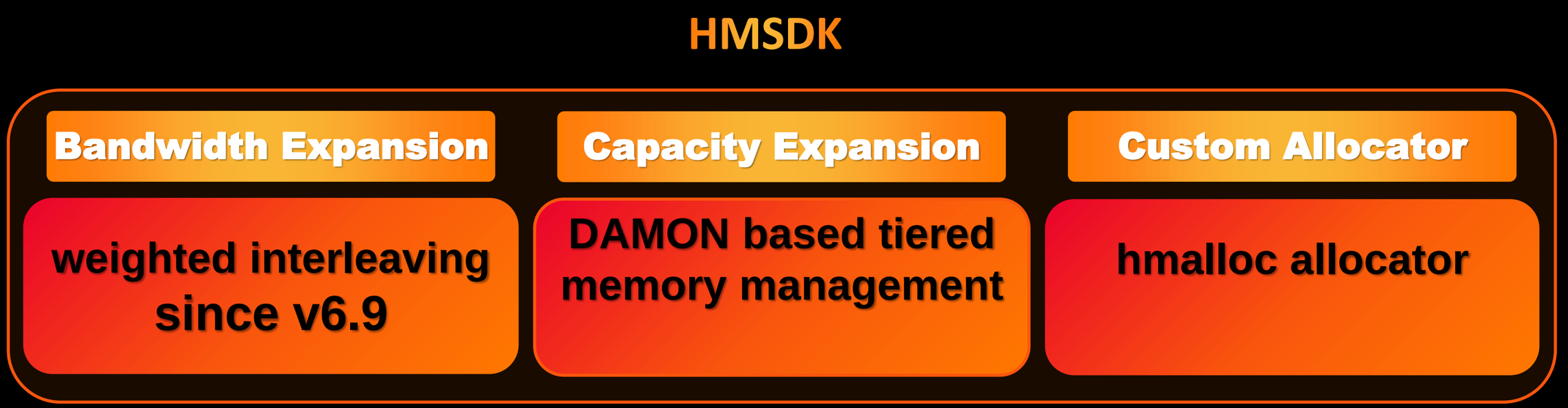
HMSDK 3.0 release

- **HMSDK 3.0 was released today based on Linux v6.11!**
  - Fully aligned with open source community
  - No more local patches included



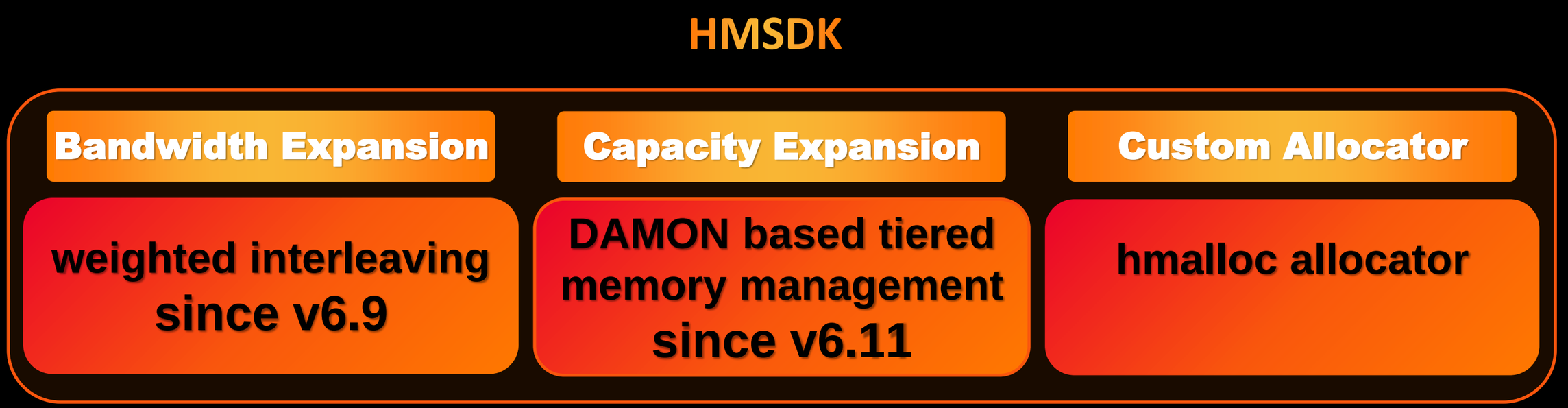
HMSDK 3.0 release

- **HMSDK 3.0 was released today based on Linux v6.11!**
  - Fully aligned with open source community
  - No more local patches included



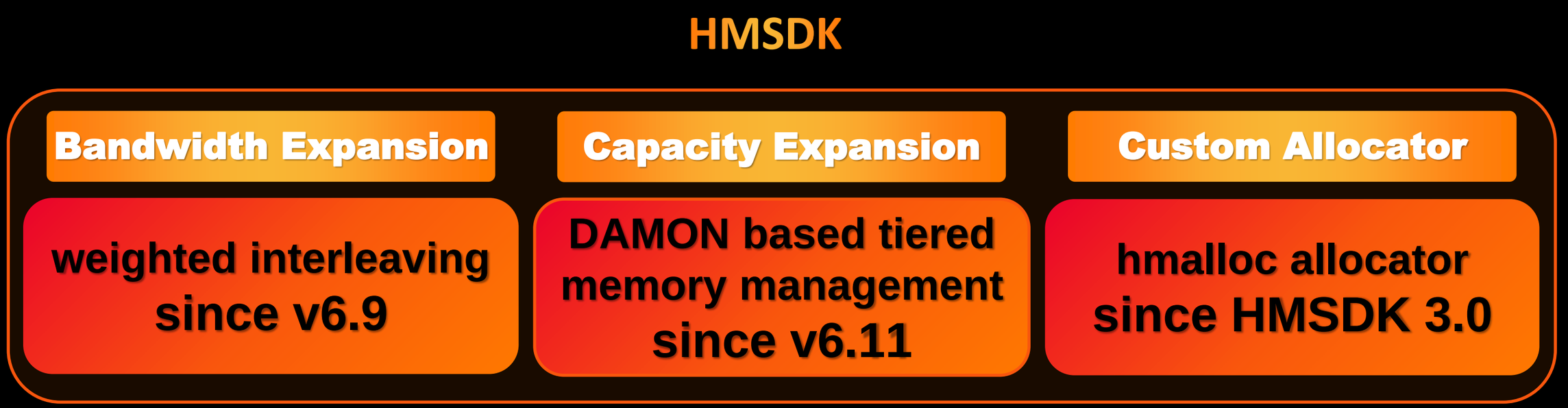
HMSDK 3.0 release

- **HMSDK 3.0 was released today based on Linux v6.11!**
  - Fully aligned with open source community
  - No more local patches included



HMSDK 3.0 release

- **HMSDK 3.0 was released today based on Linux v6.11!**
  - Fully aligned with open source community
  - No more local patches included



Thanks!

GitHub: https://github.com/skhynix/hmsdk

Document: https://github.com/skhynix/hmsdk/wiki

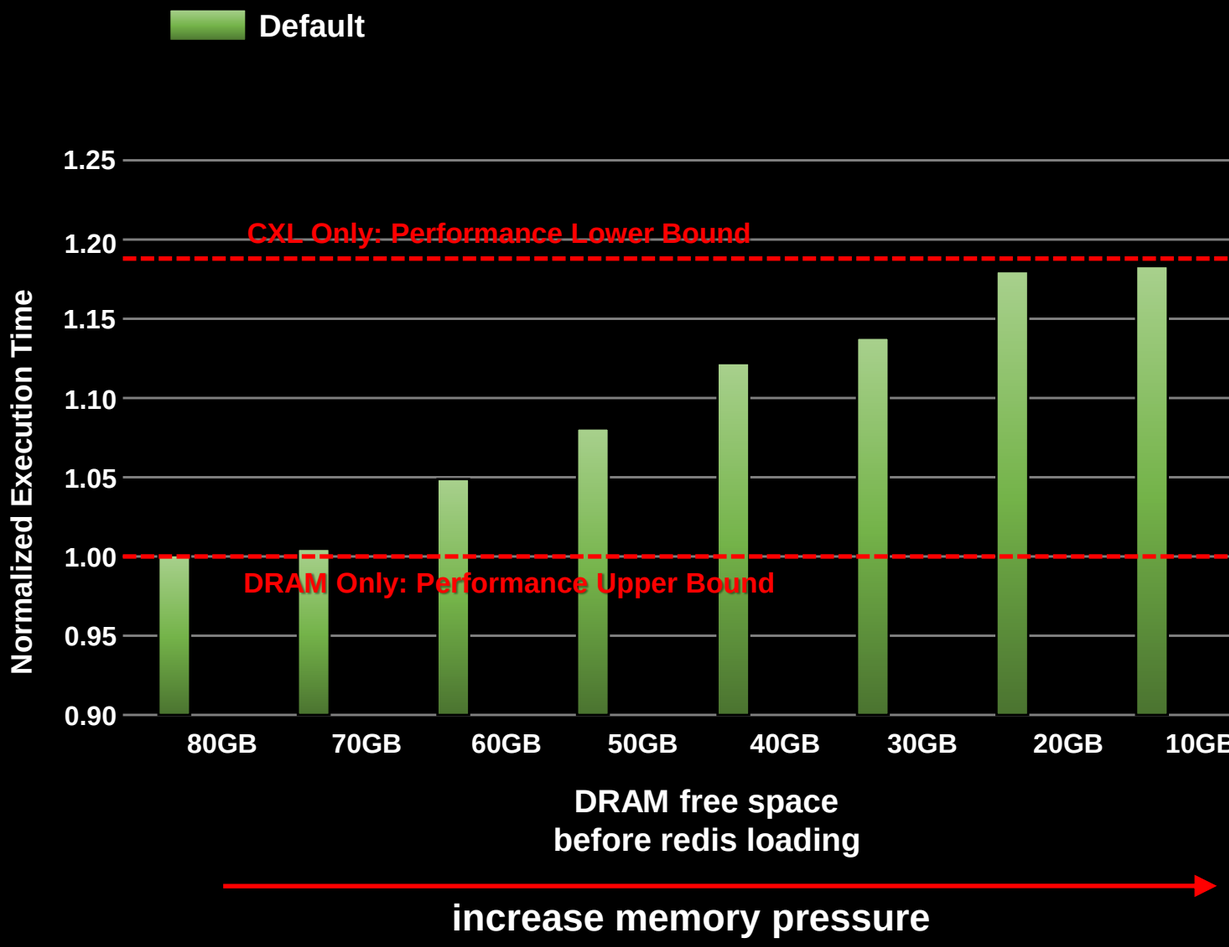
Appendix

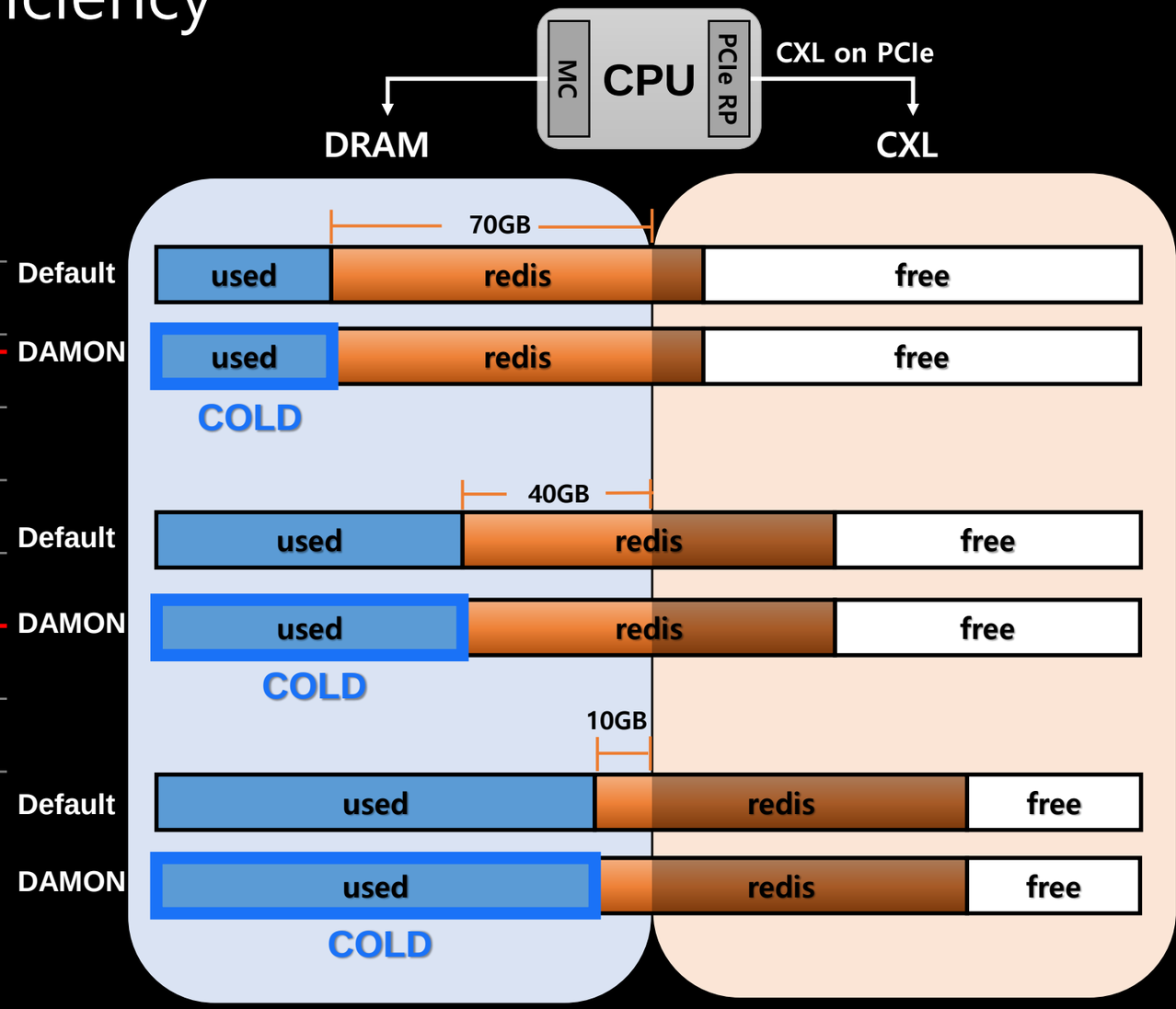
HMSDK Capacity Expansion User Guide

- **HMSDK Capacity Expansion User Guide provides a way to generate the config.**
  - https://github.com/skhynix/hmsdk/wiki/Capacity-Expansion#user-guide

```
# The -d/--demote and -p/--promote options can be used multiple times.
# The SRC and DEST are migration source node id and destination node id.
$ sudo ./tools/gen_migpol.py -d SRC DEST -p SRC DEST -o hmsdk.yaml

# Enable demotion to slow tier.
# This prevents from swapping out from fast tier.
$ echo true | sudo tee /sys/kernel/mm/numa/demotion_enabled

# Make sure cgroup2 is mounted under /sys/fs/cgroup,
# then create "hmsdk" directory below.
$ sudo mkdir -p /sys/fs/cgroup/hmsdk

# Start HMSDK Capacity Expansion based on hmsdk.yaml.
$ sudo ./damo/damo start hmsdk.yaml

# Stop HMSDK Capacity Expansion
$ sudo ./damo/damo stop
```

HMSDK: Enhancing CXL Memory Efficiency





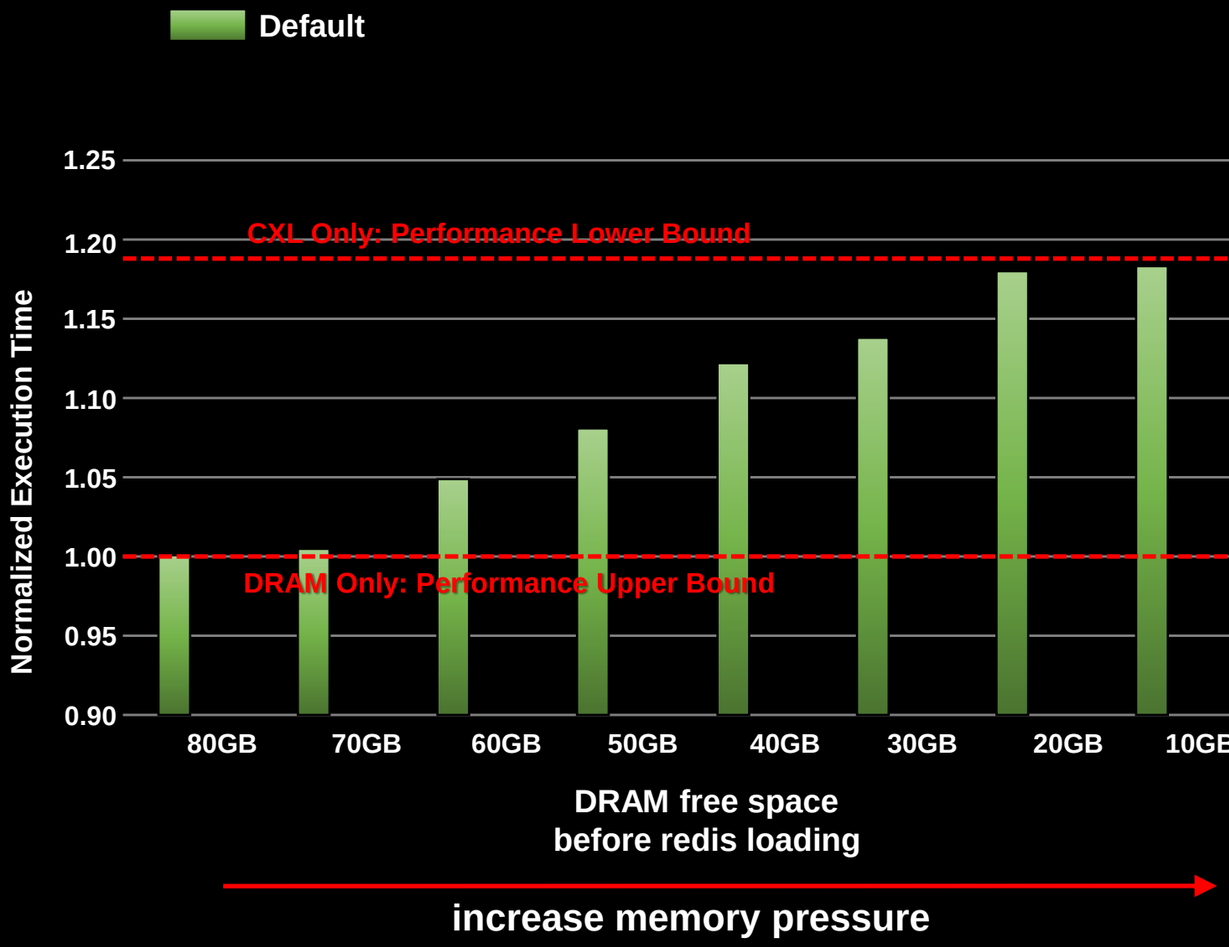
<Default>

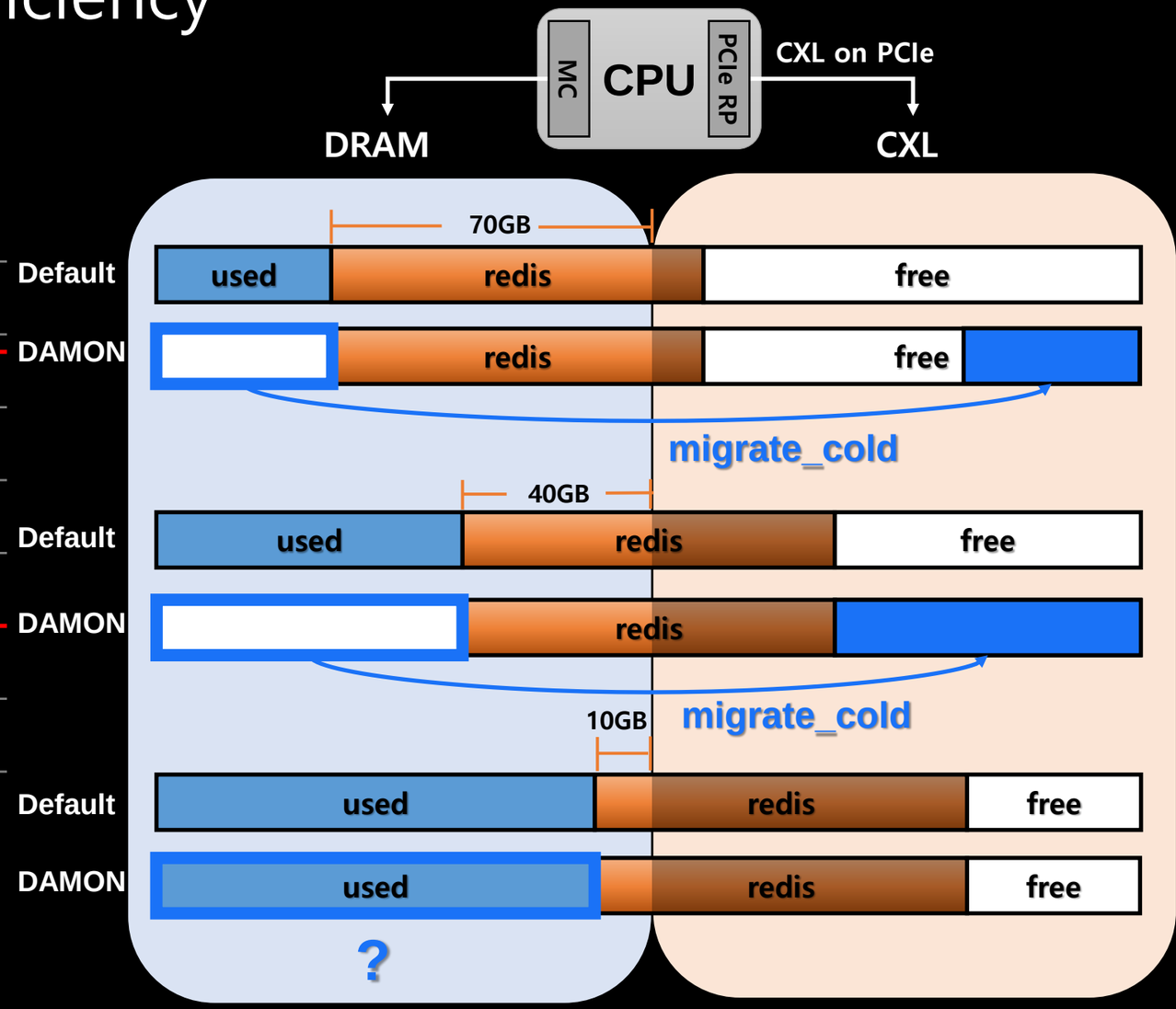
1. DRAM is partially used by non-redis cold data.
2. Partial redis data is allocated on CXL memory.
   (due to insufficient space on DRAM)

<HMSDK>

1. Demote cold data from DRAM to CXL memory.
2. Promote more redis data to fast DRAM.
   (while keeping cold data on CXL memory)

HMSDK: Enhancing CXL Memory Efficiency





<Default>
1. DRAM is partially used by non-redis cold data.
2. Partial redis data is allocated on CXL memory.
   (due to insufficient space on DRAM)

<HMSDK>
1. Demote cold data from DRAM to CXL memory.
2. Promote more redis data to fast DRAM.
   (while keeping cold data on CXL memory)

```
"targets": [ { <DRAM physical addr range> } ],
"schemes": [
    {
        "action": "migrate_cold",
        "access_pattern": {
            "sz_bytes": {"min": "4.000 KiB", "max": "max"},
            "nr_accesses": {"min": "0 %", "max": "0 %"},
            "age": {"min": "30 s", "max": "max"}
        },
        "quotas": {
            "time_ms": "1 s",
            "sz_bytes": "50 GiB",
            "reset_interval_ms": "20 s",
            "weights": {
                "sz_permil": "0 %",
                "nr_accesses_permil": "0 %",
                "age_permil": "1 %"
            }
        },
        "watermarks": { <none> },
        "filters": [ {
                "filter_type": "memcg",
                "memcg_path": "/hmsdk",
                "matching": false
        } ]
    }
]
```

```
"targets": [ { <CXL physical addr range> } ],
"schemes": [
    {
        "action": "migrate_hot",
        "access_pattern": {
            "sz_bytes": {"min": "4.000 KiB", "max": "max"},
            "nr_accesses": {"min": "5 %", "max": "max"},
            "age": {"min": "0 s", "max": "max"}
        },
        "quotas": {
            "time_ms": "2 s",
            "sz_bytes": "50 GiB",
            "reset_interval_ms": "20 s",
            "weights": {
                "sz_permil": "0 %",
                "nr_accesses_permil": "0 %",
                "age_permil": "1 %"
            }
        },
        "watermarks": { <none> },
        "filters": [ {
                "filter_type": "memcg",
                "memcg_path": "/hmsdk",
                "matching": false
        } ]
    }
]
```