Making Sense of Memory Tiering

Presented by

Alessandro Goncalves – Cloud Solutions Architect – INTEL

alessandro.goncalves@intel.com

Twitter: @alegoncalves12

AGENDA

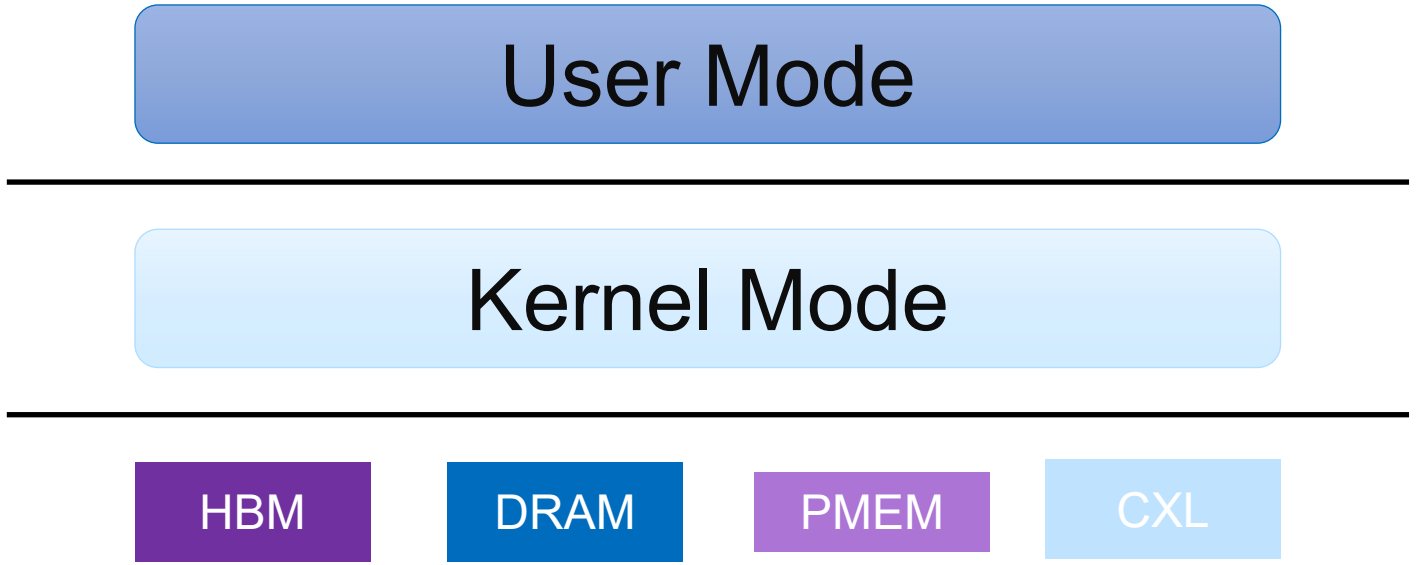
- Reasons to adopt Memory Tiering
- Taxonomy
- Memory Only NUMA Nodes
- Volatile Memory Tiering Types
- Call to Action

Reasons for Adoption

- CSP's report from 20 to 40% of stranded memory in compute hosts.
  - Hosts run out of CPU cores to rent before running out of memory
- Rented Memory is not fully utilized.
  - 50% of rented memory is infrequently touched
- CXL will bring memory pooling opportunities
  - Memory Tiering can benefit single hosts and clusters with shared memory devices.

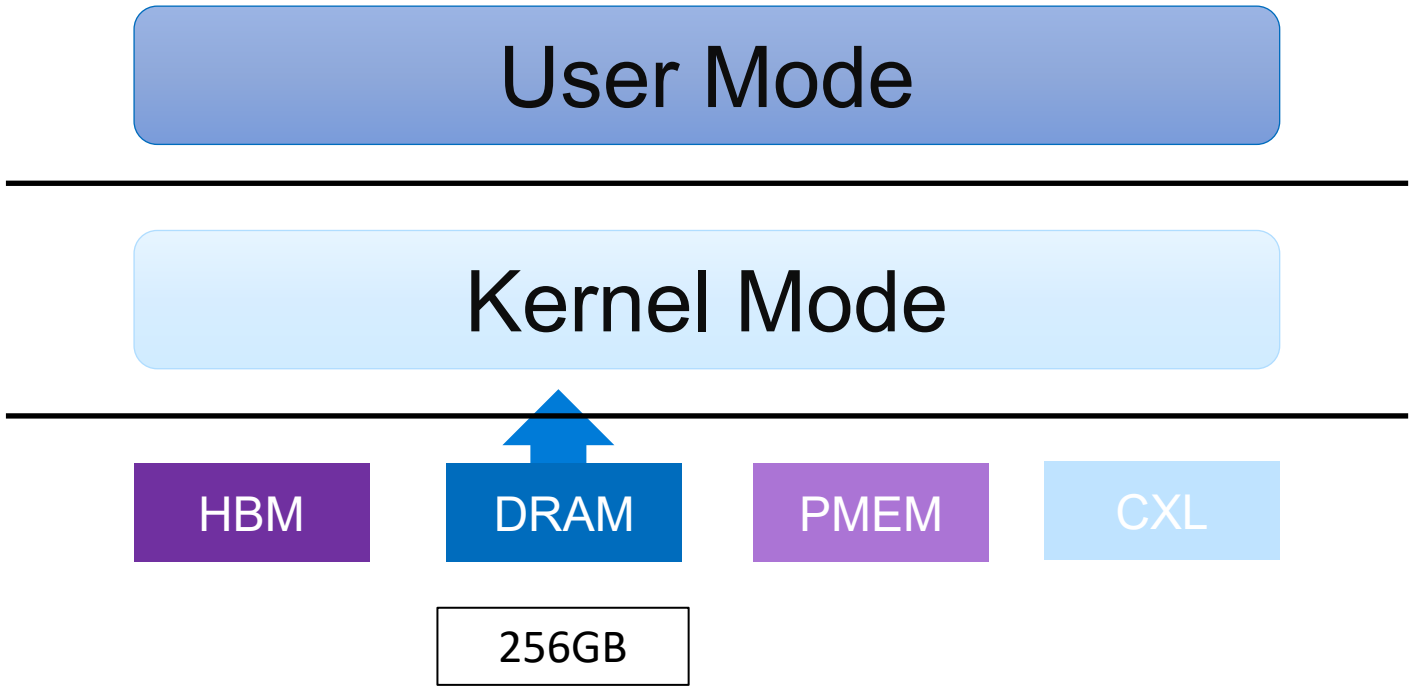
Taxonomy

- Memory Tiering – Different Memory Types working together to satisfy memory allocation. It may have dynamic page movement.



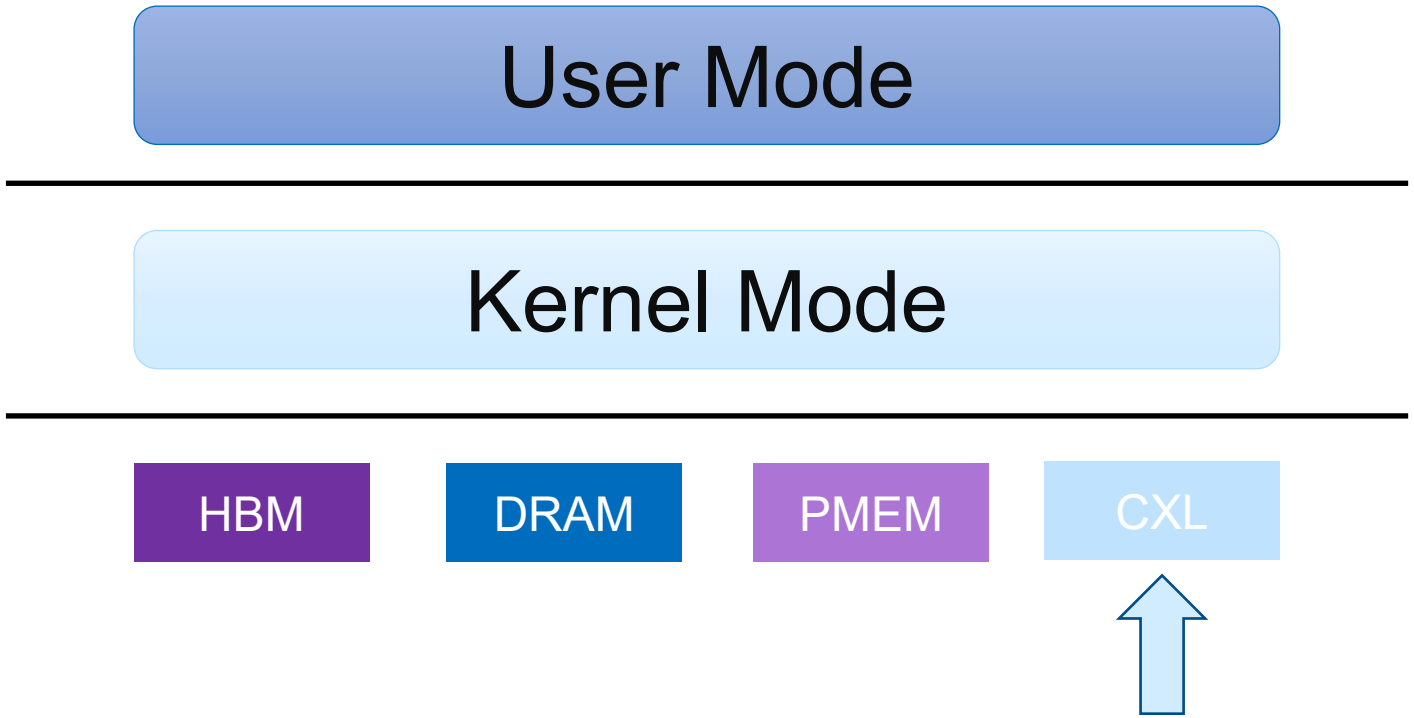
Taxonomy

- Main Memory – Total amount of addressable memory presented by the hardware to be managed by the OS kernel. It's associated with a CPU/socket.



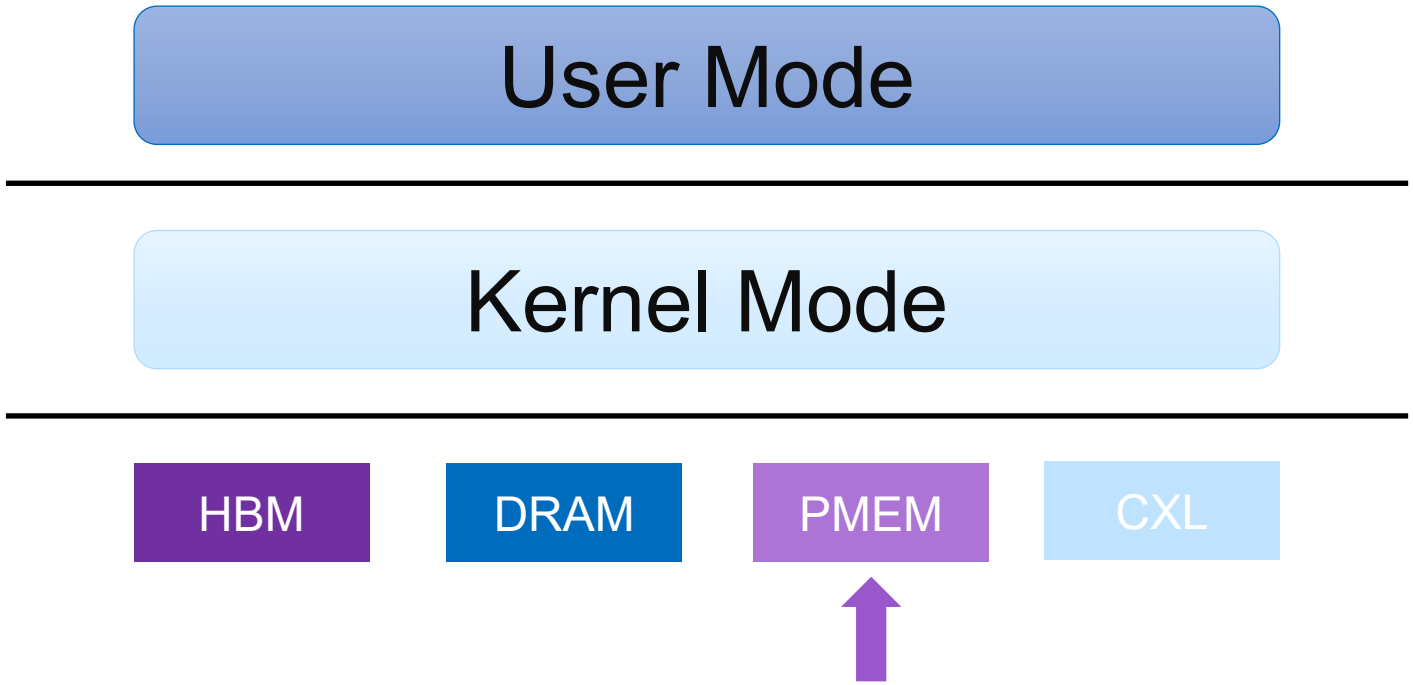
Taxonomy

- Special Purpose Memory – Memory space that is not associated with a CPU and presents itself as separated memory only NUMA node.



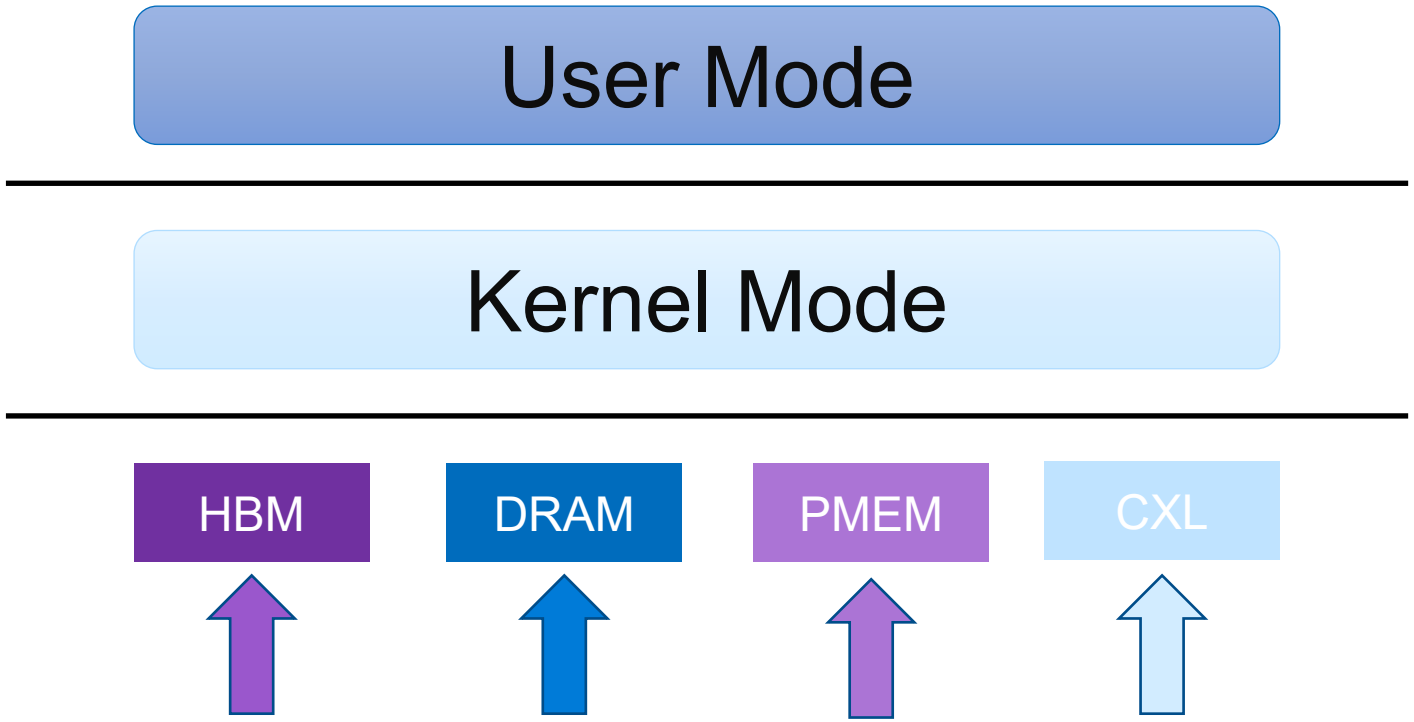
Taxonomy

- Persistent Memory – Memory device with Persistent behavior.



Taxonomy

- Addressable Memory – All Memory Spaces that can be mapped to fulfill allocation requests.

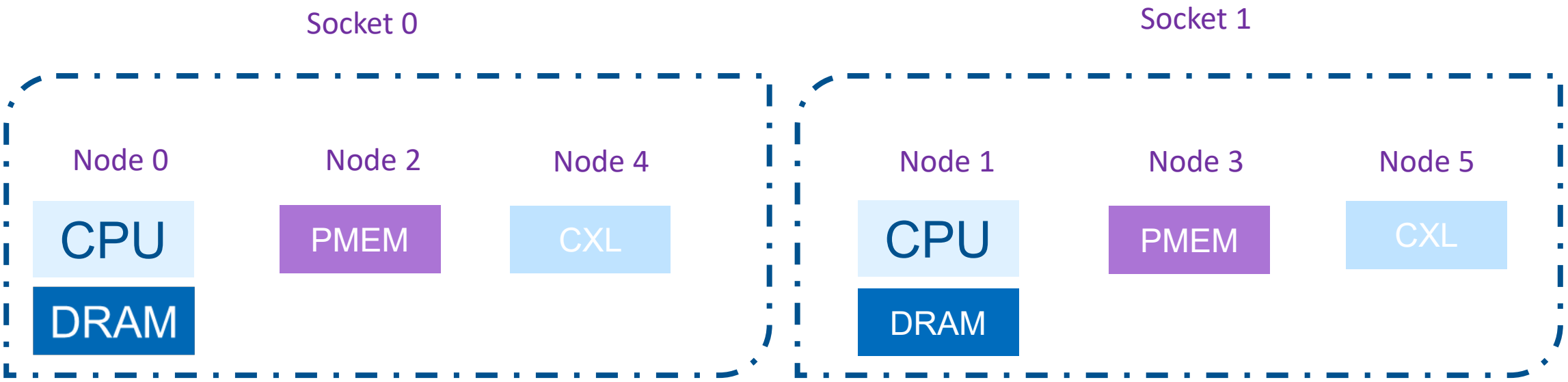


Memory Profile

- Each Memory will have properties which will place in a NUMA Domain.
- Each NUMA Memory Domain will contain same type of Memory.
  - Media
  - Bus
  - Latency
  - Bandwidth
- ACPI SLIT/SRAT Tables will be populated to display topology.

Memory Topology as NUMA

- Main Memory, Special Purpose Memory, and PMEM will show up as NUMA Nodes. However, SPM and PMEM will not be directly assigned to a CPU as traditional NUMA nodes but will be only Memory Domains.



NUMA Domains and Relative Distance

```
agoncalv@fmsolmrkt03:~$ numactl --hardware
available: 2 nodes (0-1)
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 48 49
 50 51 52 53 54 55 56 57 58 59 60 61 62 63 64 65 66 67 68 69 70 71
node 0 size: 1522675 MB
node 0 free: 1503830 MB
node 1 cpus: 24 25 26 27 28 29 30 31 32 33 72 73 74 75 76 77 78 79 80 81 82 83 84 85 86 87
6 47 72 73 74 75 76 77 78 79 80 81 82 83 84 85 86 87 88 89 90 91 92 93 94 95
node 1 size: 1524082 MB
node 1 free: 1519356 MB
node distances:
node   0   1
  0:  10  21
  1:  21  10
```

Host with Regular NUMA Nodes (CPU+Memory in every node)

```
[root@meta leader]# 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 48 49 50 51 52 53 54 55 112 113 114 115 116 117 118 119 120 121 122 1
23 124 125 126 127 128 129 130 131 132 133 134 135 136 137 138 139 140 141 142 143 144 145 146 147 148 149 1
50 151 152 153 154 155 156 157 158 159 160 161 162 163 164 165 166 167
node 0 size: 128630 MB
node 0 free: 118699 MB
node 1 cpus: 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 96 97 98 99 100 101 102 103 104 105 106 107 108 109 110 111 168 169 170 171 172 173
 174 175 176 177 178 179 180 181 182 183 184 185 186 187 188 189 190 191 192 193 194 195 196 197 198 199 200
 201 202 203 204 205 206 207 208 209 210 211 212 213 214 215 216 217 218 219 220 221 222 223
node 1 size: 128954 MB
node 1 free: 119107 MB
node 2 cpus:
node 2 size: 518144 MB
node 2 free: 518041 MB
node 3 cpus:
node 3 size: 518144 MB
node 3 free: 518144 MB
node distances:
node   0   1   2   3
  0:  10  21  22  32
  1:  21  10  32  22
  2:  22  32  10  32
  3:  32  22  32  10
```

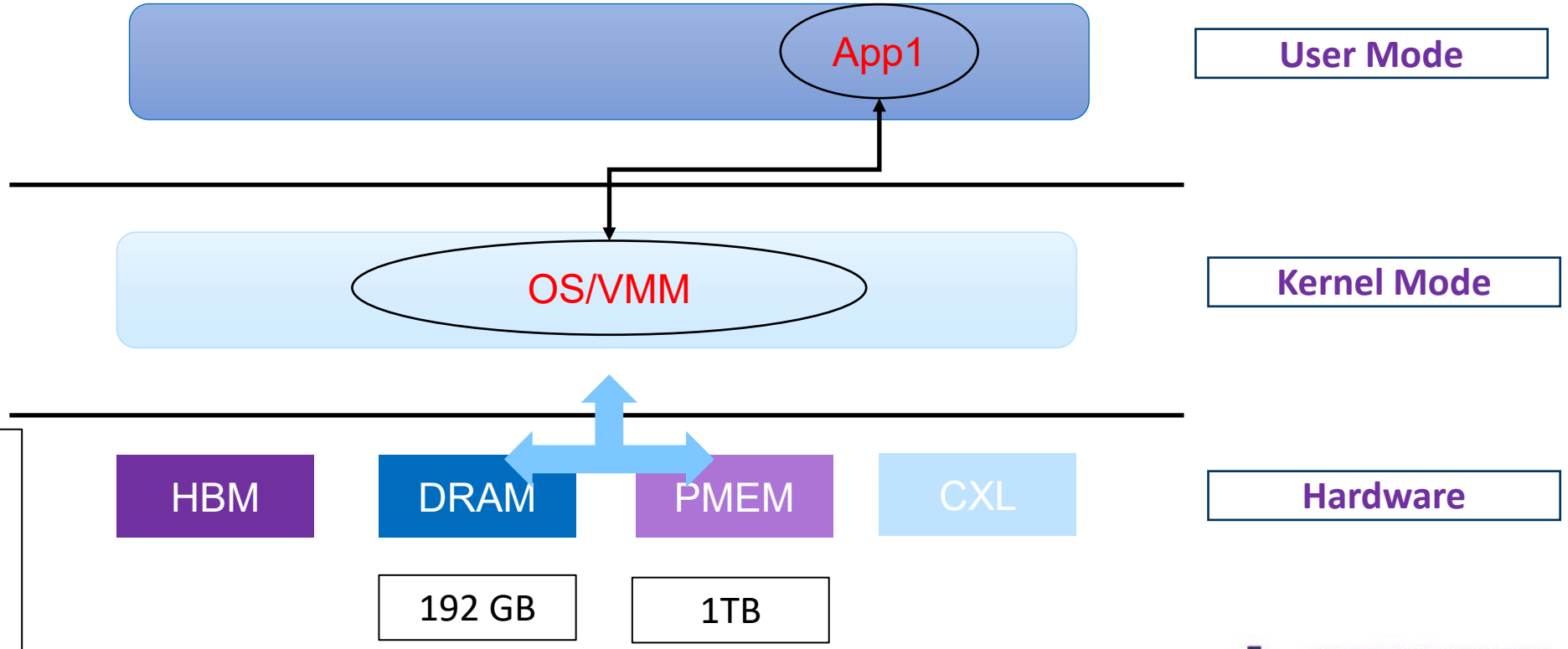
Host with Heterogenous Memory Topology. NUMA nodes with Memory Only no CPU

Tiering Types

- Classification on types of tiering methods based on where page placement logic is implemented.
- It can be software or hardware based.
- Memory topology understanding will determine if application transparency is achieved, or level of modification.
- Types are not exclusive and can be combined in advanced use cases.

Type I – Hardware Based Tiering

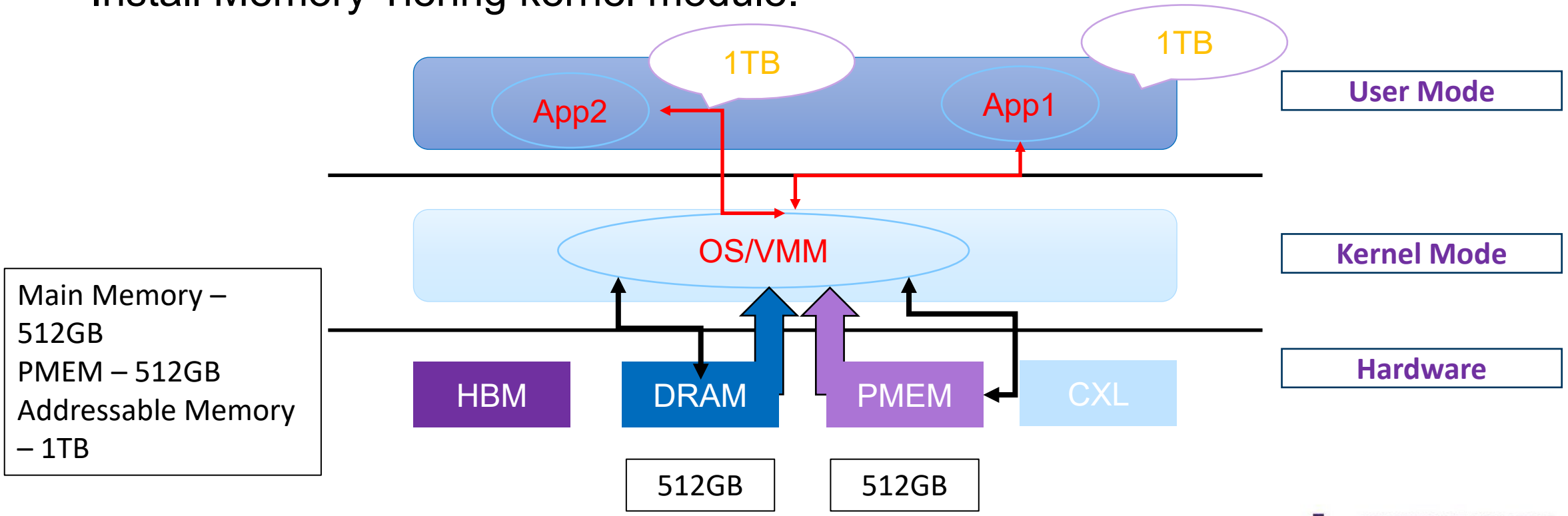
- Memory topology is abstracted from upper layers of kernel and user mode.
- No code change, and no kernel modules are needed to use Type I.
- Monolithic configuration and totally transparent to all layers.



Main Memory – 1TB Addressable Memory 1TB (Memory Mode with Optane® Persistent Memory)

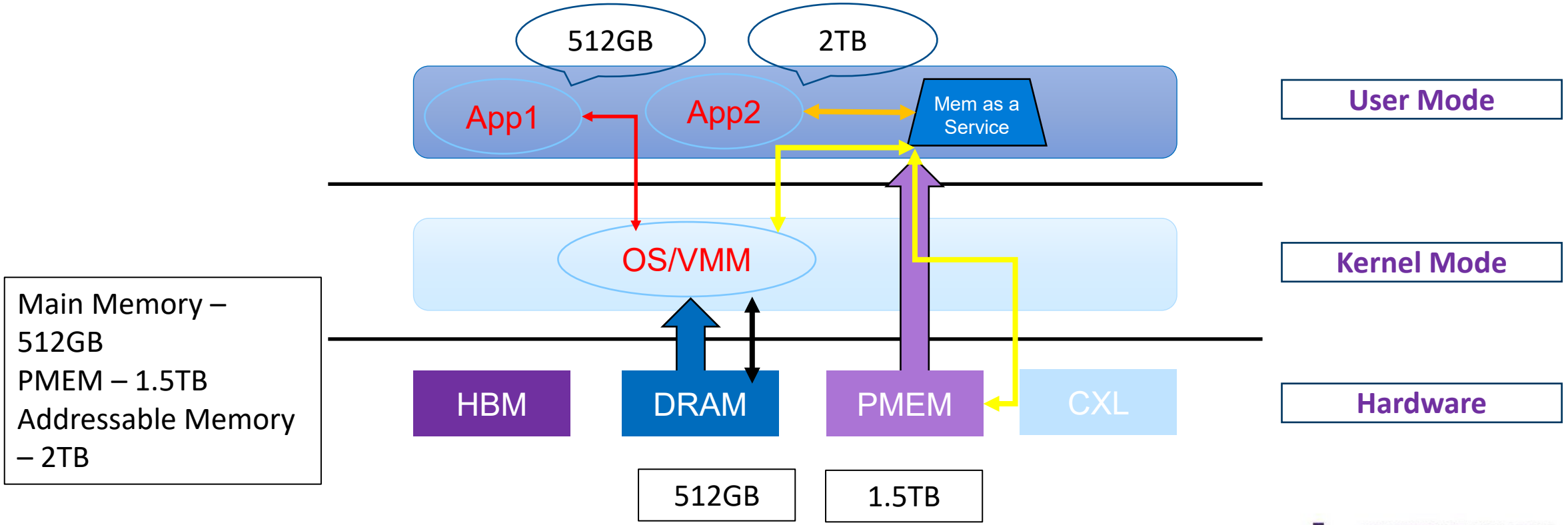
Type II – Kernel Based Tiering

- Memory topology is abstracted from User Space.
- Memory pages placement are controlled at the kernel level.
- No user space application changes needed.
- Install Memory Tiering kernel module.



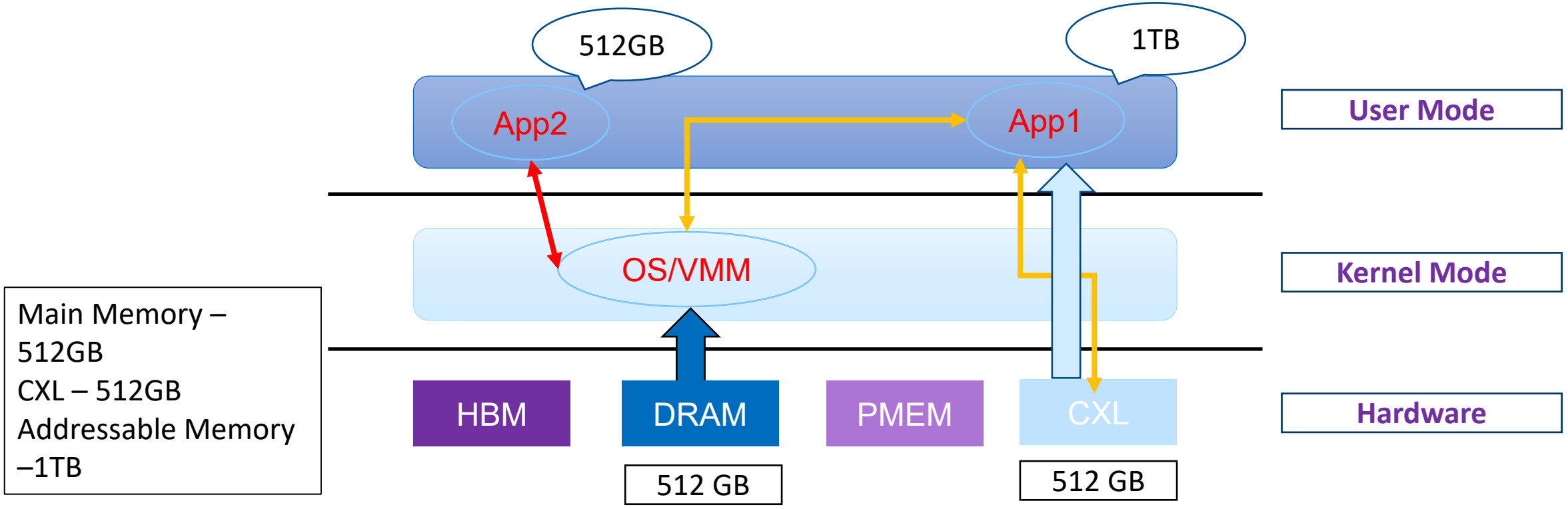
Type III – Memory as a Service

- Memory topology is presented to Mem_Service in User Space.
- Apps can use a Memory Service as “proxy” for memory allocation.
- Memory placement controlled by Memory Service.



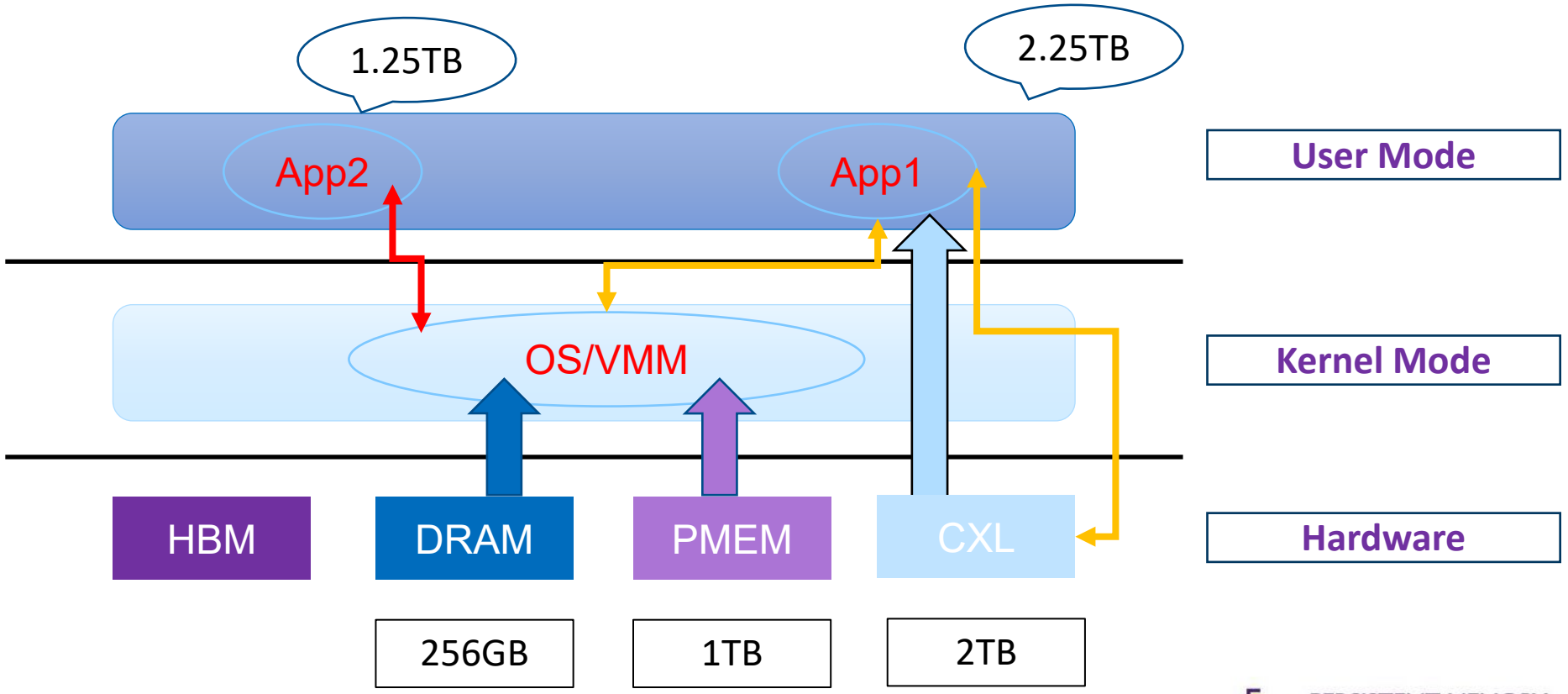
Type IV – Enlightened App

- Memory topology is visible by the application .
- Apps use Memory API to access and manage memory.
- Memory placement controlled by app.



Hybrid Mode – Multiple Types

- Combining two or more types of memory tiering in the system.
- Type II and Type IV – Kernel based plus a CXL Memory Device – in this example.



Main Memory - 128GB
PMEM – 1TB used by Kernel Tiering
CXL – 2TB used / controlled by the application.
Addressable Memory – 2.25TB

Call to Action

- Create better memory profiling and hot page detection.
- Create better prediction models to minimize effects of page promotion / demotion.
- Latency sensitivity models.
- Orchestration integration.

Please take a moment to rate this session.

Your feedback is important to us.

