CLAIMS

1. A method for demoting data from a cache comprising heterogeneous memory types, the method comprising:

maintaining, for a data element, a write access count that is incremented each time a data element is updated in a cache, the cache comprising a higher performance portion and a lower performance portion;

storing the data element in both the higher performance portion and the lower performance portion;

removing the data element from the higher performance portion in accordance with a cache demotion algorithm;

upon removing the data element from the higher performance portion, determining if the write access count is below a first threshold;

in response to removing the data element from the higher performance portion and determining that the write access count is below the first threshold, leaving the data element in the lower performance portion; and

in response to removing the data element from the higher performance portion and determining that the write access count is at or above the first threshold, removing the data element from the lower performance portion.

2. The method of claim 1, wherein the data element is a track.

3. The method of claim 1, wherein removing the data element from the higher performance portion comprises removing the data element from the higher performance portion when the higher performance portion is full.

4. The method of claim 1, wherein the higher performance portion is made up of dynamic random access memory (DRAM) and the lower performance portion is made up of storage class memory (SCM).

5. The method of claim 1, wherein the lower performance portion has a larger storage capacity than the higher performance portion.

6. A computer program product for demoting data from a cache comprising heterogeneous memory types, the computer program product comprising a non-transitory computer-readable storage medium having computer-usable program code embodied therein, the computer-usable program code configured to perform the following when executed by at least one processor:

maintain, for a data element, a write access count that is incremented each time a data element is updated in a cache, the cache comprising a higher performance portion and a lower performance portion;

store the data element in both the higher performance portion and the lower performance portion;

remove the data element from the higher performance portion in accordance with a cache demotion algorithm;

upon removing the data element from the higher performance portion, determine if the write access count is below a first threshold;

in response to removing the data element from the higher performance portion and determining that the write access count is below the first threshold, leave the data element in the lower performance portion; and

in response to removing the data element from the higher performance portion and determining that the write access count is at or above the first threshold, remove the data element from the lower performance portion.

7. The computer program product of claim 6, wherein the data element is a track.

8. The computer program product of claim 6, wherein removing the data element from the higher performance portion comprises removing the data element from the higher performance portion when the higher performance portion is full.

9. The computer program product of claim 6, wherein the higher performance portion is made up of dynamic random access memory (DRAM) and the lower performance portion is made up of storage class memory (SCM).

10. The computer program product of claim 6, wherein the lower performance portion has a larger storage capacity than the higher performance portion.

11. A system for demoting data from a cache comprising heterogeneous memory types, the system comprising:

at least one processor; and

at least one memory device operably coupled to the at least one processor and storing instructions for execution on the at least one processor, the instructions causing the at least one processor to:

maintain, for a data element, a write access count that is incremented each time a data element is updated in a cache, the cache comprising a higher performance portion and a lower performance portion;

store the data element in both the higher performance portion and the lower performance portion;

remove the data element from the higher performance portion in accordance with a cache demotion algorithm;

upon removing the data element from the higher performance portion, determine if the write access count is below a first threshold;

in response to removing the data element from the higher performance portion and determining that the write access count is below the first threshold, leave the data element in the lower performance portion; and

in response to removing the data element from the higher performance portion and determining that the write access count is at or above the first threshold, remove the data element from the lower performance portion.

12. The system of claim 11, wherein removing the data element from the higher performance portion comprises removing the data element from the higher performance portion when the higher performance portion is full.

13. The system of claim 11, wherein the higher performance portion is made up of dynamic random access memory (DRAM) and the lower performance portion is made up of storage class memory (SCM).

14. The system of claim 11, wherein the lower performance portion has a larger storage capacity than the higher performance portion.