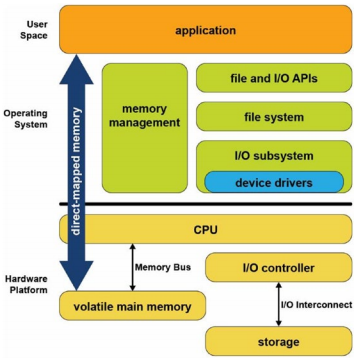
# Những gì đã tìm hiểu được:

1. Bộ nhớ không bay hơi (BNKBH) có thể truy cập dữ liệu theo từng byte giống như RAM.
2. Ứng dụng có thể truy xuất dữ liệu trực tiếp trên BNKBH mà không cần thông qua các thành phần như khi truy xuất bằng ổ cứng.



1. Bộ nhớ không bay hơi vẫn có tốc độ chậm hơn RAM, nên ta cần phải cân nhắc khi nào nên sử dụng RAM, khi nào nên sử dụng BNKBH, hoặc khi nào nên dùng cả hai. Intel khuyên rằng tỉ lệ giữa BNKBH và RAM là 4:1.
2. Bộ thư viện Persistent Memory Development Kit (PMDK) do Intel phát triển dùng để hổ trợ lập trình viên lập trình trên BNKBH dễ dàng hơn, và nó bao gồm các thư viện sau:
   * **Thư viện bay hơi (Volatile Libraries):**
     1. **libmemkind:**
        + Nó cho phép ta kiểm soát những đặc tính của bộ nhớ và phân vùng của heap giữa các loại bộ nhớ khác nhau.
        + Sử dụng khi ta muốn chuyển thủ công những đối tượng từ bộ nhớ sang BNKBH trong một chương trình bay hơi trong khi vẫn giữ lại được mô hình chương trình truyền thống.
     2. **libvmemcache:**
        + Là một giải pháp caching in-memory có thể nhúng và nhẹ mà vẫn có những lợi thế đến từ bộ nhớ có dung lượng lớn.
        + Sử dụng khi ta triễn khai caching cho những khối lượng công việc thường có hiệu suất vùng nhớ thấp mà cached trong một hệ thống với cách cấp phát bộ nhớ thông thường.
     3. **libvmem:**
        + Có nguồn gốc từ jemalloc, với metadata và việc cấp phát các đối tượng được trong file-base mapping.
        + Chỉ sử dụng nếu ứng dụng của ta đã sử dụng lbvmem hoặc ta cần phải có nhiều heap bộ nhớ hoàn toàn tách lập nhau. Nếu không ta nên dùng libmemkind.
   * **Thư viện không bay hơi (Persistent Libraries):**
     1. **libpmem:**
        + Là một thư viện C cấp thấp mà trừu tượng hoá các thao tác của hệ điều hành. Nó sẽ tự động phát hiện các tính năng có sẵn trong một nền tảng và chọn ra các biện pháp đúng đắn để tối ưu BNKBH.
        + Sử dụng khi ứng dụng của ta đã sử dụng memory-map I/O. Và khi ta muốn xây dựng lại mọi thứ từ đầu.
     2. **libpmemobj:**
        + Là một thư viện dùng để lưu trữ một transactional object, gồm có cấp phát vùng nhớ động thủ công, transaction và cơ sở chung cho lập trình BNKBH.
        + Sử dụng khi ngôn ngữ ta chọn là C và khi ta muốn sự linh hoạt trong việc thiết kế cấu trúc dữ liệu nhưng vẫn dùng cấp phát vùng nhớ chung và các transactions.
     3. **libpmemobj-cpp:**
        + Giống như thư viện libpmemobj nhưng hổ trợ ngôn ngữ C++.
        + Sử dụng khi ngôn ngữ ta chọn là C++. Nó được nhiều người ưa chuộng hơn thư viện libpmemobj.
     4. **libpmemkv:**
        + Là một thư viện lưu trự dạng key-value được tối ưu cho bộ nhớ không bay hơi.
        + Đây là thư viện được khuyên dùng cho người mới bắt đầu lập trình bộ nhớ không bay hơi. Sử dụng khi ta không cần các cấu trúc dữ liệu phức tạp mà vẫn có thể giải quyết vấn đề.
     5. **libpmemlog:**
        + Là một thư viện C ứng dụng cho việc log trên một file persistent memory append-only với các thao tác an toàn khi mất điện.
        + Chỉ dụng khi nhu cầu của ta được đáp ứng hoàn toàn với API của thư viện. Nếu không, một thư viện chung chung như libpmemobj hoặc libpmemobj-cpp có thể có ích hơn.
     6. **libpmemblk:**
        + Là một thư viện được dùng để quản lý các mảng block có kích thước cố định. Nó đảm bảo cho ta có thể cập nhật các block thông qua các hàm dựa trên buffer.

# Vấn đề gặp phải:

1. Chỉ chạy được một vài code mẫu vì gặp phải một số hàm không chạy được trên Windows hoặc thao tác sai ở một bước nào đó.
2. Có nhiều khái niệm vẫn chưa rõ.
3. Chưa biết nên bắt đầu từ đâu.