- What are some benefits of abstraction layers when it comes to hardware support in the OS?

Because there are many differences in lower level components in our computer's architecture, and because those components can differ in how they communicate with each other, without creating layers of abstraction, the OS would have many more problems to deal with and would be much more complex. Instead using layers of abstraction the OS doesn't need to be concerned with the behind the scenes minutia. It can instead focus on just an interface between it and the components, which allows many different components from different manufacturers to work well with one-another.

- If you switched from PIO to DMA-with-interrupts, how would that potentially improve performance?

PIO stands for Programed I/O, where the CPU is responsible for moving data to and from peripherals. This is relatively handholdy and can be costly considering it takes time to copy data from a peripheral to memory. Instead in DMA (Direct Memory Access) we can give the peripheral access and permissions to copy to memory. This can free up the CPU to work on other things and when the peripheral is done it can send an interrupt back to the CPU notifying it of the tasks completion.

- Why might you prefer memory-mapped I/O to port-based I/O?

There are two ways a CPU can actually communicate with a peripheral when it is plugged into the motherboard at a lower-level, either port I/O or Memory-Mapped I/O. For port I/O, the makers of a CPU and system might decide that the CPU can speak directly with ports, so it can then send IN or OUT instructions that send to or from individual ports. For Memory-Mapped I/O certain memory addresses are given special meanings that when accessed would cause some peripheral interaction to occur. There is not a great advantage to using one over the other, and both are still regularly used today.