Adam Prieto
Professor Ranjidha Rajan
CS 3600 - Operating Systems
10 May 2022

Final Paper

On the ninth page of the pdf, we can see the memory layout for the XV6 operating system on the left-hand side. Aside from the glaring comment at the top of the page, I was able to identify that this section is related to memory since the program begins with a mention of "PHYSTOP": a reference to physical memory. To quote from the Zybook chapter directly, "a computer's **Physical Memory (Ram)** is a hardware structure consisting of a linear sequence of words that hold a program during execution. A **word** is a fixed-size unit of data. A typical word size is four bytes but can also be a single byte." -(Zybooks Section 6.1). This section of code within the operating system is defining what addresses in RAM will be assigned to what functions as well as laying the groundwork for the **Kernel** (more on this in a moment). As I previously referenced, this content is consistent with Chapter 6 in the Zybook, and also leads into chapter 7 when we talked about **Virtual Memory**. To quote the Zybook again, "Virtual Memory is a collection of one or more logical memory spaces, each which may exceed the size of physical memory. A logical address is then referred to as a virtual address." -(Zybooks Section 7.1). In short, memory is the intermediary component between a computer's hard drive and CPU that holds data in an easily accessible state before it has operations performed on it by the processing unit.

I previously alluded to the **Kernel** in my last paragraph, since it's first mentioned on the same page as the memory references, but now I'm going to formally introduce the term: "The **Kernel**…is the minimal set of functions necessary to manage the system resources safely and efficiently. The kernel typically provides the most essential services for memory and device management, for creating and managing units of computation, and for communication among the different concurrent activities within the system." -(Zybook Section 1.2). Other mentions of the Kernel can be found on pages 17, 24, and 25; the last of which is where the actual defining code is found. Again, the source code has multiple, explicit comments stating that this section of code is dedicated to the kernel, but I could also tell that this section talks about the kernel since the left-hand side talks about mappings while the right-hand side of the page talks about CPU management: two subtopics explicitly mentioned in the Zybook. However, since Xv6 is a bit unique in terms of its kernel, I think it would be best to end this paragraph by quoting from the operating system directly: "The kernel keeps a cache of in-use inodes in memory to provide a place for synchronizing access to inodes used by multiple processes. The cached inodes include book-keeping information that is not stored on disk: ip->ref and ip->flags." -(Xv6 page 55 RHS).

With these two topics covered in-depth, I'd like to now introduce the final discussion topic: Files. We all have an intuitive understanding of dealing with computer files, whether it be pdfs, excel spreadsheets, computer programs, etcetera, a file is a condensed collection of data that performs some task for the user. To quote our friend, Mr. Zybook, again: "A **file** is a named collection of information managed on secondary storage by the [File System (FS)]. The user can view a file as a single abstract unit of data storage and access the file's contents using high-level operations provided by the FS's interface, without knowing the specific characteristics of the underlying storage devices. Depending on the FS, a file can be viewed as an unstructured stream of bytes or a series of records." -(Zybook Section 8.1). For this particular topic, I decided to look a little deeper into the operating system code presented and found a plethora of relevant code on page 63 of the pdf. The operating systems devote this entire section to file descriptors, file structures, and other relevant information. This whole page covers most of the content we introduced in chapter 8. It's also easy to tell that this section is dealing with files since the word itself is explicitly stated multiple times in code. Not a bad way of knowing that this part of the O.S. is dealing with relevant file operations such as read/ write, file types, etc.

In conclusion, this course built on my previous knowledge of computer software by formally defining and elaborating on my understanding of some topics. For example, I already had a vague sense of what a kernel was when I heard colleagues discuss the Linux kernel and Linus Torvalds, but now I have a concrete understanding of what they were talking about. I also learned of topics that were previously alien to me such as tracks when talking and disk scheduling. In all, this course was pretty enjoyable and gave me a deeper appreciation of the bridge between software and hardware that's known as the computer's operating system.