MazOS – Problem Statement

This document explains why this project was built and what it aims to explore.

Contents

1. Introduction

Operating systems are the backbone of the computing world. They serve a crucial role in facilitating the interaction of hardware and software. Historically operating systems were proprietarily owned by the hardware manufacturers, limiting access to their internal workings. Understanding the challenges involved in building an operating system from the ground up aims to serve as a learning experience of how the computers and machines we use daily function without the added layers of abstractions and oversimplifications.

The primary aim of this project is to produce a simple yet functional operating system that demonstrates the core OS components. The final product is envisioned to be a user-friendly command line interface rather than a production-ready OS.

2. Objectives

After considering all the background research the following objectives have been derived:

Build a 32-bit operating system able to support a x86 architecture.

- The OS must handle process and memory management efficiently.
- The OS must implement a FAT32 filesystem driver that will enable file operations.
- The OS mush have support for basic hardware components.
- The OS must include a command line interface, taking in consideration usability.
- The OS must provide a support for a number of basic POSIX system calls.

3. Background Research

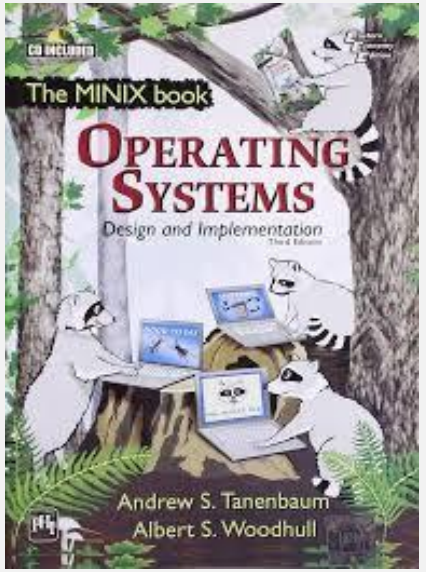
Looking at existing artifacts and studying the details behind their relevancy was a crucial step before taking any architectural and design choices. During the research stage the following existing operating systems were studied:

Unix

Many modern operating systems are built on the foundations of UNIX, making it an essential point of reference in the development of MazOS. Part of Unix's success could be attributed to its design principles that separate machine dependent code from machine independent code. MazOS will attempt to replicate this "separability" to maintain structure during the development stages.

Unix is known to have pioneered the POSIX standards, which are a set of IEEE defined rules that ensure that programs interact with the OS in a standardised way. To continue this idea of cross-compatibility MazOS will include support for some basic POSIX system calls, offering a familiar interface for POSIX-compliant software.

Minix



Minix was developed by Andrew S. Tanenbaum as a companion to his textbook 'Operating Systems: Design and Implementation', to exemplify the concepts of the OS development. The ethos behind MazOS is similar to the one of MINIX, it is intended as an exploration of low-level computing.

Minix is built around a microkernel architecture, which prioritises the strict isolation of components by having them executed in the user space. Whilst using a microkernel approach for MazOS was briefly considered, this idea was ultimately scratched due to the higher overhead associated with microkernels

Linux

Linux is considered as a pioneer not just for open-source operating systems but open-source software in general. Therefore, studying its intricacies was crucial in understanding what makes a successful software.

Linux's influence on MazOS is the decision of using a monolithic kernel, with a focus on simplicity and efficiency.

FreeRTOS

FreeRTOS is a real time operating system primarily used in embedded systems, because of its ability to guarantee that task execution is done in a precise, timely manner.

Although, MazOS's primary goals do not align with FreeRTOS's, its interrupt handling strategies to prioritise critical tasks has had a noticeable influence of MazOS's design.