SimpleOS – One Page Report

Done by Sai Shravan Neelamsetty, Sai Ganesh Venkata Vipanch Gadicherla, Vikas Meneni

1. **Introduction :** SimpleOS is a minimal 16-bit operating system implemented entirely in x86 real-mode assembly. The system is designed as an educational platform to demonstrate low-level OS architecture, BIOS-level hardware interaction, and command processing. Emphasis is placed on clarity, predictability, and hands-on exploration of fundamental OS mechanisms.

2. **System Purpose :** The primary purpose of SimpleOS is to provide students and developers with a transparent environment for studying system bootstrapping, memory operations, interrupt handling, and kernel command execution. The system includes both core shell functionality and a RAM-based file system, enabling exploration of command parsing, file table structures, and memory-backed storage models.

3. **Implemented Functions 3.1 Core Kernel Commands :**

- HELP: Displays all supported commands.
- LIST: Prints system and OS information.
- CLEAR: Clears the screen through video memory operations.
- ECHO: Outputs user text.
- TIME: Retrieves and displays the system time via BIOS interrupt 1Ah.
- DATE: Reads and prints the system date.
- PEEK: Reads a byte from a user-specified memory address.
- POKE: Writes a byte to a specified memory address.
- REBOOT: Performs a warm reboot by invoking BIOS routines.

3.2 **RAM-Based File System Commands :**

- CREATE: Allocates a new file entry and stores up to 64 bytes of data.
- DELETE: Removes a file from the RAM file table.
- RENAME: Changes the filename associated with an existing file.
- FILES: Lists all active files and their sizes.

4. **Technical Overview :** The system boots from a 512-byte bootloader, which is responsible for loading the kernel into memory and transferring control. The kernel implements a text-based shell, string utilities, memory utilities, and BIOS-interrupt-based hardware access routines. The RAM file system is implemented using fixed-size table entries for predictable behavior and low overhead.

5. **Total Lines of Code :**
   boot.asm: 97 lines
   kernel.asm: 1,236 lines
   Total: 1,333 lines