K-OS Project Proposal

Kai Chen

CS134c Spring 2003

Project Goals

- **Experiencing low level system design hardware interfacing.**
- **Understanding details of OS.**
- **Supporting other research projects.**

Objective

- **Support basic features of an OS.**
- **Keep everything as simple as possible.**
- **Extensible.**

Components

- **Boot Sector**
- **Interrupt Handler**
- **Memory Management**
- **I/O**
- **Process Management**
- **Application**
- **File System**

Boot Sector

- 512 bytes, ends with 0xAA55
- Initialize GDT
- Switch to protected mode
  - Test for CPU (want 386 or above)
  - Enable A20 address line
- Load rest of the kernel

Interrupt Handler

- **Initialize IDT, PIC**
- **Interrupt handlers**
  - **keyboard, video, timer, page fault, syscalls ...**
- **Central dispatching function**
  - **Maps interrupts to handlers**
  - **Each handler is implemented in C with assembly wrapper**

Memory Management

- **Basic Management**
- **Limited Virtual Memory with Paging**
  - **kernel space**
  - **user space**

I/O

- **Low level**
  - **I/O memory space. IN/OUT. I/O ports**
    - **video - 0x68000, 2 bytes each character**
    - **keyboard - interrupt driven, buffered, processes**
- **High level**
  - **library functions: printf() etc...**

Process Management

- **Process, TSS, Page directory**
- **Multi-tasking**
  - **Low Level - context switch**
  - **High Level - queues, synchronization primitives (e.g. semaphores)**
- **Scheduling**
  - **Round-Robin**
  - **User assigned priority**
  - **Reschedule**

Application

- **Linking/Loading**
- **Interfacing with kernel**
  - **System calls**

File System (Optional)

- **Avoid complication**
  - **DOS FAT12**
- **Floppy based**
  - **BIOS disk calls not available in pmode**
  - **load floppy image into memory**

Summary

- 4/25 - Boot Sector
- 5/2 - Interrupt Handler
- 5/11 - Memory Management
- 5/18 - I/O
- 5/25 - Process Management
- 5/30 - Application
- Optional - File System