CSCE 611: Operating Systems

Fall 2022

Machine Problem 6: Primitive Disk Device Driver

Name: Rajesh Satpathy

UIN: 931006158

Goal:

Investigate kernel-level device drivers on top of a simple programmed I/O device and implement support for blocking read and write operations without busy waiting in the device driver code.

Problem Statement:

Implement the functions for Blocking Disk and update the function implementations for read() and write().

Scope

- Implement the Blocking Disk as described above. Make sure that the disk does not use busy waiting to wait until the disk comes back from an I/O operation on blocking_disk.C and blocking_disk.H.
- Design and implement a thread-safe disk system. (Option 3 and Option 4)

Implementation:

The scheduler has been imported from the previous MP with modified changes. Changes are added to the simple_disk.H file to make the function issue_operations accessible at blocking_disk functions.

The wait_until_ready() function is modified to accommodate the block reading of the disk, compared to the loop that was getting used in the simple disk. The code checks if the disk is ready and adds it to the end of the scheduler's ready queue.

Changes were added in kernel.C file to support block disk implementation using #define _USES_BLOCKING and uncommenting and adding scheduler operations in makefile.

To support bonus question Option 3 the design is as follows,

- A list queue can be implemented at the blocking disk to accommodate multiple read and write requests to the disk. This will allow multiple-thread concurrency safely without facing race conditions.

To support bonus question Option 4, it is implemented as follows,

- A disk thread node queue is maintained and the CPU is yielded when the disk is ready. This is checked through the modified version wait_until_ready().

- Additional functions that are added to support this are:
    - add_threadnode_blockdisk: Add in a block disk thread to the concurrent queue.
    - get_blockdisk_head_thread: Returns the first thread by popping it from the disk queue.
    - wait_until_ready: Wait until the thread is ready for read/write operation and then add the thread node with read/write operation into the disk queue.

Reference is taken from simple disk functions to make modifications.

The output screenshots of the running kernel is attached below,