MP 6 Document

Option 1 & 2 Implemented

1. System Architecture

- The disk I/O subsystem is implemented in three layers:
- SimpleDisk (provided) - Basic blocking disk I/O operations
- NonBlockingDisk (implemented) - Extends SimpleDisk to provide non-blocking I/O
- ThreadSafeDisk (implemented) - Extends NonBlockingDisk to provide thread-safe disk access

2. NonBlockingDisk

NonBlockingDisk inherits from SimpleDisk and adds non-blocking I/O capabilities:

- Overrides wait_while_busy() to yield CPU instead of busy-waiting
- Maintains a request queue for pending disk operations
- When disk is busy, adds the request to a queue and yields
- Processes queued requests when disk becomes available
- Implements proper interrupt handling for thread operations

3. ThreadSafeDisk

ThreadSafeDisk inherits from NonBlockingDisk and adds thread safety:

- Uses mutexes to protect shared data structures (request queue)
- Makes a copy of data buffers for write operations to prevent race conditions
- Ensures concurrent access to disk operations is properly synchronized

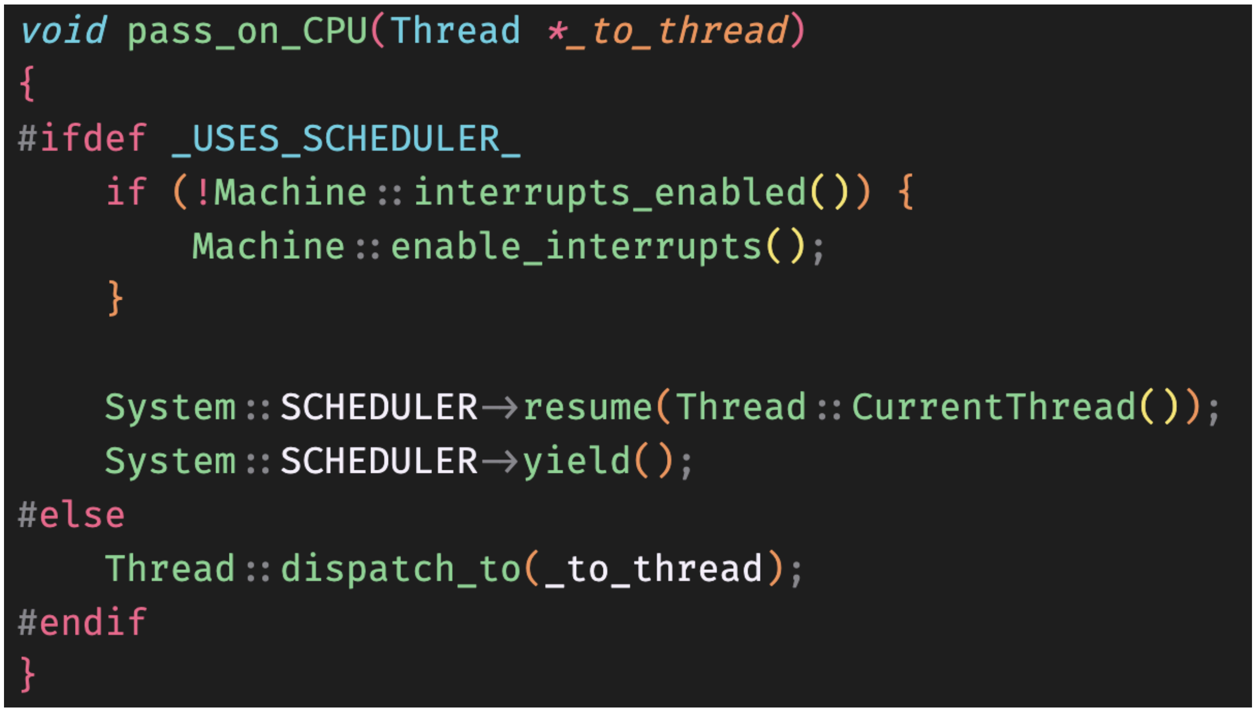
4. Mutex

- Carefully tracks interrupt state
- Properly handles yielding when the mutex is locked
- Maintains thread ownership information

5. Interrupt Handling

A critical aspect of the implementation is proper interrupt handling. The code ensures that:

- Interrupts are enabled when yielding the CPU
- Interrupts are disabled when modifying shared data structures
- The original interrupt state is preserved across operations
- For example, in kernel.C, the pass_on_CPU function was modified to safely handle interrupts:



```
void pass_on_CPU(Thread *_to_thread)
{
#ifdef _USES_SCHEDULER_
    if (!Machine::interrupts_enabled()) {
        Machine::enable_interrupts();
    }

    System::SCHEDULER->resume(Thread::CurrentThread());
    System::SCHEDULER->yield();
#else
    Thread::dispatch_to(_to_thread);
#endif
}
```

6. Modified Files

The following files were modified or created:

- NonBlockingDisk.H - Definition of the NonBlockingDisk class

- NonBlockingDisk.C - Implementation of non-blocking disk operations
- ThreadSafeDisk.H - Definition of the ThreadSafeDisk class
- ThreadSafeDisk.C - Implementation of thread-safe disk operations
- kernel.C - Modified to safely handle interrupt state in pass_on_CPU
- mutex.C - Enhanced to properly handle interrupt states