MP6: Primitive Disk Device Driver  
Kai-Chih Huang  
UIN: 333000021   
CSCE611: Operating System

Assigned Tasks

Main Part: Finished  
Bonus Option 1:   
Bonus Option 2: Not Finished  
Bonus Option 3: Not Finished  
Bonus Option 4: Not Finished

System Design

The goal of this machine problem is to implement kernel-level device driver that will not hold up CPU when the disk is not available. Instead of busy waiting, I call the scheduler, that I have done in previous machine problem, to queue up current thread then give up CPU to the next thread in the queue. The thread will wait in a FIFO queue until next round to check if the disk is available.

Code Description

I added the scheduler.H, scheduler.C that I did on MP5, then changed makefile, blocking\_disk.H, blocking\_disk.C for this machine problem. To compile code, simply run following command lines under MP6\_Sources directory:  
  
$ make clean // clean the old compile files before we compile

$ make // compile files

$ ./copykernel.sh (if permission denied, try chmod u+x ./copykernel.sh then do it again) //copy kernel

$ bochs -f bochsrc.bxrc // run bochs  
  
I will walk through the functions/methods defined in above files as follows.

**blocking\_disk.H**  
The design is basically the same as SimpleDisk. I rewrite the functions in the class and move theissue\_operation, is\_ready() and wait\_until\_reaty to public to prevent some unexpected bugs.  
A computer screen shot of text

Description automatically generated  
  
  
**blocking\_disk.C**  
BlockingDisk::wait\_until\_ready()  
This is the only function that I change in this MP. If the disk is not ready, then we will call the scheduler to add current thread to the end of the queue, then give up CPU to the next thread waiting in the queue. On the next round when this blocked thread is load back to CPU, then it will continue to another loop to check the disk.   
A screen shot of a computer code

Description automatically generated

BlockingDisk::read() / BlockingDisk::write()  
The implentation in the function is identical to which in the SimpleDisk. The difference is that the algorithm of wait\_until\_ready() it calls has changed, therefore when doing read or write, the thread will not keep holding up the CPU.

A computer screen shot of text

Description automatically generated  
  
A computer screen shot of text

Description automatically generated  
  
**makefile**  
I added the scheduler.o file to the makefile in order to correctly compile the scheduler to or kernel file  
A computer screen shot of a computer program

Description automatically generated

Testing

I did not add additional test data in this MP6, since I believe the provided test set is sufficient to check if the page table system works as expected. The test result is shown as follows.

Following screenshot shows that the threads can terminate appropriately. The thread first tries to read the disk then found it not available then give up the CPU. Therefore, we can see there is no writing messgage printed.

