Design Document

CISC 301-01

Matthew Sabo

12/6/2023

The design that was used was that mtsync.h was created to be a library to hold mtsync\_fence\_init() and mtsync\_fence\_point() and mtsync.c was designed to implement the logic of the two functions. The mtsync\_fence\_init() function was designed to initialize the MTSync fence to a specified length which was defined by the integer n. The mtsync\_fence\_point() function was designed to “catch” all of the threads until every thread was done with the specific step.

I implemented this design by using a mutex lock. After implementing the functions needed to use a mutex lock, in mtsync\_fence\_init(int number) I set the value of an integer named counter then defined n to the number variable. I then initialize a mutex lock so each thread that was created could reach the fence point at different times. After I initialized the mutex lock I then initialized a semaphore to control the access to the computer’s resources. In mtsync\_fence\_point(void), I locked the mutex lock and then increased the counter variable by 1. I then checked if all of the threads were at the fence point and if they were, I unlocked the semaphore. If all the threads weren’t at the fence point yet, then I unlocked the mutex lock and then made the threads wait until every thread got there.