CSC 456 Operating Systems
Dr. Christer Karlsson
Benjamin Kaiser
Programming Assignment 1
Part 1 PID Manager

Description of Program

This program is a simulation of a PID manager. The requirements of the assignment required us to use mutex locks to make sure there are no race conditions in the actual assignment of process identifiers. Another requirement was that we use POSIX threads to simulate each of the processes.

Description of algorithms and libraries used

The only special library that was used in this program was the pthread.h library. This library provides the functionality and everything for using POSIX threads.

A description of the algorithm is given in the section on functions and program structure.

Description of functions and program structure

This function was a procedurally designed program. It consists of five individual functions including main. These functions are spread throughout three different files.

The file entitled test.c contains the main function which initializes our mutex lock and one copy of the array containing which PID's are currently in use. The array has all of its values set to 0. The function then creates and allocates the pid map by calling the allocate_map function. This particular function is in the pid.c file. This function will be described later in the document. From here, main proceeds to create all of the threads necessary to simulate a process. It then loops through all of these threads and joins them onto the main thread so that it is blocked until the processes complete their iterations. Once they complete, the main thread resumes and the mutex lock is destroyed as well as a cleanup function is called which does some other cleanup.

The main file also has an array which contains a mapping of process ids which are currently in use. It also contains a mutex lock for this array so there is no race condition to this array. I left these two things in the program but in reality they are pretty useless since all actual important stuff is taken care of in the pid.c and pid.h files.

The pid.c file has a header file entitled pid.h. In this function the primary mutex is declared as well as the pid_map variable which is essentially the main and actual map of currently used process ID's. A variable called last is also declared. This variable is actually never used.

The pid.c file is a little more interesting. It contains four functions. The first is allocate_map. Because of the nature of how the PID map was declared, there isn't a whole lot

doing in here that actually has to do with initializing the PID map. Mostly it just sets the array to all 0's indicating that all of these PIDs are not currently in use. I also initialize the primary mutex in here.

The next function is the allocate_pid function. This function locks down the pid_map array list and only allows one thread ("process") into the list to get a PID. The way I implemented the function, it always starts at the minimum PID allowed and goes through until it finds a PID that is not currently in use. I then assign the index where I found the available PID and since I ignore any spots in the array which are lower than this index, the index is the PID. The variable that the index is assigned to is the value which gets returned from the allocate_pid function.

The third function is the release_pid function which takes in a PID. I lock the other processes out of the function so that they don't race. This isn't as necessary here but just to be safe, I did it anyways.

Description of testing and verification process

The way I tested this program was two run it several different times and examine the output by hand. The first few times, I ran it with 100 threads run 10 times as it was originally given to us by Dr. Karlsson. After this, I needed to test what would happen if there were no pids available. To do this, I bumped the thread count up to 1000 and the iterations to 10. This forced the program to run out of PIDs. It appeared to work appropriately after several runs.

Data

There was not really any data collected for this project. The only data that the program produces is its output which can be redirected to a file.

Analysis of Data

The only analysis of the data that I did was a manual observation of the file.

Description of Submission

The contents of the submission of this program are as follows:

- pid.c -- This file contains the functions to allocate, release and initialize the PIDs
- pid.h -- This file contains the declaration for the primary pid maps
- test.c -- This file contains the main function and all the code to test the manager