Jonathan J. Getahun
9/16/16

Project 1 Report

Part 1 (time-syscall.c):

In main(), I call the function syscall_timer(). At the beginning of syscall_timer(), I start a timer using gettimeofday(). Then, I have a while loop run 100,000 times, tracking the count with a global variable (glob_count). Inside the loop, I call getpid() and increment glob_count. After the loop has finished, I end the timer using gettimeofday(). In main(), when the program returns from syscall_timer(), the number of system calls performed is printed, using glob_count. Then, I calculate the length of time measured in syscall_timer() in microseconds (interval), and convert it to milliseconds (interval_in_ms). Finally, I print the total elapsed time in milliseconds using interval_in_ms, and the average time per system call by dividing interval_in_ms by glob_count (which is cast as a float).

Part 2 (time-signal.c):

In main(), I have three variables; x (equal to 1), y (equal to 2), and z (equal to 0). After these variables are declared and initialized, I call signal(SIGFPE, sigfpe_handler). Then, I set x equal to y/z (1 = 2/0). This division by 0 triggers the signal handler sigfpe_handler(). Inside it, it checks the global count (glob_count). If the count is 0, I start a timer using gettimeofday(). If the count is less than 100,000, I increment the global count. Otherwise, when the global count reaches 100,000, I end the timer and print the number of exceptions that have occurred, using glob_count. Then, I calculate the length of time measured in sigfpe_handler() in microseconds (interval), and convert it to milliseconds (interval_in_ms). Finally, I print the total elapsed time in milliseconds using interval_in_ms, and the average time per exception by dividing interval_in_ms by glob_count (which is cast as a float), followed by an exit call.