Homework Report

Homework Question: Calculate overhead of a syscall. Extend this to find the overhead of a context switch - between 2 processes and 2 threads.

Implementation Details:

Measurement:
Syscall and context switch overhead measurement requires high precision for which we have used the time stamp counter (TSC) that is present in x86 processors. To prevent Out-of-order execution we have used serializing instruction - CPUID before reading start time using RDTSC. To prevent variances introduced by CPUID, the instruction RDTSCP is used to read the end time from the counter. Also, to ensure code after RDTSCP executes in-order, CPUID instruction is called after RDTSCP.

The above set of instructions will introduce an extra overhead which needs to be subtracted from the final measurements. TSC has to be invariant for reliable measurements. Invariance of TSC can be verified by checking for constant_tsc and nonstop_tsc flags in /proc/cpuinfo

Syscall Overhead:
To calculate overhead of a syscall, we use the getpid() syscall (minimal execution time).

For Measurement overhead and Syscall overhead, we take 100 samples of 10000 measurements each. To exclude outliers and other factors affecting the measurements like interrupts, we have taken the minimum measurement from each sample.

Context Switch Overhead:

1. Between 2 processes:
   We use the fork() system call to create 2 processes. We measure the time taken by the parent process to switch to the child process. For this context switch, a pipe is used. The start time is written to the pipe in the parent process. To initiate a context switch, the parent process tries to read from another pipe. This results in parent process going to a blocked state and child process starts to run. The child reads the start time from the pipe and captures the end time. Child process then writes the time difference to the second pipe for which the parent process is waiting.
   The time difference that the parent process reads also includes the time taken to read and write the time to a pipe, which is an additional overhead. This also needs to be measured and subtracted from the context switch overhead measurement.

2. Between 2 threads:

We use the pthread_create() call to create 2 threads. Just like context switch measurement between 2 processes, we use pipe, write and read system calls in a similar fashion to switch between the threads.
The overhead of read and write to pipe is also subtracted from the final measurement.

For context switch overhead measurements, we took a sample of 100 measurements. To discard outliers, the minimum measurement out of the 100 is chosen as the final measurement.

The measurements are divided by the CPU frequency to get time in Nano seconds.

Other factors:

- To avoid inconsistencies in TSC due to the program running in multiple cores, we have pinned the process to a particular core using cpu_setaffinity().
- We have also assigned maximum priority to the process running the program to avoid inconsistencies due to preemptive scheduling.

Overhead.sh
The shell script checks if RDTSCP is a valid instruction in the instance. If yes, the CPU frequency is taken from /proc/cpuinfo and sample measurements are taken.

References:
https://www.intel.com/content/dam/www/public/us/en/documents/white-papers/ia-32-ia-64-benchmark-code-execution-paper.pdf