System Call Implementation

Definition of a System Call

A system call is a programmed request from a user-space application to the operating system's kernel to perform a privileged operation. Since user applications are restricted from directly accessing hardware and critical system resources, they rely on system calls to interact with these resources securely. Examples include file manipulation, process control, memory management, and time handling.

System calls serve as the interface between user programs and the kernel, allowing the user to request services like reading from a file, allocating memory, or retrieving the system time.

The clock_gettime() System Call

The clock_gettime() system call is used to retrieve the current time of a specified clock, most commonly the real-time clock. It returns the time in two components:

Seconds: The whole seconds part of the current time.

Nanoseconds: The fractional part of a second, offering high precision.

This system call is essential in systems that require precise timing, such as performance profilers, schedulers, or real-time applications.

The Purpose of Implementing clock_gettime() system call

The purpose of implementing the clock_gettime() system call is to:

-Understand how time is managed in the Linux kernel.

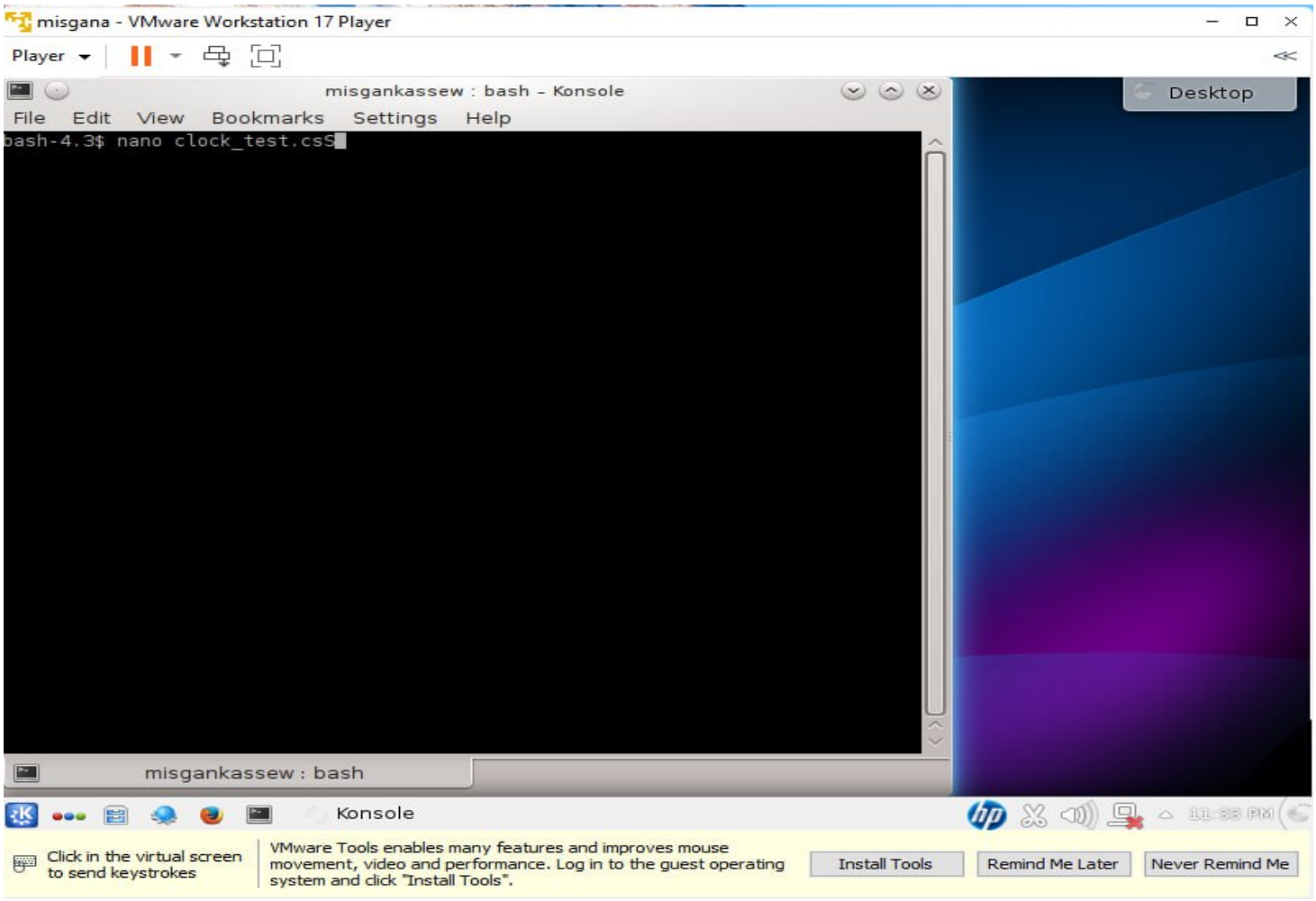
-Gain practical experience in kernel development.

-Enable applications to access accurate and high-resolution system time.

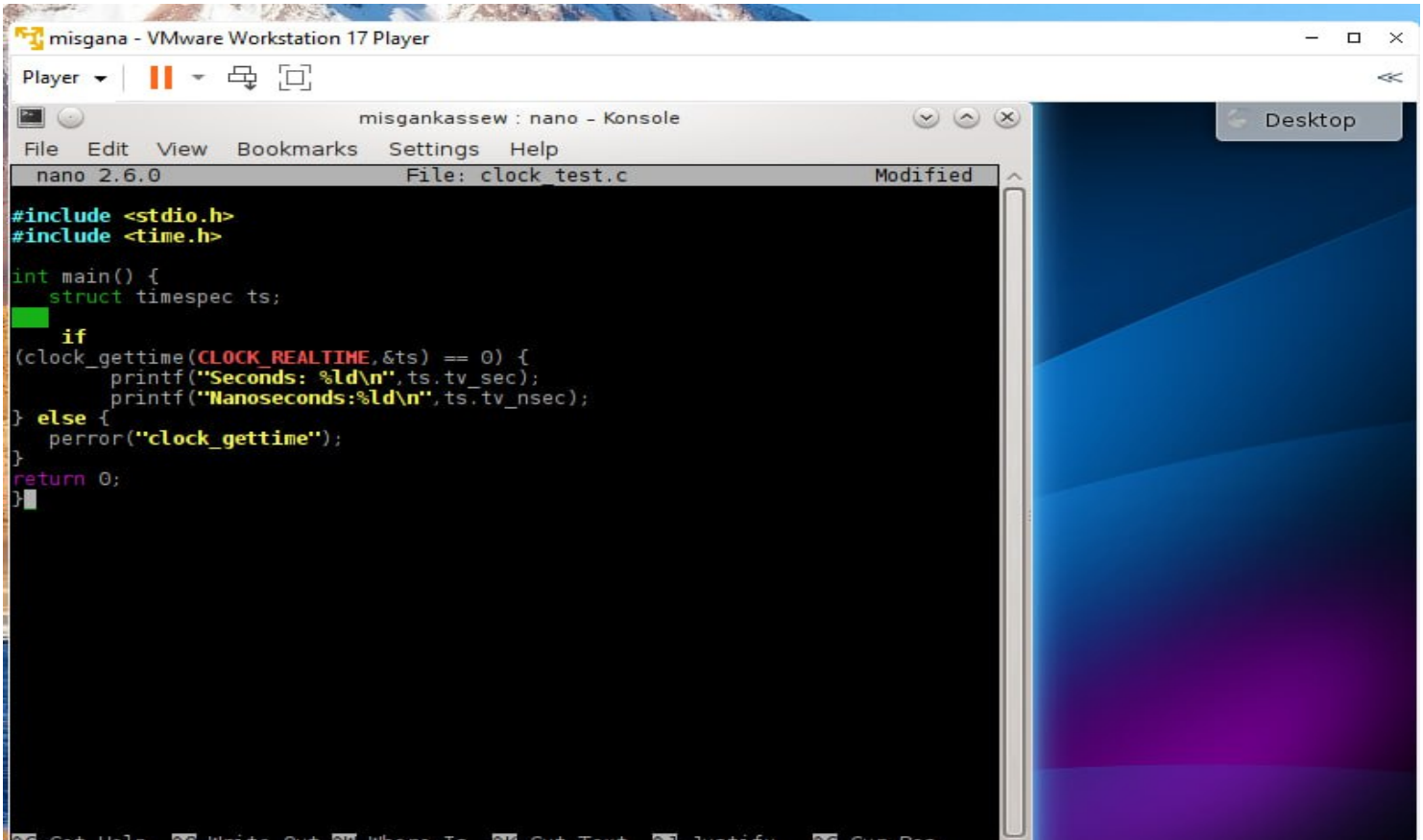
-This implementation also strengthens the understanding of how user-space and kernel-space communicate.

Steps that we followed during implementation are the following with their screeshots:

1.opend terminal and create a source file/a.c file named clock_test.c and then press enter.



Step 2.write a code to implement a given system call and press ctrl + O to save and then press enter and then press Ctrl +X to exit.



Step 3. Finally compile and run the program and see the results of second and nanosecond.