Department Of Software Engineering

Operating System and System Programming

Section B Individual Assignment

Title : Solaris OS Installation

Name	ID Number
Yaschilal Enyew	BDU1602719

Submitted To Lec Wendimu B.

Submission date 16/08/17

Question 3: **Implement System Calls (sigsuspend() - Suspends the calling process until a signal is received.)**

In Solaris, like in other Unix-like systems, the sigsuspend() system call is used to pause a running process until it receives a specific signal. This is useful in situations where a process needs to wait for a certain event, like waiting for user input, or when it's part of a larger system that needs to respond to signals from other processes.

How does sigsuspend() work?

- When a process calls sigsuspend(), it pauses and goes into a state where it waits for a signal to be sent to it.
- This system call is often used in combination with a signal handler, which is a function that handles the signal once it's received.
- Once the signal arrives, the process wakes up and continues from where it was paused. If a signal handler is set up, it will run first before continuing the execution.

Why use sigsuspend()?

It's helpful in scenarios where a program needs to wait for an event or action to occur, like when a user sends a specific signal (for example, pressing Ctrl+C or some custom action). This allows the process to remain inactive until the desired event takes place.

Example Code to Implement sigsuspend()

Here's a simple example to show how sigsuspend() can be used in a C program on Solaris: ***#include <stdio.h>***

```
#include <signal.h> #include
<unistd.h>
                void signal_handler(int sig) {
              printf("Signal %d received! Process is now resuming...\n",
sig);
                                    }
int main() {
                // Set up the signal handler for SIGUSR1
signal(SIGUSR1, signal_handler);
                printf("Process   suspended.   Waiting   for   SIGUSR1...\n");
//     Suspend     the     process     and     wait     for     the     signal
sigsuspend(NULL);  // Pauses the process until a signal is received


                     // This part will execute after the signal is handled
printf("Process resumed after receiving SIGUSR1!\n");     return 0;

}
```

Explanation of the Code:

1. **Setting up the Signal Handler:**
    - **signal(SIGUSR1, signal_handler); — This line tells the program to call the signal_handler function whenever it receives the SIGUSR1 signal.**
2. **Suspending the Process:**

- ***sigsuspend(NULL);* — This line pauses the process. It will remain paused until it gets a signal (in this case, SIGUSR1).**

3. **Handling the Signal:**
   - **Once the program receives SIGUSR1, it calls the signal_handler function, which prints a message and then lets the program continue executing.**

How to Test:

- **First => I Compile the Program using this code:** *cc -o sigsuspend_example sigsuspend_example.c*

- **then second =>I Run the Program**

```
./sigsuspend_example
```

Thirdly => I was Send the Signal from Another Terminal: Find the process ID (PID) of the running program using:

```
ps aux | grep sigsuspend_example
```

Then, send the SIGUSR1 signal Using this code:

```
kill -SIGUSR1 <PID>
```

After sending the signal, the process will resume and print the message "Process resumed after receiving SIGUSR1!"

Conclusion

The sigsuspend() system call is a powerful way to make a process pause and wait for specific signals. It is commonly used when a program needs to wait for an event, such as user input or other system events, before continuing execution. The combination of sigsuspend() with signal handlers makes it easy to control process flow in response to external actions or signals.

REFERENCES

✞ **. Solaris Operating System Documentation:**

- **Oracle Solaris Documentation:** **https://docs.oracle.com/en/solaris/**
- **Oracle Solaris 11.4 Installation Guide:** **https://docs.oracle.com/cd/E53394_01/html/E55603/index.html**
- **Oracle Solaris 11.4 System Administration Guide:** **https://docs.oracle.com/cd/E53394_01/html/E55603/index.html**

✞ **System Calls in Unix/Linux:**

- **The Linux Programming Interface (Book) by Michael Kerrisk:**
  - **A comprehensive reference for system calls in Unix-like systems. While focused on Linux, many system calls and**

principles are the same across Unix-like systems, including Solaris. ○ The Linux Programming Interface

- **POSIX System Calls (Manual):**
  - **POSIX Signal Handling** ○ **This standard documentation gives a detailed explanation of how signals are handled and includes system calls like sigsuspend().**

✞ **. Oracle Solaris System Calls and Signals:**

- **sigsuspend() Manual Page in Solaris:** https://docs.oracle.com/cd/E53394_01/html/E55603/sigsuspend-2.html
  This page explains how sigsuspend() works, its usage, and related functions.

✞ **Solaris File Systems (ZFS, UFS, etc.):**

- **Oracle Solaris ZFS Documentation:** https://docs.oracle.com/cd/E53394_01/html/E55603/zfsfile-system-usage.html
  An in-depth look at ZFS, which is the default file system in Solaris and a key topic in your assignment.

✞ **Oracle Solaris Network Configuration:**

- **Solaris Networking:** https://docs.oracle.com/cd/E53394_01/html/E55603/solaris-networking-guide.html **This guide provides detailed steps on configuring networking settings in Solaris, which was part of your installation process.**

✞ **Oracle Documentation for Solaris Installation and Setup:**

- **Oracle Solaris 11.4 Installation and Configuration Official guide for installing Solaris OS.**

✞ Practical Resources on Signals and System Calls:

- **Unix Signal Programming: https://beej.us/guide/bgipc/output/html/multipage/signals.html Beej's Guide is an excellent resource for understanding Unix signals and system calls like sigsuspend().**

✞ System Programming and Practical Examples:

✞ The Art of Unix Programming by Eric S. Raymond: http://www.faqs.org/docs/artu/ This book is a great resource for understanding the philosophy and practical aspects of Unix system calls and how they work in practice.