



Strings Arrays Pointers Memory Management File Handling C Programs C Interview Questions C MCQ

Signals in C language

[Read](#) [Discuss](#) [Courses](#) [Practice](#) [Video](#)

Prerequisite : [Fork system call](#), [Wait system call](#)

A signal is a software generated interrupt that is sent to a process by the OS because of when user press ctrl-c or another process tell something to this process.

There are fix set of signals that can be sent to a process. signal are identified by integers.

Signal number have symbolic names. For example **SIGCHLD** is number of the signal sent to the parent process when child terminates.

Examples:

```
#define SIGHUP 1 /* Hangup the process */
#define SIGINT 2 /* Interrupt the process */
#define SIGQUIT 3 /* Quit the process */
#define SIGILL 4 /* Illegal instruction. */
#define SIGTRAP 5 /* Trace trap. */
#define SIGABRT 6 /* Abort. */
```

OS Structures for Signals

- For each process, the operating system maintains 2 integers with the bits corresponding to a signal numbers.
- The two integers keep track of: **pending signals and blocked signals**
- With 32 bit integers, up to 32 different signals can be represented.

Example :

In the example below, the SIGINT (= 2) signal is blocked and no signals are pending.

Pending Signals

31	30	29	28	...	3	2	1	0
0	0	0	0	...	▲	0	0	0

Blocked Signals

31	30	29	28	...	3	2	1	0
0	0	0	0	...	0	1	0	0

A signal is sent to a process setting the corresponding bit in the pending signals integer for the process. Each time the OS selects a process to be run on a processor, the pending and blocked integers are checked. If no signals are pending, the process is restarted normally and continues executing at its next instruction.

If 1 or more signals are pending, but each one is blocked, the process is also restarted normally but with the signals still marked as pending. If 1 or more signals are pending and NOT blocked, the OS executes the routines in the process's code to handle the signals.

Default Signal Handlers

There are several default signal handler routines. Each signal is associated with one of these default handler routine. The different default handler routines typically have one of the following actions:

- Ign: Ignore the signal; i.e., do nothing, just return
- Term: terminate the process
- Cont: unblock a stopped process
- Stop: block the process

// CPP program to illustrate



```
// default Signal Handler
#include<stdio.h>
#include<signal.h>

int main()
{
    signal(SIGINT, handle_sigint);
    while (1)
    {
        printf("hello world\n");
        sleep(1);
    }
    return 0;
}
```

Output: Print hello world infinite times. If user presses ctrl-c to terminate the process because of **SIGINT** signal sent and its default handler to terminate the process.

```
hello world
hello world
hello world
terminated
```

User Defined Signal Handlers

A process can replace the default signal handler for almost all signals (but not SIGKILL) by its user's own handler function.

A signal handler function can have any name, but must have return type void and have one int parameter.

Example: you might choose the name sigchld_handler for a signal handler for the **SIGCHLD** signal (termination of a child process). Then the declaration would be:

```
void sigchld_handler(int sig);
```

When a signal handler executes, the parameter passed to it is the number of the signal. A programmer can use the same signal handler function to handle several signals. In this case the handler would need to check the parameter to see which signal was sent. On the other hand, if a signal handler function only handles one signal, it isn't necessary to bother examining the parameter since it will always be that signal number.

```
// CPP program to illustrate
// User-defined Signal Handler
```



```
#include<stdio.h>
#include<signal.h>

// Handler for SIGINT, caused by
// Ctrl-C at keyboard
void handle_sigint(int sig)
{
    printf("Caught signal %d\n", sig);
}

int main()
{
    signal(SIGINT, handle_sigint);
    while (1) ;
    return 0;
}
```

Output:

```
^CCaught signal 2 // when user presses ctrl-c
^CCaught signal 2
```

Sending signals via kill()

We can send a signal using kill() to the process.

```
int kill(pid_t pid, int signal);
pid: id of destination process
signal: the type of signal to send
Return value: 0 if signal was sent successfully
```

Example:

```
pid_t iPid = getpid(); /* Process gets its id.*/
kill(iPid, SIGINT); /* Process sends itself a SIGINT signal
```

```
(commits suicide?)(because of SIGINT
signal default handler is terminate the process) */
```

Questions

1. What is the Output of the following program?

```
#include<stdio.h>
#include<wait.h>
#include<signal.h>
int main()
{
    int stat;
    pid_t pid;
    if ((pid = fork()) == 0)
        while(1) ;
    else
    {
        kill(pid, SIGINT);
        wait(&stat);
        if (WIFSIGNALED(stat))
            psignal(WTERMSIG(stat), "Child term due to");
    }
}
```

Output:

Child term due to: Interrupt

2. What is the Output of the following program?

```
#include<stdio.h>
#include<signal.h>
#include<wait.h>
int val = 10;
void handler(int sig)
{
    val += 5;
}
int main()
{
    pid_t pid;
    signal(SIGCHLD, handler);
    if ((pid = fork()) == 0)
    {
        val -= 3;
        exit(0);
    }
    waitpid(pid, NULL, 0);
```



```
    printf("val = %d\n", val);
    exit(0);
}
```

Output:

```
val = 15
```

3. Consider the following code. What is the output?

```
#include<stdio.h>
#include<wait.h>
#include<signal.h>
pid_t pid;
int counter = 0;
void handler1(int sig)
{
    counter++;
    printf("counter = %d\n", counter);
    /* Flushes the printed string to stdout */
    fflush(stdout);
    kill(pid, SIGUSR1);
}
void handler2(int sig)
{
    counter += 3;
    printf("counter = %d\n", counter);
    exit(0);
}

int main()
{
    pid_t p;
    int status;
    signal(SIGUSR1, handler1);
    if ((pid = fork()) == 0)
    {
        signal(SIGUSR1, handler2);
        kill(getppid(), SIGUSR1);
        while(1) ;
    }
    if ((p = wait(&status)) > 0)
    {
        counter += 4;
        printf("counter = %d\n", counter);
    }
}
```

Output

```
counter = 1          //(parent's handler)
```



```
counter = 3          //(child's handler)
counter = 5          //(parent's main)
```

This article is contributed by **Kadam Patel**. If you like GeeksforGeeks and would like to contribute, you can also write an article using contribute.geeksforgeeks.org or mail your article to contribute@geeksforgeeks.org. See your article appearing on the GeeksforGeeks main page and help other Geeks.

Please write comments if you find anything incorrect, or you want to share more information about the topic discussed above.

Last Updated : 08 Feb, 2018

25

Similar Reads

1. Program error signals
2. Control Signals in 8155 Microprocessor
3. Communication between two process using signals in C
4. Difference Between C Language and LISP Language
5. A C Programming Language Puzzle
6. Convert C/C++ code to assembly language
7. Difference between %d and %i format specifier in C language
8. lvalue and rvalue in C language
9. Difference between while(1) and while(0) in C language
10. kbhit in C language

Previous

**Communication between two process
using signals in C**

Next

Wait System Call in C



Article Contributed By :



GeeksforGeeks

Vote for difficulty

Current difficulty : [Medium](#)

[Easy](#)[Normal](#)[Medium](#)[Hard](#)[Expert](#)

Article Tags : [C-Library](#), [system-programming](#), [C Language](#)

[Improve Article](#)[Report Issue](#)

GeeksforGeeks

A-143, 9th Floor, Sovereign Corporate
Tower, Sector-136, Noida, Uttar Pradesh -
201305

feedback@geeksforgeeks.org

Company

[About Us](#)[Careers](#)[In Media](#)[Contact Us](#)[Terms and Conditions](#)[Privacy Policy](#)[Copyright Policy](#)[Third-Party Copyright Notices](#)[Advertise with us](#)

Explore

[Job Fair For Students](#)[POTD: Revamped](#)[Python Backend LIVE](#)[Android App Development](#)[DevOps LIVE](#)[DSA in JavaScript](#)

Languages

[Python](#)[Java](#)[C++](#)[GoLang](#)[SQL](#)[R Language](#)

Data Structures

[Array](#)[String](#)[Linked List](#)[Stack](#)[Queue](#)[Tree](#)

[Android Tutorial](#)[Graph](#)

Algorithms

[Sorting](#)[Searching](#)[Greedy](#)[Dynamic Programming](#)[Pattern Searching](#)[Recursion](#)[Backtracking](#)

Computer Science

[GATE CS Notes](#)[Operating Systems](#)[Computer Network](#)[Database Management System](#)[Software Engineering](#)[Digital Logic Design](#)[Engineering Maths](#)

Data Science & ML

[Data Science With Python](#)[Data Science For Beginner](#)[Machine Learning Tutorial](#)[Maths For Machine Learning](#)[Pandas Tutorial](#)[NumPy Tutorial](#)[NLP Tutorial](#)[Deep Learning Tutorial](#)

Competitive Programming

[Top DSA for CP](#)[Top 50 Tree Problems](#)[Top 50 Graph Problems](#)[Top 50 Array Problems](#)[Top 50 String Problems](#)[Top 50 DP Problems](#)[Top 15 Websites for CP](#)

Web Development

[HTML](#)[CSS](#)[JavaScript](#)[Bootstrap](#)[ReactJS](#)[AngularJS](#)[NodeJS](#)

Python

[Python Programming Examples](#)[Django Tutorial](#)[Python Projects](#)[Python Tkinter](#)[OpenCV Python Tutorial](#)[Python Interview Question](#)

DevOps

[Git](#)[AWS](#)[Docker](#)[Kubernetes](#)[Azure](#)[GCP](#)

System Design

[What is System Design](#)[Monolithic and Distributed SD](#)[Scalability in SD](#)[Databases in SD](#)[High Level Design or HLD](#)[Low Level Design or LLD](#)[Top SD Interview Questions](#)

Interview Corner

Company Preparation
Preparation for SDE
Company Interview Corner
Experienced Interview
Internship Interview
Competitive Programming
Aptitude

Commerce

Accountancy
Business Studies
Microeconomics
Macroeconomics
Statistics for Economics
Indian Economic Development

SSC/ BANKING

SSC CGL Syllabus
SBI PO Syllabus
SBI Clerk Syllabus
IBPS PO Syllabus
IBPS Clerk Syllabus
Aptitude Questions
SSC CGL Practice Papers

GfG School

CBSE Notes for Class 8
CBSE Notes for Class 9
CBSE Notes for Class 10
CBSE Notes for Class 11
CBSE Notes for Class 12
English Grammar

UPSC

Polity Notes
Geography Notes
History Notes
Science and Technology Notes
Economics Notes
Important Topics in Ethics
UPSC Previous Year Papers

Write & Earn

Write an Article
Improve an Article
Pick Topics to Write
Write Interview Experience
Internships
Video Internship

@geeksforgeeks , Some rights reserved

