Multithreading in C

Difficulty Level : Medium ● Last Updated : 10 Oct, 2018

What is a Thread?
A thread is a single sequence stream within in a process. Because threads have some of the properties of processes, they are sometimes called *lightweight processes*.

What are the differences between process and thread?
Threads are not independent of one other like processes as a result threads shares with other threads their code section, data section and OS resources like open files and signals. But, like process, a thread has its own program counter (PC), a register set, and a stack space.

Why Multithreading?
Threads are popular way to improve application through parallelism. For example, in a browser, multiple tabs can be different threads. MS word uses multiple threads, one thread to format the text, other thread to process inputs, etc.
Threads operate faster than processes due to following reasons:
1) Thread creation is much faster.
2) Context switching between threads is much faster.
3) Threads can be terminated easily
4) Communication between threads is faster.

See http://www.personal.kent.edu/~rmuhamma/OpSystems/Myos/threads.htm for more details.

Can we write multithreading programs in C?
Unlike Java, multithreading is not supported by the language standard. POSIX Threads (or Pthreads) is a POSIX standard for threads. Implementation of pthread is available with gcc compiler.

A simple C program to demonstrate use of pthread basic functions
Please note that the below program may compile only with C compilers with pthread library.

```
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>  //Header file for sleep(). man 3 sleep for details.
#include <pthread.h>

// A normal C function that is executed as a thread
// when its name is specified in pthread_create()
void *myThreadFun(void *vargp)
{
    sleep(1);
    printf("Printing GeeksQuiz from Thread \n");
    return NULL;
}

int main()
{
    pthread_t thread_id;
    printf("Before Thread\n");
    pthread_create(&thread_id, NULL, myThreadFun, NULL);
    pthread_join(thread_id, NULL);
    printf("After Thread\n");
    exit(0);
}
```

main() we declare a variable called thread_id, which is of type pthread_t, which is an integer used to identify the thread in the system. After declaring thread_id, we call pthread_create() function to create a thread.

pthread_create() takes 4 arguments.
The first argument is a pointer to thread_id which is set by this function.
The second argument specifies attributes. If the value is NULL, then default attributes shall be used.
The third argument is name of function to be executed for the thread to be created.
The fourth argument is used to pass arguments to the function, myThreadFun.
The pthread_join() function for threads is the equivalent of wait() for processes. A call to pthread_join blocks the calling thread until the thread with identifier equal to the first argument terminates.

How to compile above program?
To compile a multithreaded program using gcc, we need to link it with the pthreads library. Following is the command used to compile the program.

```
gfg@ubuntu:~/$ gcc multithread.c -lpthread
gfg@ubuntu:~/$ ./a.out
Before Thread
Printing GeeksQuiz from Thread
After Thread
gfg@ubuntu:~/$
```

A C program to show multiple threads with global and static variables
As mentioned above, all threads share data segment. Global and static variables are stored in data segment. Therefore, they are shared by all threads. The following example program demonstrates the same.

```
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
#include <pthread.h>

// Let us create a global variable to change it in threads
int g = 0;

// The function to be executed by all threads
void *myThreadFun(void *vargp)
{
    // Store the value argument passed to this thread
    int *myid = (int *)vargp;

    // Let us create a static variable to observe its changes
    static int s = 0;

    // Change static and global variables
    ++s; ++g;
```

```
    // Print the argument, static and global variables
    printf("Thread ID: %d, Static: %d, Global: %d\n", *myid, ++s, ++g);
}

int main()
{
    int i;
    pthread_t tid;

    // Let us create three threads
    for (i = 0; i < 3; i++)
        pthread_create(&tid, NULL, myThreadFun, (void *)&tid);

    pthread_exit(NULL);
    return 0;
}
```

```
gfg@ubuntu:~/$ gcc multithread.c -lpthread
gfg@ubuntu:~/$ ./a.out
Thread ID: 3, Static: 2, Global: 2
Thread ID: 3, Static: 4, Global: 4
Thread ID: 3, Static: 6, Global: 6
gfg@ubuntu:~/$
```

Please note that above is simple example to show how threads work. Accessing a global variable in a thread is generally a bad idea. What if thread 2 has priority over thread 1 and thread 1 needs to change the variable. In practice, if it is required to access global variable by multiple threads, then they should be accessed using a mutex.

References:

http://www.csc.villanova.edu/~mdamian/threads/posixthreads.html
Computer Systems : A Programmer

This article is contributed by **Rahul Jain**. Please write comments if you find anything incorrect, or you want to share more information about the topic discussed above

Like

Previous Next

RECOMMENDED ARTICLES

Page : 1 2

01 **Handling multiple clients on server with multithreading using Socket Programming in C/C++**
10, Aug 21

02 **Multithreading in C++**
08, Jan 18

03 **strcat() function in C/C++ with Example**
14, Oct 21

04 **All forms of formatted scanf() in C**
22, Sep 21

05 **How to input or read a Character, Word and a Sentence from user in C?**
21, Sep 21

06 **What will happen if a print() statement is written inside a if() such as if(print())**
13, Aug 21

07 **Handling multiple clients on server with multithreading using Socket Programming in C/C++**
10, Aug 21

08 **Animation of Tower Of Hanoi using computer graphics in C/C++**
06, Aug 21

Article Contributed By :

GeeksforGeeks

Vote for difficulty

Current difficulty : Medium

Easy | Normal | Medium | Hard | Expert

Improved By : CristianBurungiu, PratikRoy, DipakAgrawal, abhikbose23

Article Tags : C-Library, cpp-multithreading, system-programming, C Language

Report Issue

Writing code in comment? Please use ide.geeksforgeeks.org, generate link and share the link here.

Load Comments

GeeksforGeeks
5th Floor, A-118,
Sector-136, Noida, Uttar Pradesh - 201305

feedback@geeksforgeeks.org