

**SPONGE LIU**

Love what you do



- [SpongeLiu的博客](#)
- [linux](#)
  - [linux内核](#)
  - [linux系统](#)
- [系统结构](#)
- [算法](#)
  - [其他](#)
  - [数据结构](#)
- [语言学习](#)
  - [C语言](#)
- [笔试面试](#)
- [应用问题](#)
  - [Gentoo](#)
- [About](#)
- [留言板](#)

[gcc的内联汇编取全局变量地址](#) [快速判断一个32位的字中是否存在值为"0"的byte](#)

## do{...}while(0)的意义和用法

On October 9, 2012, in [C语言](#), [语言学习](#), by sponge

linux内核和其他一些开源的代码中，经常会遇到这样的代码：

```
do{
...
}while(0)
```

这样的代码一看就不是一个循环，do..while表面上在这里一点意义都没有，那么为什么要这么用呢？

实际上，do{...}while(0)的作用远大于美化你的代码。查了些资料，总结起来这样写主要有以下几点好处：

### 1、辅助定义复杂的宏，避免引用的时候出错：

举例来说，假设你需要定义这样一个宏：

```
#define DOSOMETHING()\
    foo1();\
    foo2();
```

这个宏的本意是，当调用DOSOMETHING()时，函数foo1()和foo2()都会被调用。但是如果你在调用的时候这么写：

```
if(a>0)
    DOSOMETHING();
```

因为宏在预处理的时候会被直接展开，你实际上写的代码是这个样子的：

```
if(a>0)
    foo1();
    foo2();
```

这就出现了问题，**因为无论a是否大于0，foo2()都会被执行，导致程序出错。**

那么**仅仅使用{}**将foo1()和foo2()包起来行么？

我们在写代码的时候都习惯在语句右面加上分号，如果在宏中使用{}，代码里就相当于这样写了：“{...};”，展开后就是这个样子：

```
if(a>0)
{
    foo1();
    foo2();
};
```

这样甚至不会编译通过。所以，很多人才采用了do{...}while(0);

```
#define DOSOMETHING() \
do{ \
    foo1();\
    foo2();\
}
```

```
    }while(0)\n\n...\n\nif(a>0)\n    DOSOMETHING();\n\n...
```

这样，宏被展开后，才会保留初始的语义。GCC提供了[Statement-Expressions](#)用以替代do{...}while(0); 所以你也可以这样定义宏：

```
#define DOSOMETHING() ({\n    fool(); \n    foo2(); \n})
```

<http://www.spongeliu.com/>

## 2、避免使用goto对程序流进行统一的控制：

有些函数中，在函数return之前我们经常会进行一些收尾的工作，比如free掉一块函数开始malloc的内存，goto一直都是一个比较简便的方法：

```
int foo()\n{\n    somestruct* ptr = malloc(...);\n\n    dosomething...;\n    if(error)\n    {\n        goto END;\n    }\n\n    dosomething...;\n    if(error)\n    {\n        goto END;\n    }\n    dosomething...;\n\nEND:\n    free(ptr);\n    return 0;\n}
```

由于goto不符合软件工程的结构化，而且有可能使得代码难懂，所以很多人都不倡导使用，那这个时候就可以用do{ }while(0)来进行统一的管理：

```
int foo()\n{\n    somestruct* ptr = malloc(...);\n\n    do{\n        dosomething...;\n        if(error)\n        {\n            break;\n        }\n\n        dosomething...;\n        if(error)\n        {\n            break;\n        }\n        dosomething...;\n    }while(0);\n\n    free(ptr);\n    return 0;\n}
```

这里将函数主体使用do()while(0)包含起来，使用break来代替goto，后续的处理工作在while之后，就能够达到同样的效果。

## 3、避免空宏引起的warning

内核中由于不同架构的限制，很多时候会用到空宏，在编译的时候，空宏会给出warning，为了避免这样的warning，就可以使用do{}while(0)来定义空宏：

```
#define EMPTYMICRO do{}while(0)
```

#### 4、定义一个单独的函数块来实现复杂的操作：

当你的功能很复杂，变量很多你又不愿意增加一个函数的时候，使用do{}while(0);，将你的代码写在里面，里面可以定义变量而不用考虑变量名会同函数之前或者之后的重复。

分享到：

anyShare

#### More from my site



gcc的内联汇编取全局变量地址



关于fork的有意思的两道C语言题目



&&与||的妙用



从编译器角度分析C语言中数组名和指针的区别



C语言可变参数函数取参方法



快速判断一个32位的字中是否存在值为“0”的byte

Tagged with: [C语言](#) • [do{}while\(0\)](#)

#### 6 Responses to “do{...}while(0)的意义和用法”

1.  [编译点滴](#) says:

[October 9, 2012 at 10:16 pm](#)

好文！

[Reply](#)

2.  [nonoob](#) says:

[October 10, 2012 at 12:31 am](#)

2.个人感觉java中的break 的方式对于控制流解决较好

4.如果只使用{.....}而不用do{.....}while(0);也有同样的效果吧？


[Reply](#)

-  [sponge](#) says:

[October 10, 2012 at 10:15 am](#)

@nonoob re，4是有同样的效果，这个看个人喜好吧，毕竟用do while可以随时break跳出

[Reply](#)

3.  [nmsoccer](#) says:

[November 9, 2012 at 11:01 am](#)

这个~~ 碉堡了~~感谢楼主

[Reply](#)

4.  [luobende](#) says:

[July 26, 2013 at 10:40 am](#)

讲得太好了

[Reply](#)

5.  [CJ](#) says:

[February 16, 2014 at 3:38 pm](#)

- 1、避免GOTO而使用do while不可理喻！滥用GOTO才是魔鬼，没有明白为什么goto不可取才会作出这样的决定吧。用一个无厘头的do while只会使程序更加难以理解。不足取！
- 2、有一大段的功能（function）代码为什么不考虑抽象出一个函数（function）？这样得过且过使得程序失去了重构的最佳时机。不提取出一个函数也就算了，竟然还加了一个do while！？同理为什么不能用if true block？此法实不足取！如果要避免函数调用的开销，完全可以交由编译器的inline优化嘛

[Reply](#)

## Leave a Reply

<input type="text"/>	Name (required)
<input type="text"/>	Mail (will not be published) (required)
<input type="text"/>	Website
<div></div>	

**Note:** Commenter is allowed to use '@User+blank' to automatically notify your reply to other commenter. e.g, if ABC is one of commenter of this post, then write '@ABC' (exclude ' ') will automatically send your comment to ABC. Using '@all' to notify all previous commenters. Be sure that the value of User should exactly match with commenter's name (case sensitive).



**云服务器**

**¥0/半年**  
原价：¥280/半年

**免费抢**

**CEFHOST 优易主机**  
**优质香港主机**  
中文 cPanel 面板

  
cPanel Your hosting DirectAdmin  
免费安装 免费搬家  
**Website**

## About this blog!

分享程序猿那些有意思的东西！~~ [RSS feed](#).



## 文章排行

- [跳表\(Skip List\)的介绍以及查找插入删除等操作](#) - 7,385 views
- [Linux内核信号处理机制介绍](#) - 6,437 views
- [do{...}while\(0\)的意义和用法](#) - 6,320 views
- [gdb的基本工作原理是什么？](#) - 4,566 views
- [内存屏障什么的](#) - 3,586 views
- [结构体的sizeof到底多大？](#) - 2,863 views
- [Unix/Linux下的stdout,stdin同stderr](#) - 2,689 views
- [C语言可变参数函数取参方法](#) - 2,376 views
- [为什么C++中空类和空结构体大小为1？](#) - 2,175 views
- [一个百度笔试中的首尾相连的珠子问题解法](#) - 2,155 views

## 最新发表日志

- [快速判断一个32位的字中是否存在值为"0"的byte](#)
- [do{...}while\(0\)的意义和用法](#)
- [gcc的内联汇编取全局变量地址](#)
- [URL相似度计算的思考](#)
- [C语言可变参数函数取参方法](#)

## 最新回复评论

- yang on [留言板](#)
- Michael on [留言板](#)
- Jeanice on [Linux内核信号处理机制介绍](#)
- air jordan shoes for sale authentic on [留言板](#)
- fendi handbags Hot Sale on [留言板](#)

## 分类

- [linux](#)
- [系统结构](#)
- [算法](#)
- [语言学习](#)
- [笔试面试](#)
- [应用问题](#)
- [C语言](#)
- [Gentoo](#)
- [linux内核](#)
- [linux系统](#)
- [其他](#)
- [数据结构](#)

## archives

- [October 2012](#) (2)
- [September 2012](#) (1)
- [April 2012](#) (1)
- [February 2012](#) (2)
- [April 2011](#) (1)
- [March 2011](#) (1)
- [November 2010](#) (2)
- [October 2010](#) (14)
- [September 2010](#) (10)
- [August 2010](#) (4)

## Blogroll

- [dutor](#) 熟读而精思，循序而渐进，厚积而薄发 0
- [sagi](#) 好好学习，天天向上 ~ 0
- [大数据](#) 0
- [王素涵](#) 0
- [编译点滴](#) 研究生同学，做编译器优化的，比较有想法 0

- [靖难's Blog](#) 关注算法,C/C++,Linux,互联网 0

[Go To Top »](#)

## [SpongeLiu的blog](#)



### Pages

- [Home](#)
- [About](#)
- [留言板](#)

### Stay In Touch

- [Site RSS Feed](#)

### More

分享程序猿那些有意思的东西！~~ [RSS feed](#).  

© 2010 Sponge Liu



---

[站长统计](#)