程序 14-7 linux/include/stdarg.h

```
1  #ifndef _STDARG_H
2  #define _STDARG_H
3
4  typedef char *va_list;  // 定义 va_list 是一个字符指针类型。
5
6  /* Amount of space required in an argument list for an arg of type TYPE.
7     TYPE may alternatively be an expression whose type is used.  */
   /* 下面给出了类型为 TYPE 的 arg 参数列表所要求的空间容量。
      TYPE 也可以是使用该类型的一个表达式 */
8
   // 下面这句定义了取整后的 TYPE 类型的字节长度值。是 int 长度(4)的倍数。
9  #define __va_rounded_size(TYPE)  \
10   (((sizeof (TYPE) + sizeof (int) - 1) / sizeof (int)) * sizeof (int))
11
   // 下面这个宏初始化指针 AP，使其指向传给函数的可变参数表的第一个参数。
   // 在第一次调用 va_arg 或 va_end 之前，必须首先调用 va_start 宏。参数 LASTARG 是函数定义
   // 中最右边参数的标识符，即'...'左边的一个标识符。AP 是可变参数表参数指针，LASTARG 是
   // 最后一个指定参数。&(LASTARG) 用于取其地址（即其指针），并且该指针是字符类型。加上
   // LASTARG 的宽度值后 AP 就是可变参数表中第一个参数的指针。该宏没有返回值。
   // 第 17 行上的函数 __builtin_saveregs() 是在 gcc 的库程序 libgcc2.c 中定义的，用于保存
   // 寄存器。 相关说明参见 gcc 手册“Target Description Macros”章中“Implementing the
   // Varargs Macros”小节。
12 #ifndef __sparc__
13 #define va_start(AP, LASTARG)                                          \
14  (AP = ((char *) &(LASTARG) + __va_rounded_size (LASTARG)))
15 #else
16 #define va_start(AP, LASTARG)                                          \
17  (__builtin_saveregs (),                                               \
18   AP = ((char *) &(LASTARG) + __va_rounded_size (LASTARG)))
19 #endif
20
   // 下面该宏用于被调用函数完成一次正常返回。va_end 可以修改 AP 使其在重新调用
   // va_start 之前不能被使用。va_end 必须在 va_arg 读完所有的参数后再被调用。
21 void va_end (va_list);          /* Defined in gnulib */  /* 在 gnulib 中定义 */
22 #define va_end(AP)
23
   // 下面宏用于扩展表达式使其与下一个被传递参数具有相同的类型和值。
   // 对于缺省值，va_arg 可以用字符、无符号字符和浮点类型。在第一次使用 va_arg 时，它返
   // 回表中的第一个参数，后续的每次调用都将返回表中的下一个参数。这是通过先访问 AP，然
   // 后增加其值以指向下一项来实现的。va_arg 使用 TYPE 来完成访问和定位下一项，每调用一
   // 次 va_arg，它就修改 AP 以指示表中的下一参数。
24 #define va_arg(AP, TYPE)                                        \
25  (AP += __va_rounded_size (TYPE),                               \
26   *((TYPE *) (AP - __va_rounded_size (TYPE))))
27
28 #endif /* _STDARG_H */
29
```