# 使用make更新函数库文件

函数库文件也就是对Object文件（程序编译的中间文件）的打包文件。在Unix下，一般是由命令"ar"来完成打包工作。

## 函数库文件的成员

一个函数库文件由多个文件组成。你可以以如下格式指定函数库文件及其组成：

archive(member)

这个不是一个命令，而一个目标和依赖的定义。一般来说，这种用法基本上就是为了"ar"命令来服务的。如：

foolib(hack.o) : hack.o

ar cr foolib hack.o

如果要指定多个member，那就以空格分开，如：

foolib(hack.o kludge.o)

其等价于：

foolib(hack.o) foolib(kludge.o)

你还可以使用Shell的文件通配符来定义，如：

foolib(\*.o)

## 函数库成员的隐含规则

当make搜索一个目标的隐含规则时，一个特殊的特性是，如果这个目标是"a(m)"形式的，其会把目标变成"(m)"。于是，如果我们的成员是 "%.o"的模式定义，并且如果我们使用"make foo.a(bar.o)"的形式调用Makefile时，隐含规则会去找"bar.o"的规则，如果没有定义bar.o的规则，那么内建隐含规则生效，make会去找bar.c文件来生成bar.o，如果找得到的话，make执行的命令大致如下：

cc -c bar.c -o bar.o

ar r foo.a bar.o

rm -f bar.o

还有一个变量要注意的是"$%"，这是专属函数库文件的自动化变量，有关其说明请参见"自动化变量"一节。

## 函数库文件的后缀规则

你可以使用"后缀规则"和"隐含规则"来生成函数库打包文件，如：

.c.a:

$(CC) $(CFLAGS) $(CPPFLAGS) -c $< -o $\*.o

$(AR) r $@ $\*.o

$(RM) $\*.o

其等效于：

(%.o) : %.c

$(CC) $(CFLAGS) $(CPPFLAGS) -c $< -o $\*.o

$(AR) r $@ $\*.o

$(RM) $\*.o

## 注意事项

在进行函数库打包文件生成时，请小心使用make的并行机制（"-j"参数）。如果多个ar命令在同一时间运行在同一个函数库打包文件上，就很有可以损坏这个函数库文件。所以，在make未来的版本中，应该提供一种机制来避免并行操作发生在函数打包文件上。

但就目前而言，你还是应该不要尽量不要使用"-j"参数。

|  |  |
| --- | --- |
|  |  |
|  |