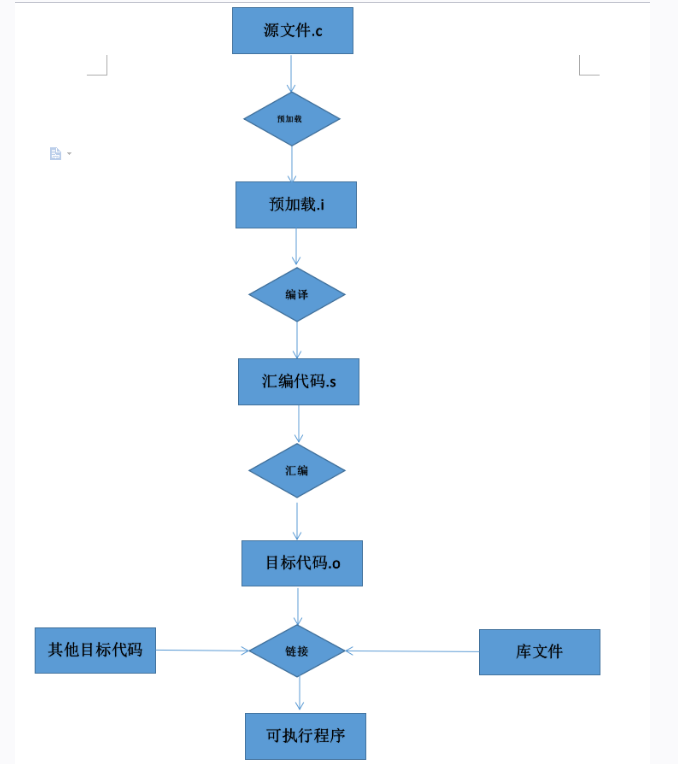
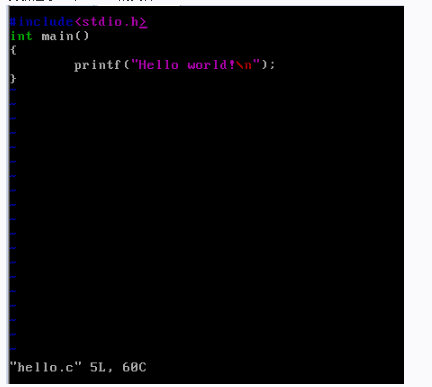
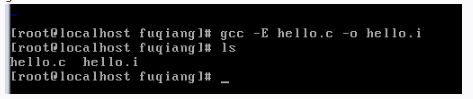
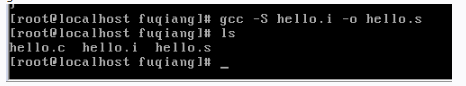
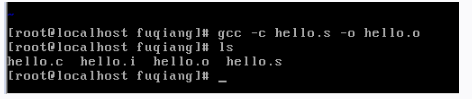
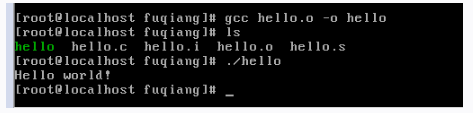
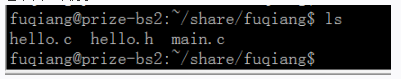
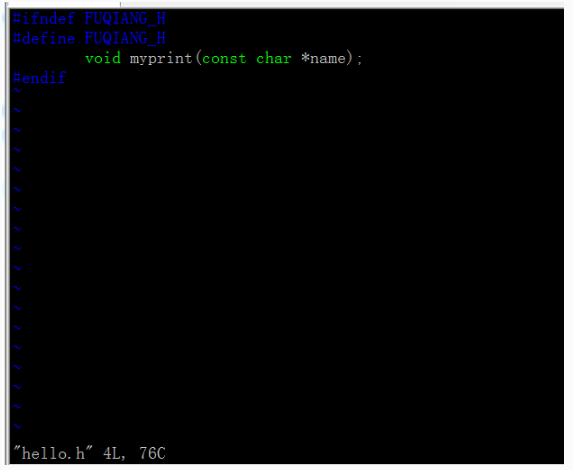
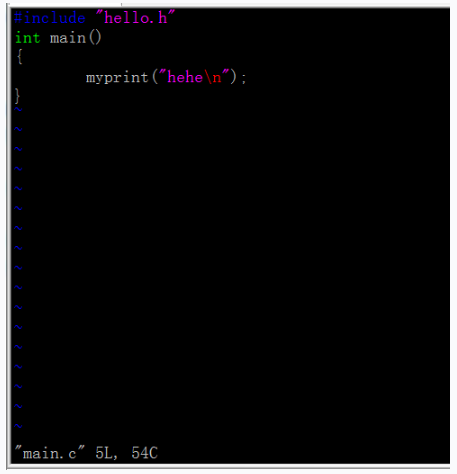
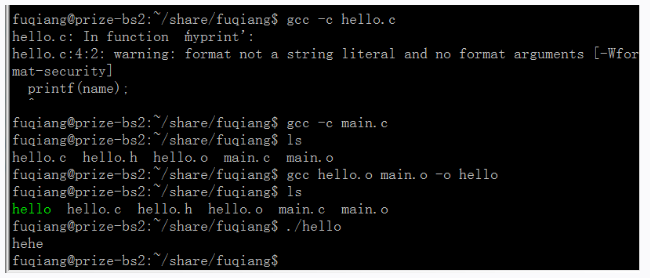
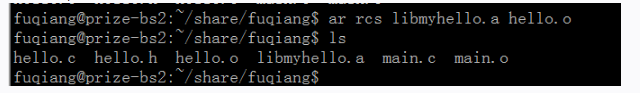
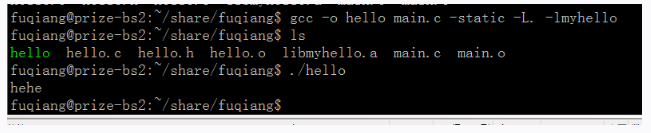
<https://www.linuxidc.com/Linux/2016-11/137253.htm>

一 gcc编译过程   
我们知道gcc是一个强大的编译器，很多Linux下的GNU工具都是用C语言写的，并且用gcc编译的，那么gcc的编译过程是怎样的呢，先来看一个总的流程图，我自己简单画的，凑合着看

  
  
1首先是源文件经过预加载变成了.i结尾的文件，可以通过-E这个参数来生成这个中间文件，这里主要是把一些include的头文件和一些宏定义，放到源文件中。   
2从预加载的文件经过编译就会变成汇编语言的文件，这一步可以通过-S这个参数来生成这个中间文件   
3从汇编语言的文件通过汇编，就会变成目标代码.o的文件，这一步可以通过-C这个参数来生成这个中间文件   
4最后经过链接，生成最终的可执行文件   
可能这样说比较难懂，我们通过一个例子来说明下：   
我新建了一个hello.c的文件   
   
然后我先生成.i结尾的预加载文件   
gcc -E hello.c -o hello.i   
   
我们看到这个文件生成了，我们看一下这个文件的内容   
  
大家会发现这个文件的内容非常多，但是最后是我们本来的代码，上面的代码都是头文件和一些宏的内容，全加载进来了   
下面我们通过编译生成汇编文件.s   
gcc -S hello.i -o hello.s   
   
然后我们看一下这个hello.s

  
看到了我们熟悉的汇编语言   
我们继续，经过汇编器，生成.o的目标文件   
gcc -c hello.s -o hello.o   
   
依然来看下内容   
  
看到了一堆二进制的感觉   
最后通过链接器，生成可执行文件   
gcc hello.o -o hello   
然后执行   
   
当然如果你不想做这么多步骤，直接gcc hello.c -o hello即可完成上面所有的步骤了。

二 静态链接库和动态链接库   
静态链接库就是在程序编译的时候就被加载进来，这样的可执行文件会比较大一些，还不能共享   
动态链接库是在程序执行的时候加载，可共享   
看下面一个例子   
   
现在有这么三个文件   
   
   
   
如果直接编译hello.c肯定会报错，因为这里没有main函数，那么我们该怎么处理呢   
1 都编译成.o文件，然后链接，生成可执行文件   


2 利用静态链接库   
在linux下，库文件一般放在/usr/lib和/lib下，   
静态库的名字一般为libxxxx.a，其中xxxx是该lib的名称   
动态库的名字一般为libxxxx.so.major.minor，xxxx是该lib的名称，major是主版本号， minor是副版本号   
静态库的后缀是.a，它的产生分两步   
由源文件编译生成一堆.o，每个.o里都包含这个编译单元的符号表   
ar命令将很多.o转换成.a，成为静态库   
ar rcs libmyhello.a hello.o   
   
注意，gcc会在静态库名前加上前缀lib，然后追加扩展名.a得到的静态库文件名来查找静态库文件,因此，我们在写需要连接的库时，只写名字就可以，如libmyhello.a的库，只写:-lmyhello   
  
gcc -o hello main.c -static -L. -lmyhello   
-static代表使用静态链接库，-L.代表静态链接库搜索路径 .代表当前路径

3 动态链接库   
gcc -shared -fPIC -c hello.c   
gcc -shared -fPIC -o libmyhello.so hello.o

-share代表是动态链接库   
-fPIC命令行标记告诉GCC产生的代码不要包含对函数和变量具体内存位置的引用，这是因为现在还无法知道使用该消息代码的应用程序会将它连接到哪一段内存地址空间。这样编译出的hello.o可以被用于建立共享链接库   
最后gcc -o hello main.c -L. -lmyhello生成hello可执行文件，注意执行的时候可能会报错，说找不到这个   
libmyhello.so文件，如果放在/lib或者/usr/lib下，那么默认就能找到，如果放在其他目录下，需要编辑/etc/ld.so.conf文件，加入库文件所在目录的路径，然后   
运行ldconfig 目录名字，该命令会重建/etc/ld.so.cache文件即可。

好了，Linux下gcc编译过程，静态链接库和动态链接库就总结到这里，如有问题，欢迎指正，谢谢。

**本文永久更新链接地址**：[http://www.linuxidc.com/Linux/2016-11/137253.htm](https://www.linuxidc.com/Linux/2016-11/137253.htm)