# 35. 外部应用程序/库的 Makefile

外部的应用程序或库必须包含RTE\_SDK指定的位于mk目录中的Makefiles文件。 这些Makefiles包括：

* ${RTE\_SDK}/mk/rte.extapp.mk: 构建一个应用程序。
* ${RTE\_SDK}/mk/rte.extlib.mk: 构建一个静态库。
* ${RTE\_SDK}/mk/rte.extobj.mk: 购件一个目标文件。

## 35.1. 前提

必须定义以下变量：

* ${RTE\_SDK}: 指向DPDK根目录。
* ${RTE\_TARGET}: 指向用于编译的目标编译器(如x86\_64-native-linuxapp-gcc)。

## 35.2. 构建 Targets

支持构建target时指定输出文件的目录，使用 O=mybuilddir 选项。 这是可选的，默认的输出目录是build。

* all, “nothing” (仅make)  
  编译应用程序或库到指定的输出目录中。  
  例如：

make O=mybuild

* clean  
  清除make操作产生的所有目标文件。  
  例如：

make clean O=mybuild

## 35.3. Help Targets

* help  
  显示帮助信息。

## 35.4. 其他有用的命令行变量

以下变量可以在命令行中指定：

* S=  
  指定源文件的位置。默认情况下是当前目录。
* M=  
  指定需要被调用的Makefile。默认情况下使用 $(S)/Makefile。
* V=  
  使能详细编译（显示完全编译命令及一些中间命令过程）。
* D=  
  启用依赖关系调试。提供了一些有用的信息。
* EXTRA\_CFLAGS=, EXTRA\_LDFLAGS=, EXTRA\_ASFLAGS=, EXTRA\_CPPFLAGS=  
  添加的编译、连接或汇编标志。
* CROSS=  
  指定一个交叉工具链，该前缀将作为所有gcc/binutils应用程序的前缀。只有在gcc下才起作用。

## 35.5. 从其他目录中编译

通过指定输出和源目录，可以从另一个目录运行Makefile。  
例如：

export RTE\_SDK=/path/to/DPDKexport RTE\_TARGET=x86\_64-native-linuxapp-icc

make -f /path/to/my\_app/Makefile S=/path/to/my\_app O=/path/to/build\_dir