



cmake：交叉编译



大川搬砖
专注嵌入式开发, rtos, linux c, cmake, 工具。

58 人赞同了该文章

概念

我们在 PC 上进行开发时，需要使用编译器和链接器生成能够在我们机器上运行的可执行程序。但是当涉及到嵌入式开发时，情况就不同了。因为嵌入式设备的资源（CPU、RAM等）无法和 PC 相比，在设备上构建编译系统很麻烦或者根本不可能构建。因此通常做法是在 PC 上使用交叉编译工具链生成能够在嵌入式设备运行的可执行程序，然后再将程序放到设备中去执行。

此为交叉编译。一般称 PC 为主机，嵌入式设备为目标机。

编写脚本

使用 `cmake` 进行交叉编译，只需几条命令即可。

```
set(CMAKE_SYSTEM_NAME Linux)

set(TOOLCHAIN_PATH /opt/gcc-arm-linux-gnueabi)
set(CMAKE_C_COMPILER ${tools}/bin/arm-linux-gnueabi-gcc)
set(CMAKE_CXX_COMPILER ${tools}/bin/arm-linux-gnueabi-g++)
```

`set(CMAKE_SYSTEM_NAME Linux)`：该指令必须存在，其目的是设置目标机使用的操作系统名称，支持 `Linux`，`QNX`，`WindowsCE`，`Android` 等。如果没有操作系统，那么就写 `Generic`。执行该指令后，`cmake` 变量——`CMAKE_CROSSCOMPILING` 会自动被设置为 `TRUE`，此时 `cmake` 就会“知道”现在执行的是交叉编译；

由于 `cmake` 无法自动获取目标机器信息，因此需要显式指明编译工具。

`CMAKE_C_COMPILER`：设置 C 编译器；

`CMAKE_CXX_COMPILER`：设置 C++ 编译器

使用方式

1. 将上述 4 条指令保存在 `xxx.cmake` 文件中，比如 `CrossCompile.cmake`；
2. 使用 `cmake -DCMAKE_TOOLCHAIN_FILE= path/CrossCompile.cmake src-path` 构建编译系统；
3. 执行 `make` 指令；

注意：上述命令必须写入脚本中，使用 `-DCMAKE_TOOLCHAIN_FILE=xxx.cmake` 的方式使用。不能直接写入 `CMakeLists.txt` 或使用 `include(xx.cmake)`。

测试例程

目录结构：

```
sdc@ubuntu:~/cross-compile$ tree .
.
├── build
├── CMakeLists.txt
├── CrossCompile.cmake
└── main.c

1 directory, 3 files
```

结果：

```
sdc@ubuntu:~/cross-compile/build$ cmake -DCMAKE_TOOLCHAIN_FILE=./CrossCompile.cmake ..
-- The C compiler identification is GNU 4.9.4
-- Check for working C compiler: /opt/gcc-arm-linux-gnueabi/bin/arm-linux-gnueabi-gcc
-- Check for working C compiler: /opt/gcc-arm-linux-gnueabi/bin/arm-linux-gnueabi-gcc -- works
-- Detecting C compiler ABI info
-- Detecting C compiler ABI info - done
-- Detecting C compile features
-- Detecting C compile features - done
-- Configuring done
-- Generating done
-- Build files have been written to: /home/sdc/cross-compile/build
sdc@ubuntu:~/cross-compile/build$ make
Scanning dependencies of target c-test
[ 50%] Building C object CMakeFiles/c-test.dir/main.c.o
[100%] Linking C executable c-test
[100%] Built target c-test
```

知乎 @大川搬砖

其他说明

通常，我们在开发时，需要使用系统库或第三方库的功能，在生成可执行文件时，将其进行链接。`cmake` 提供了 `FIND_PROGRAM()`，`FIND_LIBRARY()`，`FIND_FILE()`，`FIND_PATH()` 和 `FIND_PACKAGE()` 实现相应的查找功能。如果我们在进行交叉编译时使用了上述指令，那么并不能生成可执行文件。因为默认情况下，上述指令查找的是主机上的相关文件，其并不适用于目标机器。还好，`cmake` 为我们提供了相应的变量：

`CMAKE_FIND_ROOT_PATH`：设置其值为一系列的目录（`set(CMAKE_FIND_ROOT_PATH path1 path2 path3 ...)`），这样在执行 `FIND_XXX()` 指令时就会从这一系列的目录中进行查找。

跟随该变量的有下述 3 个变量，它们的值为 `NEVER`、`ONLY` 或 `BOTH`：

`CMAKE_FIND_ROOT_PATH_MODE_PROGRAM`：如果设置为 `NEVER`，那么 `CMAKE_FIND_ROOT_PATH` 就

不会对 `FIND_PROGRAM()` 产生影响, `FIND_PROGRAM()` 不会在 `CMAKE_FIND_ROOT_PATH` 指定的目录中寻找; 如果设置为 `ONLY`, 那么 `FIND_PROGRAM()` 只会从 `CMAKE_FIND_ROOT_PATH` 指定的目录中寻找; 如果设置为 `BOTH`, 那么 `FIND_PROGRAM()` 会优先从 `CMAKE_FIND_ROOT_PATH` 指定的目录中寻找, 再从默认的目录中寻找。

因为 `FIND_PROGRAM()` 大部分情况下用于寻找可执行程序, 给后续的 `EXECUTE_PROCESS()` 或 `ADD_CUSTOM_COMMAND()` 指令使用。并且, 只有主机在生成编译文件时使用该可执行程序。因此通常设置 `CMAKE_FIND_ROOT_PATH_MODE_PROGRAM` 为 `NEVER` (`set(CMAKE_FIND_ROOT_PATH_MODE_PROGRAM NEVER)`);

`CMAKE_FIND_ROOT_PATH_MODE_LIBRARY`: 由于在进行交叉编译, 所以只能使用 `FIND_LIBRARY()` 查找符合目标机器的库文件, 因此设置该变量值为 `ONLY` (`set(CMAKE_FIND_ROOT_PATH_MODE_LIBRARY ONLY)`), 表示只从 `CMAKE_FIND_ROOT_PATH` 指定的目录中查找;

`CMAKE_FIND_ROOT_PATH_MODE_INCLUDE`: 同上, 将其值设置为 `ONLY`。

参考 (请仔细阅读)

[gitlab.kitware.com/cmake](https://gitlab.kitware.com/cmake/cmake) ...

[blog.kitware.com/cross-](https://blog.kitware.com/cross-compiling-with-cmake/) ...

发布于 2019-12-31 13:20


[CMake](#) [编译](#) [交叉编译 \(编译\)](#)

写下你的评论...



还没有评论, 发表第一个评论吧

文章被以下专栏收录

 cmake 实践和学习

推荐阅读

cmake编译相关的一些总结

%cd% 表示当前命令行的路径 /lib 是内核级的: 包含被 /bin/ 和 /sbin/ 中的程序使用的库文件; /usr/lib是系统级的: 目录 /usr/lib/ 中含有更多用于用户程序的库文件; /usr/local/lib是用户...

zc88



Go语言涉及CGO的交叉编译 (跨平台编译)解决办法

翱翔的赖思

发表于Let's...

Go 交叉编译 (跨平台编译)

Go 交叉编译Golang 支持交叉编译, 在一个平台上生成另一个平台的可执行程序 — Mac 下编译 Linux 和 Windows 64位可执行程序 CGO_ENABLED=0 GOOS=linux GOARCH=amd64 go build...

刘sir

发表于Pytho...

程序详细编译过程 (预处理、编译、汇编、链接)

程序编译的整体流程以下面这个简单的函数为例 #include<stdio.h> int main() { printf("hello, world "); return 0; }hello程序的生命周期是从一个 源程序 (或者说源...

YaoJ1...

发表于UE4



× 登录即可查看 超5亿 专业优质内容
超 5 千万创作者的优质提问、专业回答、深度文章和精彩视频尽在知乎。

[立即登录/注册](#)

▲ 赞同 58



● 添加评论

🔗 分享

❤️ 喜欢

★ 收藏

💡 申请转载

...