


≡ Disconnected Systems

Arduino and CMake

The arduino is a great platform for embedded development and the ide it comes with is a good starting point. However, it lacks many features that make it a brilliant ide. Most notably it only supports single source programs and doesn't really allow you to write libraries.

So where do you go after you have outgrown the arduino ide? For me cmake was the next logical step and it turns out that someone else has already written a cmake module to build and upload programs to the arduino.

Installation

You can download the module from [here](#)  and installing it is as simple as copying the cmake folder to the root of your project. You should be able to install it to the systems cmake module path, but this makes it harder to share your code as others will then also need to install the module.

Usage

Very few adjustments to your projects are needed, just change the extension of the script from .ino to .cpp and add `#include "Arduino.h"` to the top of the file. You might also need to add forward declarations of any functions you use as according to the standard c/c++ rules.

Then create a CMakeLists.txt file with the following contents:

```
set(CMAKE_TOOLCHAIN_FILE cmake/ArduinoToolchain.cmake)

cmake_minimum_required(VERSION 2.8)

project(Project_Name C CXX)

set(ARDUINO_DEFAULT_BOARD uno) # Default Board ID
```

Disconnected Systems

```
generate_arduino_library(exe_name -srcs -script.cpp)
```

changing the `Project_Name` , board type, port, `exe_name` and `script.cpp` to the appropriate values for your project.

Then to build the program simply run

```
mkdir build
cd build
cmake ..
make
```

sh

and to upload `exe_name` to the arduino run

```
make exe_name-upload
```

sh

Libraries

Out of the box the cmake module will automatically pick up and compile any simple libraries that are in the libraries directory of your project so most of the time you don't need to worry about them. But it also support more complex situation via the `generate_arduino_library` command:

```
generate_arduino_library(name
    [BOARD board_id]
    [SRCS src1 src2 ... srcN]
    [HDRS hdr1 hdr2 ... hdrN]
    [LIBS lib1 lib2 ... libN]
    [NO_AUTOLIBS])
```

c

Which you can find out more about by [reading the documentation](#) included with the cmake module.

Disconnected Systems

Conclusion

Now you should be able to code your arduino projects in your favourite ide as long as it has cmake support (or support in cmake for generating project files for it) and can start expanding your project beyond simple scripts.

Finally make sure you read the [documentation](#) for the module, as it explains how to use its advanced features far better than I can.