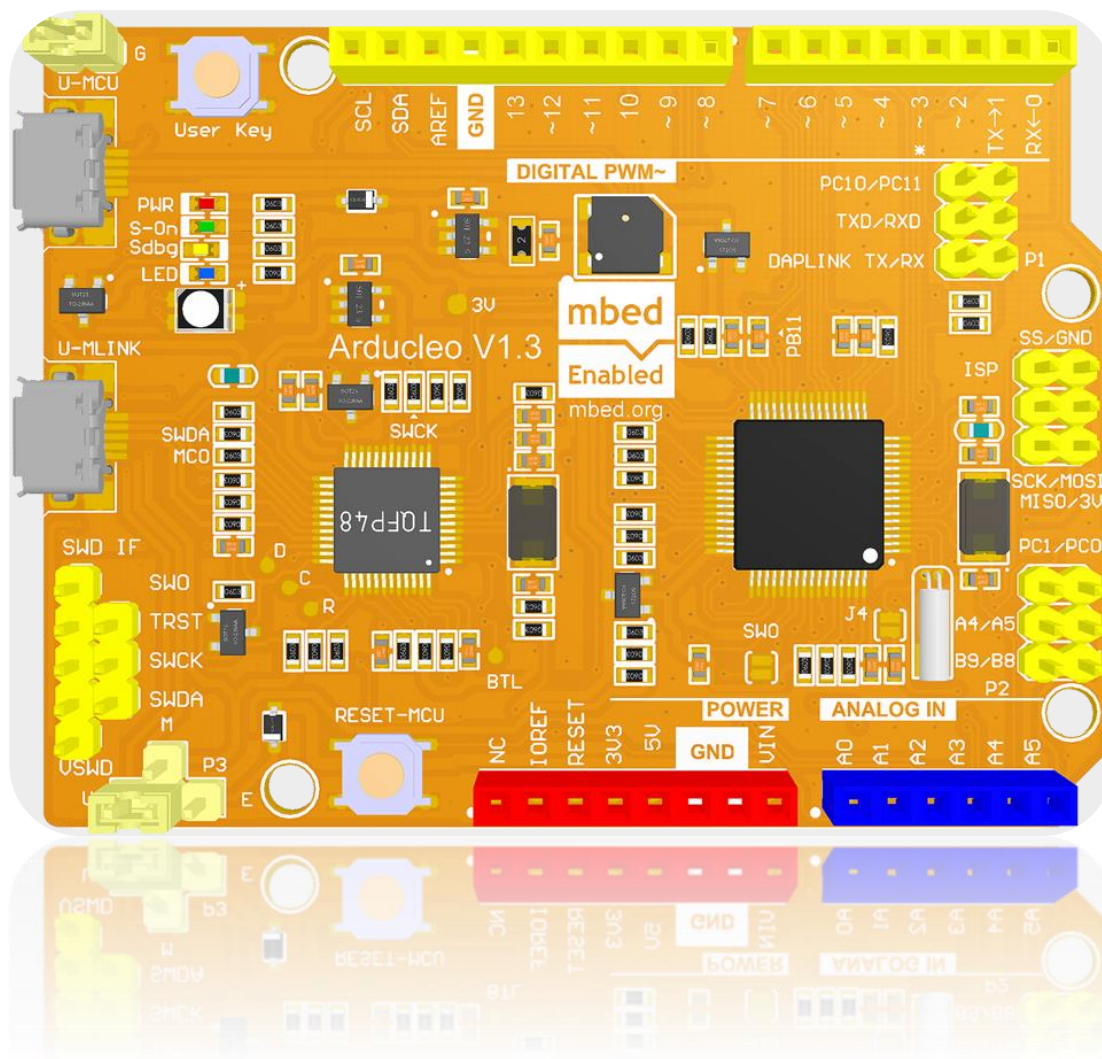


2016

# Arducleo 开发板 用户手册



在使用它的时候，可以直接采用 mbed 的官方环境、工具链以及 ST 官方的软件库进行快速的原型开发。当然也可以使用 Arduino™ 接口的扩展板进行硬件平台的搭建。



使用 LQFP64 封装的 STM32 微控制器；

Arduino Uno Revision 3 接口；

支持 ARM mbed 开发环境；

板载 ARM DAPLink 调试单元 ( SWD + MSD + CDC )；

5V 电源选择：

- micro USB 供电；
- 外部输入电源 VIN (5V only) 来自 Arduino 接口，或者通过第三 micro usb 接口；
- DAPLink 接口供电；

LED 指示灯

- 电源指示灯 PWR，DAPLink 工作指示灯 ( 2 只 )，D13 控制的用户 led；
- 三色 LED (PC6, PC8, PC9 控制)

一个用户按键，一个目标芯片复位按键：

板载无源蜂鸣器

板载晶振：

- 8MHz 晶体振荡器，32.768KHz 晶体振荡器 ( 选配 )；

板载 USB 接口

ICSP 接口 ( 兼容 Arduino LEONARDO 扩展板 )，作为 SPI 接口使用；

支持广泛的集成开发环境 ( IDEs )，包括 IAR，KEIL，基于 GCC 的 IDEs；

## 1.3 与 Nucleo 的对比

项目	Arducléo	Nucleo	备注
Arduino 兼容性	兼容 UNO 及 Leonardo	兼容 UNO	Arducléo 默认兼容 UNO，可通过跳线设为兼容 Leonardo
Arduino ICSP	有	无	带 ICSP 接口，才能支持带有该接口的扩展板
USB 接口	有	无	支持 mbed USB client api
串口调试输出	支持	不支持	NUCLEO 上 ST-LINK 的虚拟串口与 D0/D1 固定连接，链接外设就不能输出调试信息。
8MHz 晶振	有	无	使用外部 8MHz 晶振，定时会更准
32.768KHz 晶振	可选支持	无	有该晶振才支持 RTC 功能
LED 支持 pwm	支持	不支持	PWM 驱动 led, 可以调节 led 的亮度
板载蜂鸣器	有	无	学习 PWM 的利器

## 1.4 微控制器特性

STM32F103RBT6 LQFP64 封装

ARM®32-bit Cortex®-M3 CPU

72 MHz max CPU frequency

VDD from 2.0 V to 3.6 V

128 KB Flash

20 KB SRAM

GPIO (51) 具备外部中断功能

12-bit ADC (2) with 16 channels

RTC

Timers (4)

I2C (2)

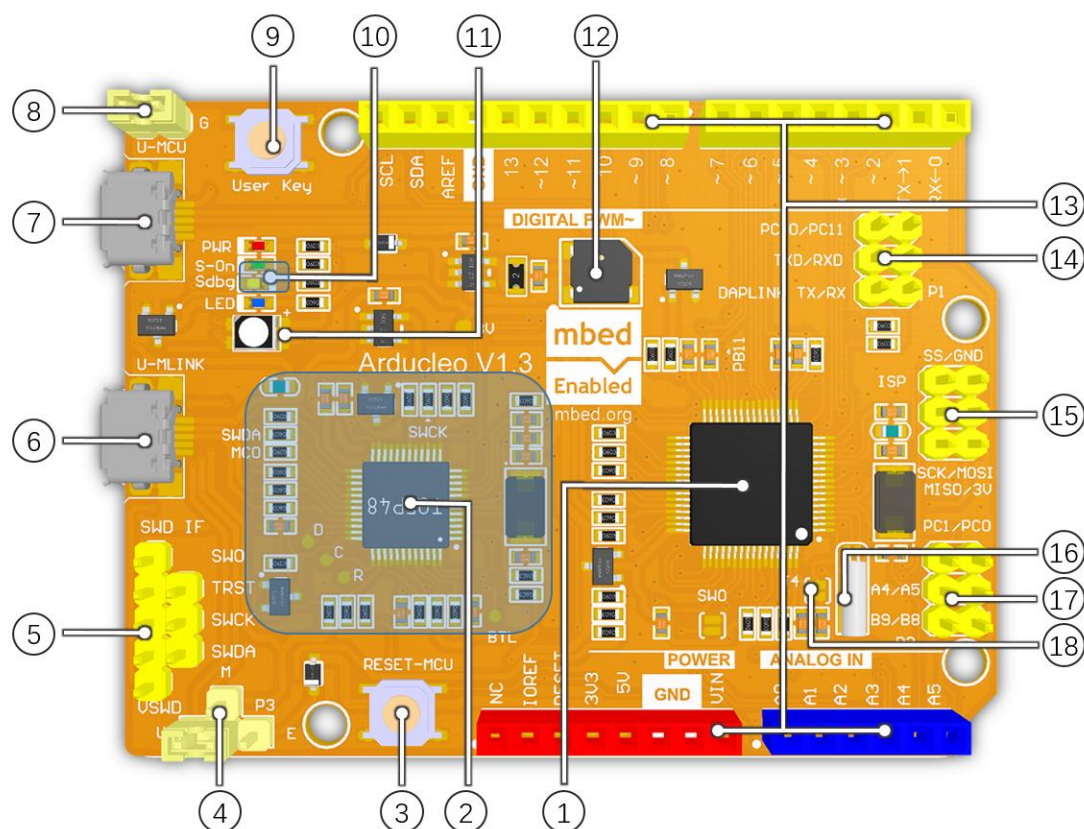
USART (3)

SPI (2)

USB 2.0 full-speed

CAN

## 1.5 图解



### 1. 目标芯片 STM32F103RBT6

内 核：ARM® 32-bit Cortex™-M3  
 工作频率：72MHz  
 工作电压：2.0V-3.6V  
 封 装：LQFP64  
 存储资源：128kB Flash，20kB SRAM  
 接口资源：2 x SPI，3 x USART，  
 2 x I2C，1 x CAN，1 x USB  
 模数转换：2 x AD(12 位, 16 通道)

### 2. DAPLink SWD 调试器

### 3. 目标芯片复位按钮

### 4. 目标芯片供电选择跳线

### 5. DAPLink SWD 调试接口

### 6. DAPLink SWD MicroUSB 接口

### 7. 目标芯片 MicroUSB 接口

### 8. BOOT1 短接冒 (可选连接)

### 9. 用户输入按键

### 10. DAPLink LED 指示灯

### 11. 三色 LED

### 12. 蜂鸣器

### 13. Arduino 兼容接口

### 14. PA2/PA3 与 DAPLink TX/RX 连接 (默认)

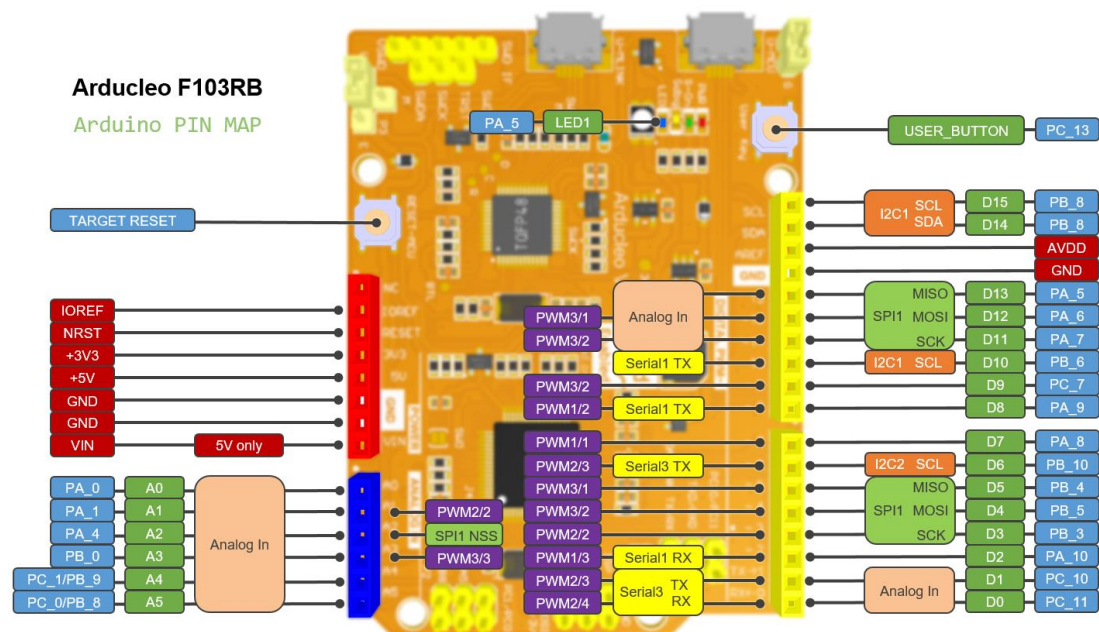
### 15. Arduino ICSP 接口

### 16. 32.768 KHz 晶振 (可选)

### 17. A4/A5 选择跳线

### 18. J4-VBAT VDD 短接 (默认)

## 1.6 接口



提示：在 mbed 代码中，可以直接使用图中蓝底白字，绿底白字的标号（比如 PA\_4, PB\_5, A0, D14, LED1...）。其它的标号则只是接口的辅助信息。当然，代码中，还可以使用如下关键字：

```
SERIAL_TX=PA_2  I2C_SCL=PB_8  SPI_MOSI=PA_7  PWM_OUT=PB_3
SERIAL_RX=PA_3  I2C_SDA=PB_9  SPI_MISO=PA_6
                        SPI_SCK =PA_5
                        SPI_CS  =PB_6
```

**注意：**D10/PB\_6 在 mbed官方代码中是软 SPI\_SS。在 SPI Slave 的功能下，SPI1 的 NSS 信号必须使用 PA\_4 才能工作正常。



## 2. 硬件配置

### 2.1 板载的 DAPLink 仿真器

DAPLink 仿真器 (1.5 ②) 及其调试接口 (1.5 ⑤) 硬件上与 STLink v2.1 是一致的, 固件则是基于 ARM DAPLink 代码修改而来。所以, 支持 mbed 所需的三个功能接口, HID、CDC、MSD:

- 软件控制的 USB 枚举
- SWD 调试, 支持 SWO 调试输出
- USB 虚拟串口, 方便串口调试输出, 通讯
- USB 虚拟磁盘, 支持 STM32F103RB 固件拖放更新
- 300mA 的供电能力

DAPLink 硬件电路中, 还有两颗 LED 指示灯, 见 1.5 图中的左上角 S-on (电源指示), Sdbg(调试通讯指示)。

如果需要如 Nucleo 相同的时钟输出功能, 请焊接 MCO 标号的焊盘即可。

如果需要调试外部 ARM Cortex 设备, 你可以在 SWDA, SWCK, SWO 标号引出 SWD 所需的控制线, 以及 Arduino 接口中的 RESET 信号。

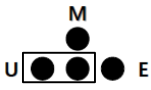
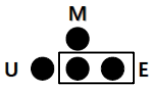
在需要升级或者切换 DAPLink 到 mLink bootloader 模式下, 请将 BTL 标号的焊盘接地, 然后插拔与 PC 端的 USB 连线即可。详情参考 mLink 硬件说明文档。

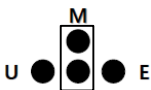
提示: 如需使用 SWD 调试外部的 MCU, 请断开 (1.5 ⑤) 的 “TRST/SWCK/SWDA”, 然后将左侧的对应 PIN 引出杜邦线到目标板即可。VSWD PIN 则提供 SWD 的 3V 供电。

当然, 如需使用外部 SWD 仿真器, 将调试信号线接入右侧对应 PIN 即可。

### 2.2 供电选择

Arducleo 通过 1.5 ④ 跳线进行供电选择, 可以选择下面三种电源之一:

	使用 MicroUSB (1.5 ⑦) 5V 供电, 作为从设备, 至少 100mA
	使用外部 VIN 的 5V 供电

	使用 DAPLink 的 MicroUSB 供电 ( 1.5 ⑥ ), 最大 300mA
---	--

**注意：**外部电源请使用 5V 输入，推荐使用 Android 手机充电器。此外，目标板对外输出的 3V 引脚，最大提供 500mA 电流。

## 2.3 LED 指示灯

### 2.3.1 目标板电源指示

在 1.5 图中，左上角 PWR 标号处，即为电源指示灯。只要目标芯片获得电力供给，即常亮。

### 2.3.2 DAPLink 工作指示灯

在 1.5 图中，左上角 S-on, Sdbg 表好处两颗 led 即为 DAPLink 工作指示灯。

### 2.3.3 用户控制 LED

在 1.5 图中，左上角 LED 标号处，即为用户代码可控 LED。默认连接于 PA\_5 引脚。由于 PA\_5 不支持 PWM 输出，所以，开发板额外提供一个跳线 ( 1.5 ⑩，需要手工焊接 )，可以将其连接到 PC\_8 上。

## 2.4 按键

### 2.4.1 用户按键

用户按键 ( 1.5 ⑨ ) 连接到 PC\_13 引脚。

### 2.4.2 目标芯片复位按键

复位按键 ( 1.5 ③ ) 连接到目标芯片的 NRST 引脚，可用于手工复位目标芯片。



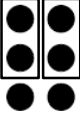
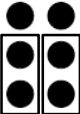
## 2.5 蜂鸣器

Arducléo 板载了一颗无源蜂鸣器，连接于 PB\_1 引脚，支持 PWM 输出。

## 2.6 串口通讯

目标芯片的 PA\_2/PA\_3 默认连接到了 DAPLink 的 RX/TX 引脚，在 mbed 环境下作为与 PC 间的调试输出。可以通过 JP3/JP2 断开 (1.5 (11))。

另外，可以通过跳线 (1.5 (13)) 控制 Arudino 接口的 D0, D1 的引出管脚。

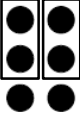
	D0 – PC_11 D1 – PC_10 默认。即，将 PA2/PA3 作为 pc 端串口输出；PC11/PC10 作串口通讯。
	D0 – DAPLink RX D1 – DAPLink TX 可以用 DAPLink 作为串口工具

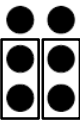
## 2.7 ICSP (SPI) 通讯

为方便使用，将目标芯片的 SPI1 (D10...D13) 单独引出，且位置和引脚序与 Arudino UNO 的相同。见 1.5 (15)。

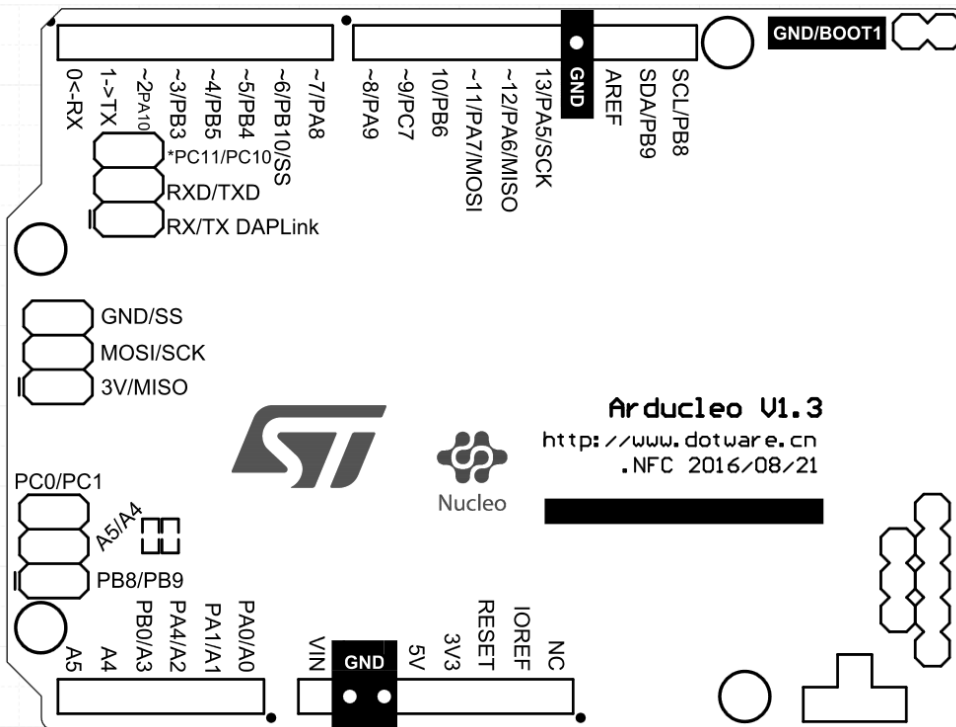
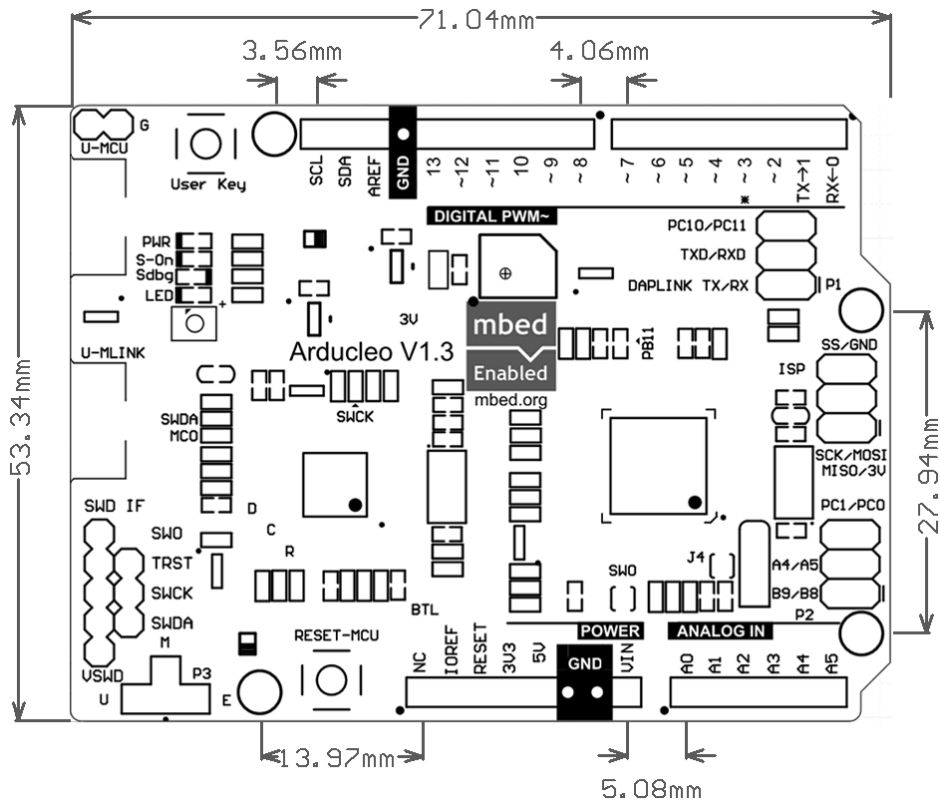
## 2.8 I2C 通讯

I2C 默认引出在 D14/D15 (默认已 10K 电阻上拉) 上，见板子的丝印。另外，还可以利用跳线，将 A4/A5 转为 I2C 引脚。

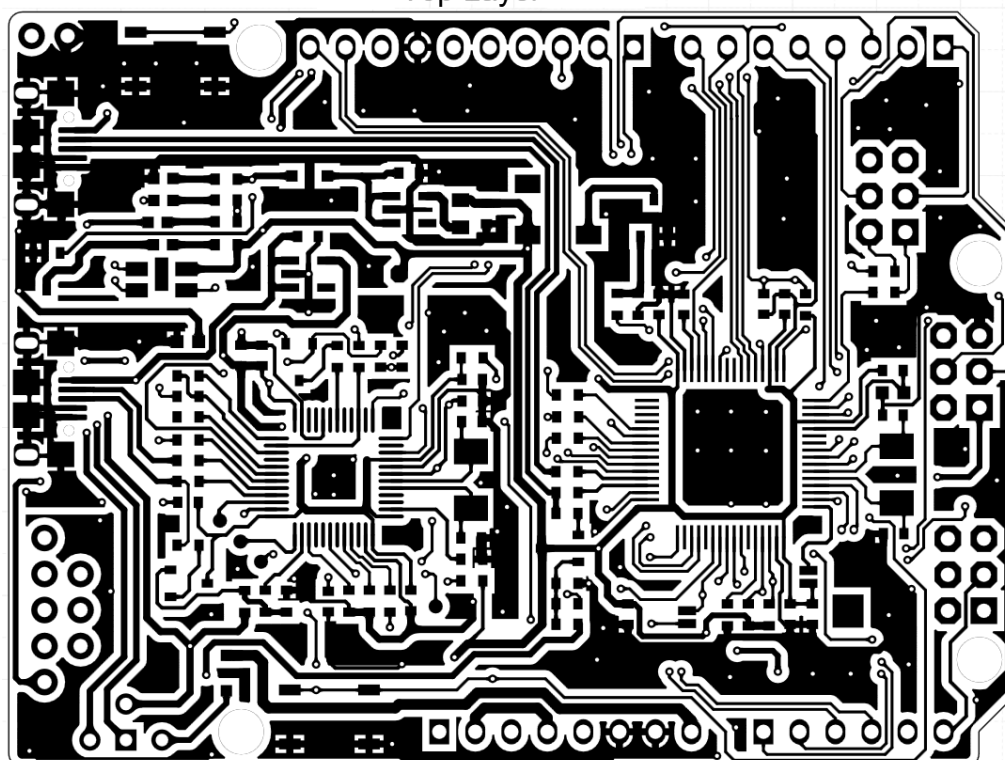
	A4 – PC_1 D5 – PC_0 默认。即，将 A4/A5 做 Analog IN。
---	---

	<p>A4 – PB9</p> <p>A5 – PB8</p> <p>将 A4/A5 作为 I2C 引脚，直接连接到 D15/D14 上。</p>
---	---

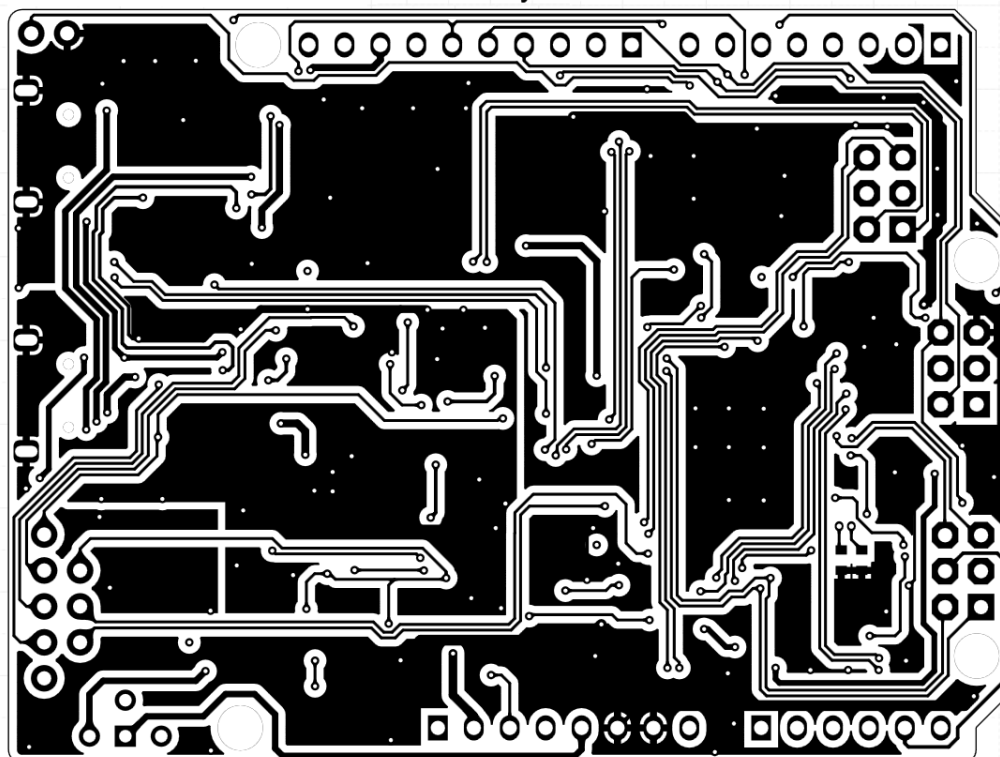
### 3. 工程图



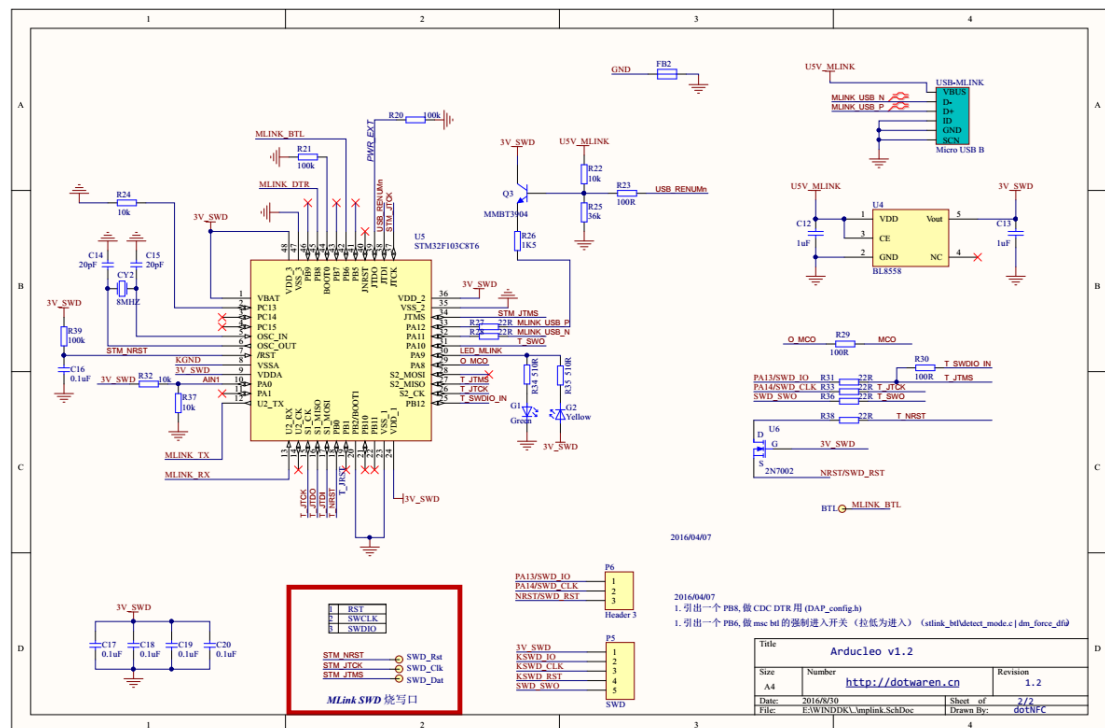
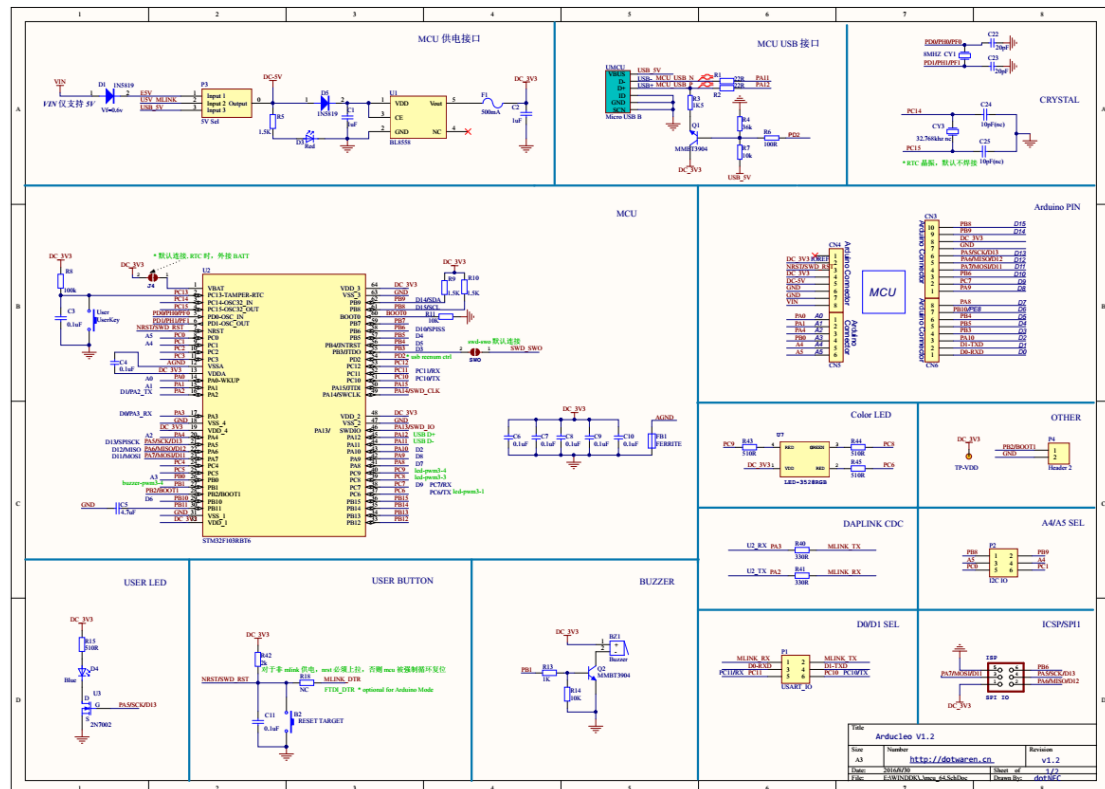
Top Layer



Bottom Layer



## 4. 电路原理图



## 5. ARM mbed 平台

### 5.1 概述

ARM® mbed™ IoT 设备平台提供了操作系统、云服务、工具和开发人员体系，以便能够大规模创建和部署基于标准的商业 IoT 解决方案。其官方网址为 <https://www.mbed.com/zh-cn/>。

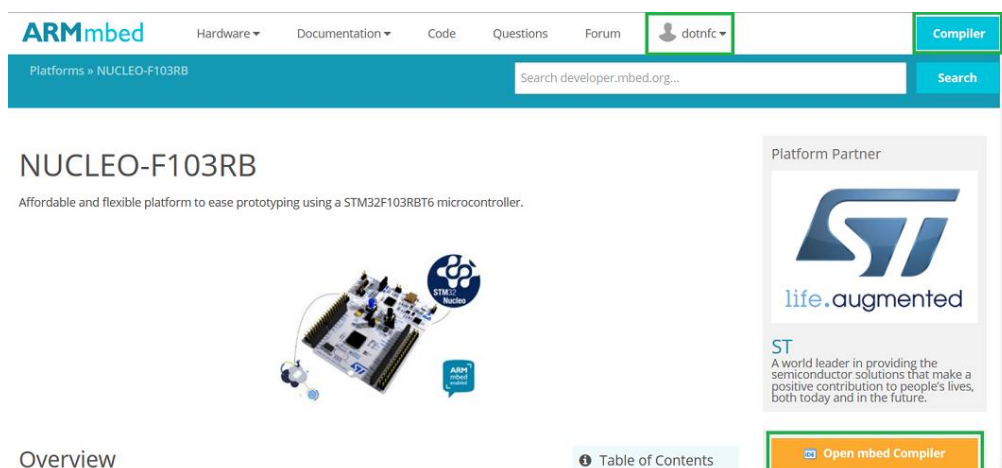
作为开发者，我们需要进入面向开发者在线 IDE 首页，其地址为：

<http://developer.mbed.org/> 【首页】

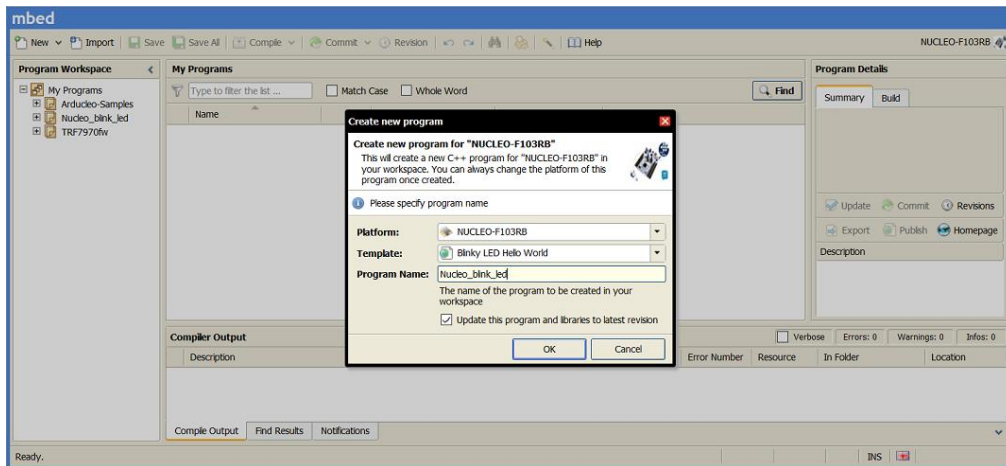
<https://developer.mbed.org/platforms/ST-Nucleo-F103RB/> 【目标平台】

当然，如果你接入 DAPLink 到 PC 机后，在资源管理器中打开对于的虚拟磁盘。位于其根目录下有一个 mbed.html，双击打开就可以直接前往“目标平台”页面；如果之前没有账户，建议你按照向导的指示，新建一个账户。

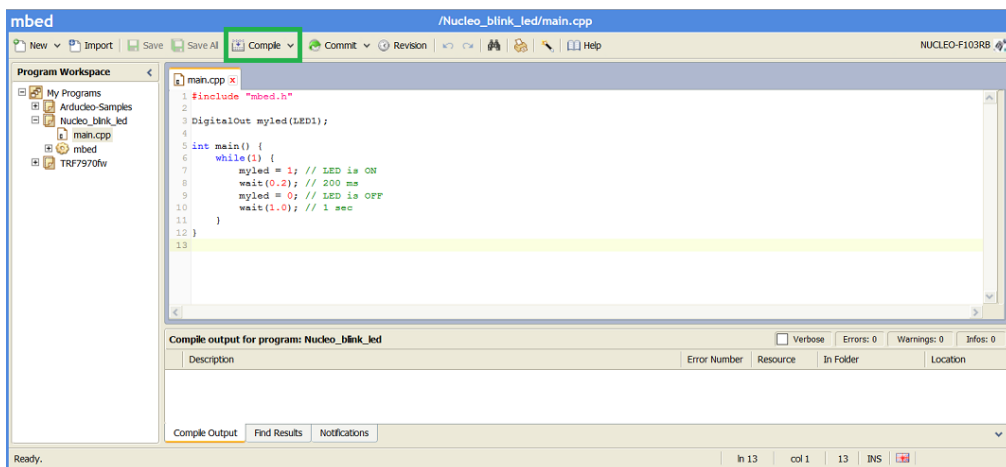
### 5.2 新建一个工程



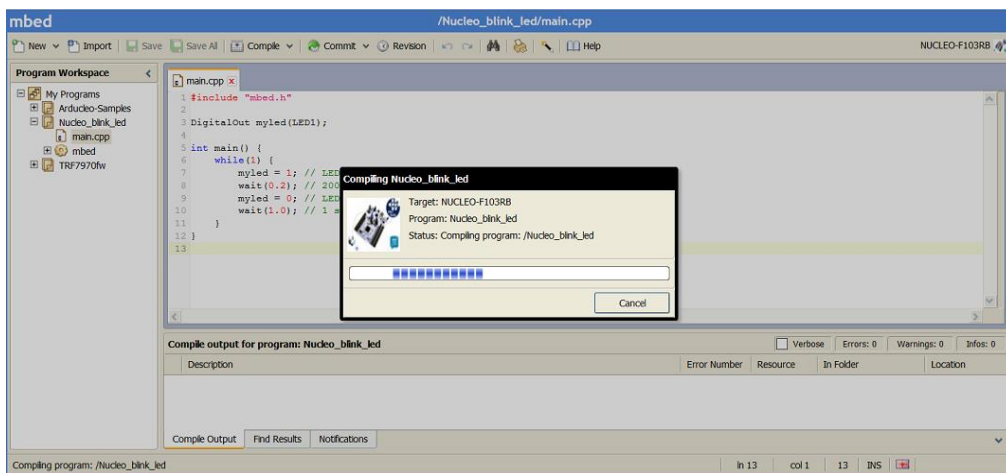
当你登陆到 mbed 后，你可能看到上图这样的界面。点击“Compiler”或者“Open mbed Compiler”即可进入 mbed IDE 界面：



在上图弹出的对话框中，直接点击 OK，就可以立刻生成一个驱动板载 LED 闪烁的工程。



此时点击工具栏中的“Compile”即可立刻在服务端后台编译、链接此工程。稍后，会弹出一个下载文件的提示：



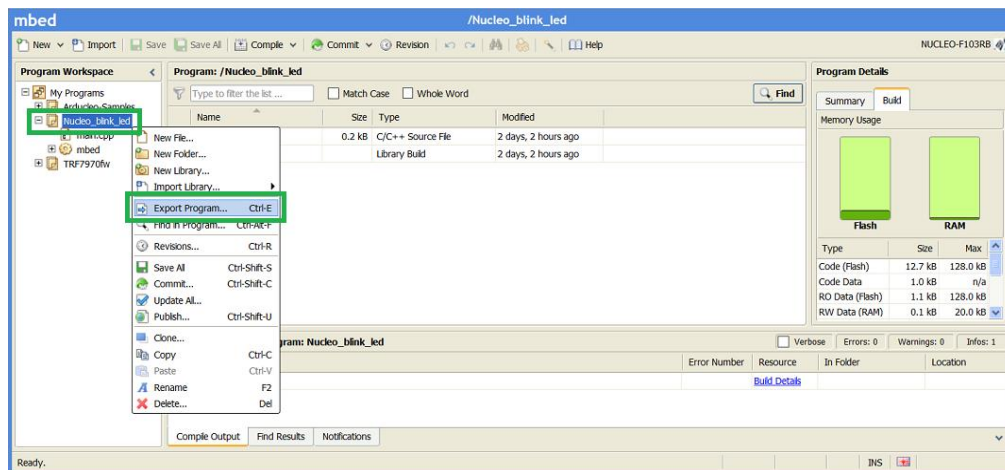




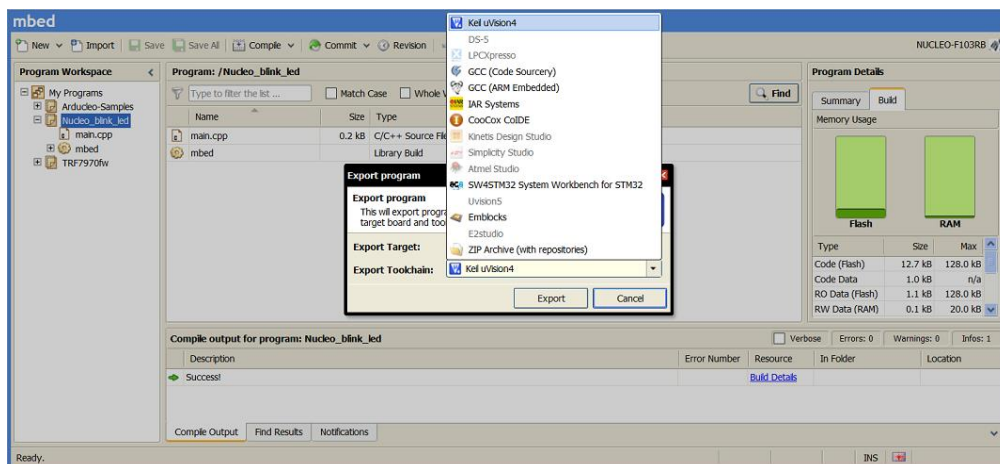
下载此文件后，将其拖放（复制/粘贴）到 DAPLink 对应的虚拟磁盘，即可实现对目标芯片固件更新的操作。稍后，你就可以看到开发板的 LED 开始闪烁。

## 5.3 导出一个工程

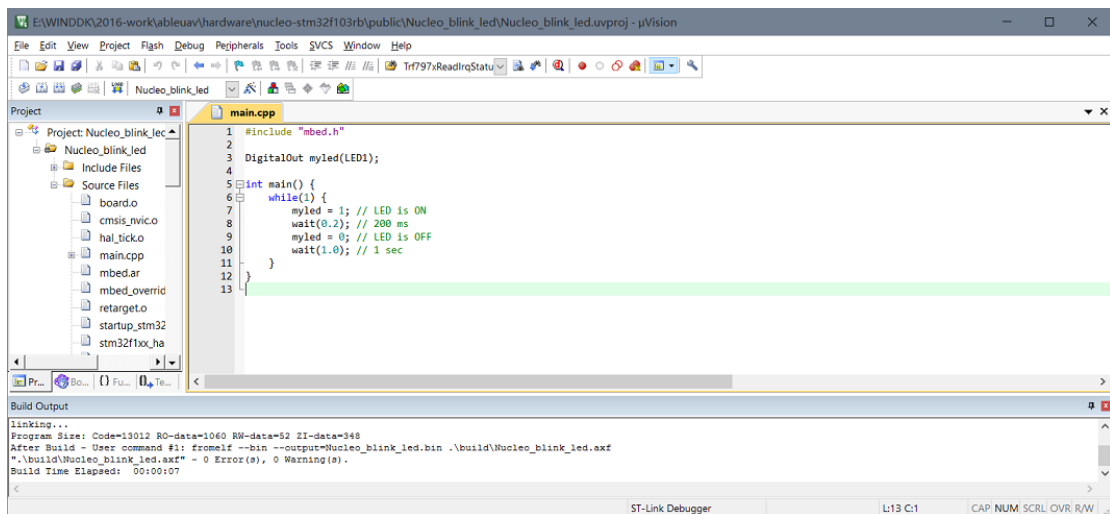
如果你打算将程序下载到本地的 IDE 比如 Keil MDK 中进行编辑，也是可以的：



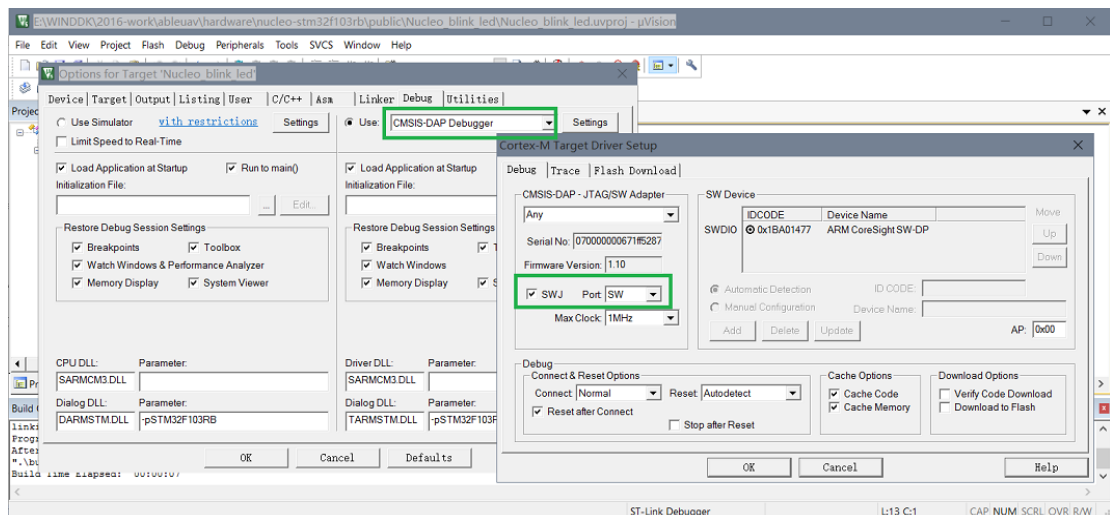
右键点击工程，在弹出的上下文菜单中，选择“Export Program...” 然后弹出工具链选择的对话框：



这里，我们选择 Keil uVision4 即可。稍后，我们将工程文件保存到本地，用 Keil uVision 打开即可编辑、链接。



为了下载、调试，我们需要做一点改动：



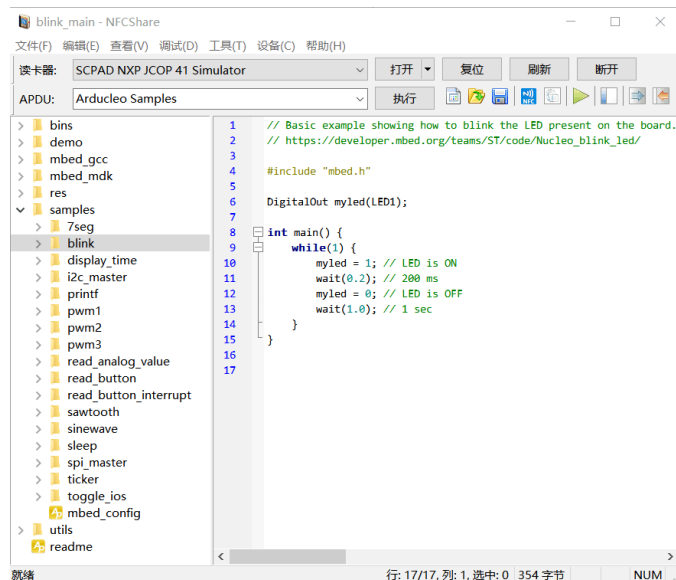
如上图，打开工程选项对话框后，在 Debug 标签页，选择“CMSIS-DAP Debugger”。在弹出的对话框中，指定“SWJ”，“SW”两个选项即可。

此时，你可以用 F8 下载固件，然后按一下开发板上的复位按钮，新代码就开始跑起来了。

或者，你用 Ctrl+F5 进入调试模式。

## 6. 示例程序管理工具

Arducleo 提供了一组简单的实例程序，包括了它们的源码和 bin 输出文件。对于 Windows 用户，使用一个图形化管理工具 NFCShare，你可以方便的修改代码，即可编译、链接，然后下载到 Arducleo 开发板。当然，如需调试代码，还是建议使用 MDK，emBitz，VisualGDB，这样的专业 IDE 进行。



### 6.1 如何下载示例

NFCShare 程序以及示例源码可以从 github 上获取：<https://github.com/dotnfc/arducleo>

如需完整发型包（包括编译、链接用的 arm-gcc，下载所需的 openocd 可从下述地址获取：

<https://github.com/dotnfc/arducleo/releases>

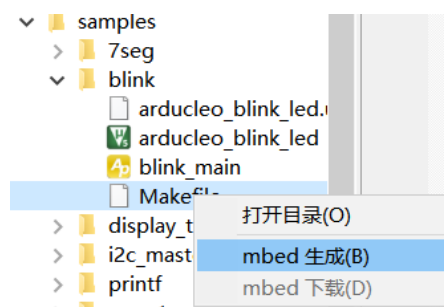
下载完整发型包后，解压缩至任何目录，双击 NFCShare.exe 即可执行。

## 6.2 子目录的说明

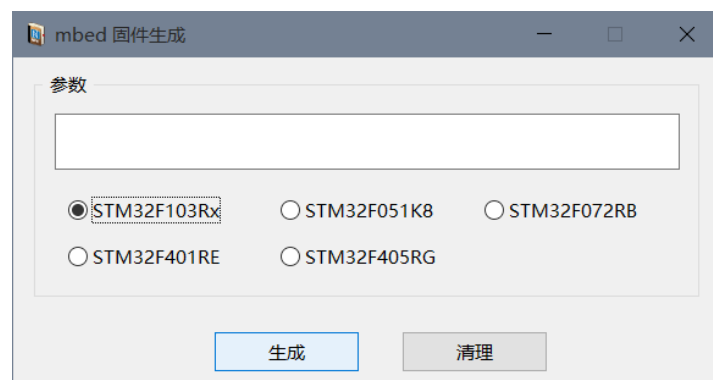
▼ Arduino-Samples	
▼ bin	编译、链接，烧写程序
> arm_gcc	
> openocd	
▼ scripts	
bins	生成的 bin 固件程序
demo	Nfcshare 的演示脚本
> mbed_gcc	Mbed arm-gcc 依赖库及头文件
> mbed_mdk	Mbed arm-mdk 依赖库及头文件
res	Nfcshare 脚本所需资源
> samples	Arducleo 示例程序集
utils	Nfcshare 启动脚本及工具脚本

## 6.3 如何编译、链接代码

- 1). 打开示例程序所在目录，并在其 Makefile 文件上右键点击“mbed 生成”：



- 2). 在弹出的对话框中，点击“生成”：



此时即可看到 nfcshare 的输出窗口出现如下信息：

```

===== Build =====
E:\temp\Arduino-Samples\scripts\samples\blink\Makefile

compiling ../blink_main.cpp
linking arducleo_blink.elf
arm-none-eabi-objcopy -O binary arducleo_blink.elf arducleo_blink.bin
copy arducleo_blink.bin ../../bins
已复制      1 个文件。
arm-none-eabi-size arducleo_blink.elf
   text    data    bss     dec      hex filename
  13184    124    564   13872   3630 arducleo_blink.elf

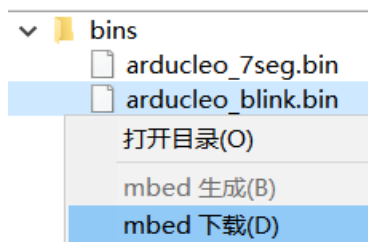
```

上图中，你可以看到，示例工程对应的 .bin 文件同时被复制了一份到 bins 目录下，方便使用。

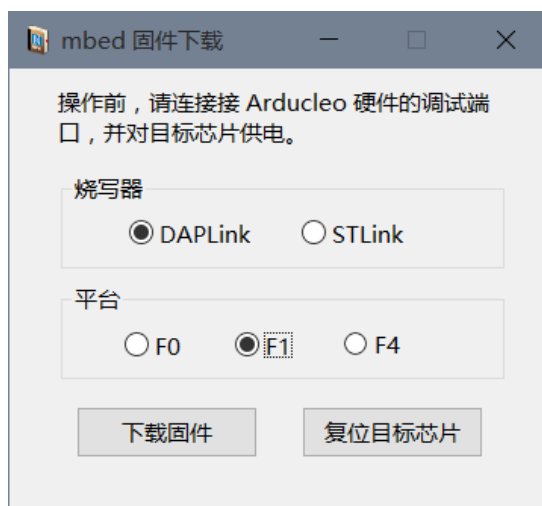
提示：编译对话框对应的脚本文件位于 utils/mbed\_build.c。生成所执行的命令是 'make'；清除则是 'make clean'。

## 6.4 如何下载新固件

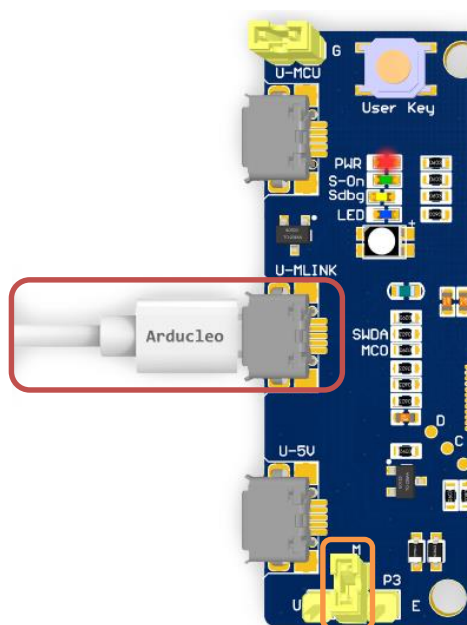
- 1). 在固件文件（扩展名为 .bin）上点击鼠标右键，在弹出的菜单上点击“mbed 下载”：



- 2). 在弹出的对话框中：



- “复位目标芯片” – 可以利用 swd 命令软复位目标芯片
- “下载固件” – 将已选中的 .bin 通过板载的 daplink 通过 swd 下载到目标芯片并运行

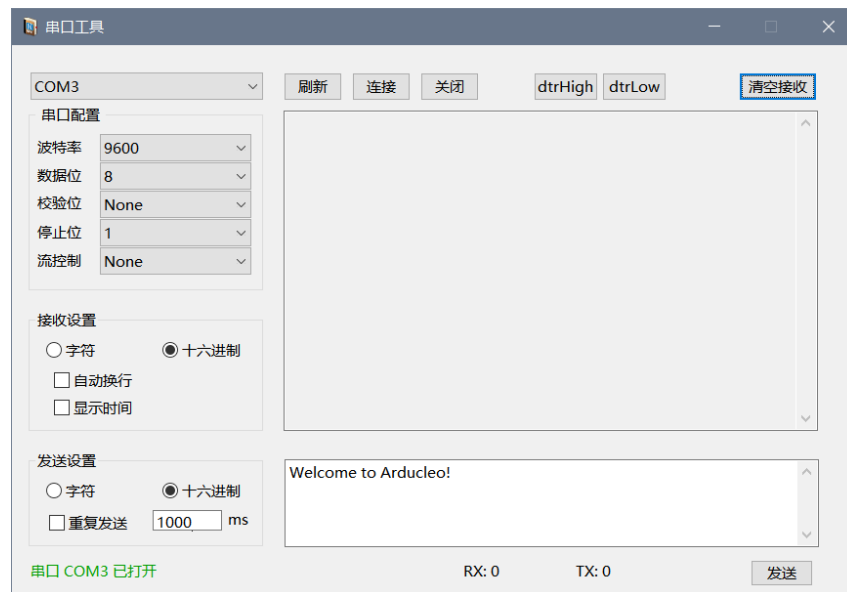


提示：编译对话框对应的脚本文件位于 `utils/mbed_download.c`

## 7. 示例程序

示例程序均采自 mbed 官网的 nucleo 参考代码，源码中也保留了原始路径。另外，针对 Arducleo 的硬件，增加了多个示例。下面选择几个具有代表性的工程进行说明。

此外，示例程序中时常会用到串口输出，你可以使用熟悉的串口工具，比如串口助手，putty，securecr 等进行输出显示；也可以采用 ‘demo/B.串口助手’来做简单的辅助操作，其 UI 截图如下：



### 7.1 Blink

这是驱动 LED 闪烁的一个示例，程序上来先点亮 LED，等候 200ms 后关闭；1 秒钟后再点亮，如此反复。这里用到的外设是 GPIO (General Purpose Input Output)。



源码中的 LED1 是一个宏定义，也就是连接到 PA\_5 这个 GPIO 的，当 LED1 高电平，U3 导通，D4 亮；



反之 U3 截止，D4 灭。

## 7.2 printf

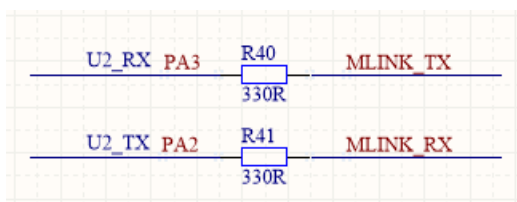
这个示例演示了在代码中，利用 printf 接口输出格式化信息到 pc 端的功能。默认的，printf 的波特率是 9600，8 位数据，无奇偶校验。示例中，重新实例化了一个 Serial 类的 pc 对象，利用这个对象，我们还可以在代码中修改这个波特率，比如 pc.baud(115200)。

```

1 // Display a message on PC using UART.
2 // https://developer.mbed.org/teams/ST/code/Nucleo_printf/
3
4 #include "mbed.h"
5
6 //-----
7 // Hyperterminal configuration
8 // 9600 bauds, 8-bit data, no parity
9 //-----
10
11 Serial pc(SERIAL_TX, SERIAL_RX);
12
13 DigitalOut myled(LED1);
14
15 int main() {
16     int i = 1;
17     pc.printf("Hello World !\n");
18     while(1) {
19         wait(1);
20         pc.printf("This program runs since %d seconds.\n", i++);
21         myled = !myled;
22     }
23 }
24

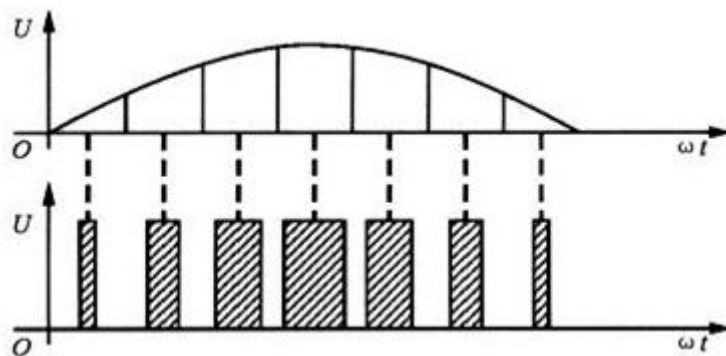
```

此外，代码中的“SERIAL\_TX”，“SERIAL\_RX”是目标芯片的 PA\_2, PA\_3 管脚。而这两个管脚，则通过 DAPLink 的 RX/TX 管脚将信息输出到 PC 端或者接收来自 PC 端的信息，参见原理图：



## 7.3 PWM 输出

Pwm 它是一种对模拟信号电平进行数字化编码的方法，也就是通过一个时钟周期内高低电平的不同占空比来表征模拟信号的方法，比如下面这个编码样例（三角波是用来生成 PWM 编码的）：



示例代码中，一共有 5 个关于 pwm 的示例。如需直观地了解其输出，需配合逻辑分析仪，或者示波器查看波形。这里，我们摘录三个来分析。

### 7.3.1 Pwm1- 基本的 pwm 输出



```

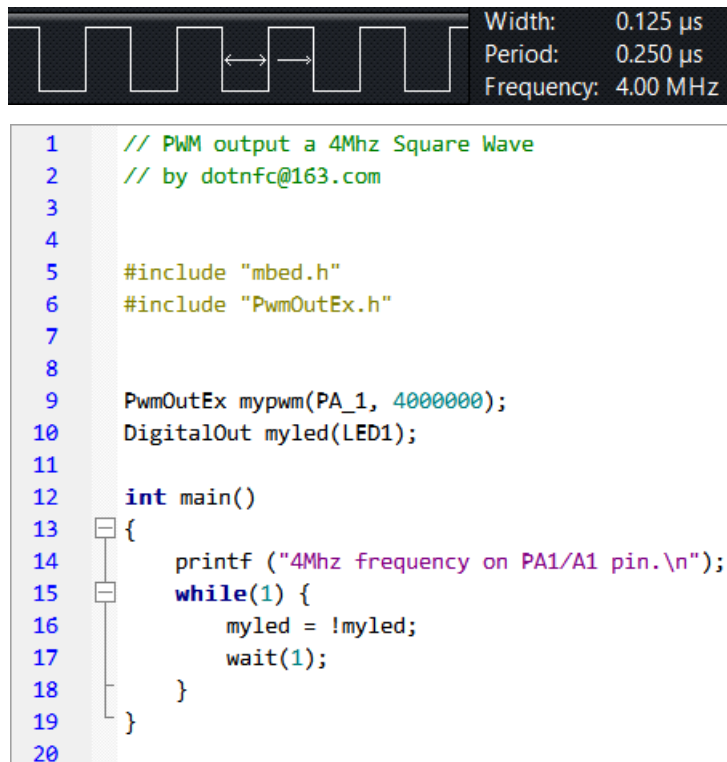
1 // Output a pwm signal.
2 // https://developer.mbed.org/teams/ST/code/Nucleo_pwm/
3
4 #include "mbed.h"
5
6 PwmOut mypwm(PWM_OUT);
7
8 DigitalOut myled(LED1);
9
10 int main() {
11
12     mypwm.period_ms(10);
13     mypwm.pulsewidth_ms(1);
14
15     printf("pwm set to %.2f %%\n", mypwm.read() * 100);
16
17     while(1) {
18         myled = !myled;
19         wait(1);
20     }
21 }
22

```

代码中，脉冲周期为 10ms，脉冲持续时间为 1ms (A)，而逻辑分析仪显示 A + B = 10ms，符合预期。

注：PWM\_OUT 宏是 Arduino 接口的 D3，也就是 PB\_3 管脚。

### 7.3.2 Pwm4- 输出 4Mhz 的方波



由于 mbed 平台众多，其统一的 api 无法实现输出大于 1Mhz 的方波，所以，这个示例对原 PwmOut 进行了扩展，从而实现高频率的方波的输出（注意，STM32F103 的最大主频为 72Mhz，此法不能输出大于主频的方波）。这种方波在某些场合下，可以为外部电路省去一个晶振，也可以用于验证电路原理。

比如，在智能卡芯片的特殊应用中，GPIO 用于做通讯，而传统的 7816 接口则作为调试输出，但其需要外部时钟才能工作。用此法，配合板载的 DAPLink 即可得到调试输出的基本电路。

### 7.3.3 Pwm5- 驱动蜂鸣器

使用 pwm 输出，可以通过调整频率和持续时间，演奏出简单的音乐出来，这个示例，包含了两段游戏音乐：超级玛丽（pwm5\_mario）和魂斗罗一代第一关背景音（pwm6\_contra）。请下载固件后，欣赏。

## 8. 常见问题

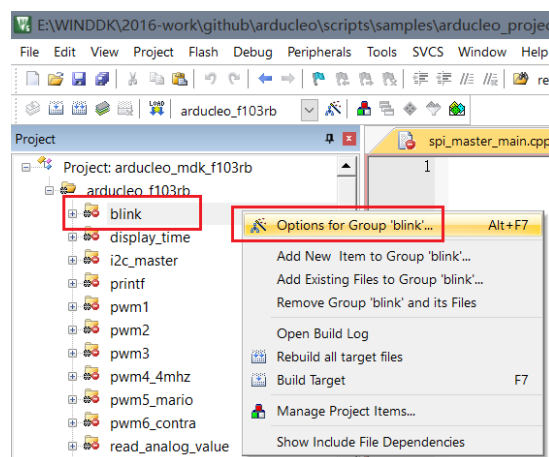
### 8.1 DAPLink 是否支持其他芯片？

DAPLink 支持 Arm Cortex 处理器的开发、调试。参见：

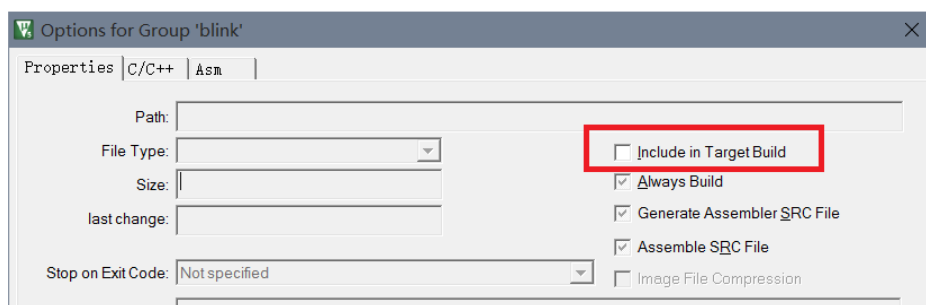
[http://www.keil.com/support/man/docs/dapdebug/dapdebug\\_introduction.htm](http://www.keil.com/support/man/docs/dapdebug/dapdebug_introduction.htm)

### 8.2 如何使用 MDK 调试我的程序？

请使用 samples/arducleo\_projects/ 的对应芯片工程进行调试( 注意调整调试所用到的仿真器和下载算法 )。



通过修改代码目录的属性，切换要调试的代码



“Include in Target Build” 切换开关代码是否进行构建。图中所示为排除；双击后复选框变为灰色，即进行构建。



