使用 J-Link 编程 9200 核心板的 FLASH

Rev1.0

Team Mcuzone

2007-03

内容：利用带 J-FLASH 授权的 J-LINK 编程 9200 核心板上的 NOR FLASH
工具：J-Link 硬件，J-FLASH 软件，串口模块
目标板：9200 核心板

使用 J-Link 编程 9200 核心板的 FLASH

——Team Mcuzone

使用 J-Link 测试和编程 9200 核心板（2 层板），测试内容包括检测 9200 芯片 ID，检测 9200 核心板上的 FLASH 的 ID，并进行 FLASH 烧写测试，烧写成功后进行调试。

首先，进行 28F128 的检测和烧写。请先把 FLASHBOOT 的跳线短接；然后把 AT49-BOOT-28F 的跳线跳到 28F 这端，即表示片选 28F128；另外，请确保 JTAG-ICE 的跳线短接在右边（即靠近 ICE 字符那边）。如果手头有串口模块，请把串口模块连接到 DBGU，即 PA31 和 PA30，其中 PA31 是 DTXD，PA30 是 DRXD。硬件连接妥当后，打开 J-FLASH 软件，在 File 菜单下选择 Open project，然后把路径指向到 ProjectFiles 下面，如下图所示：

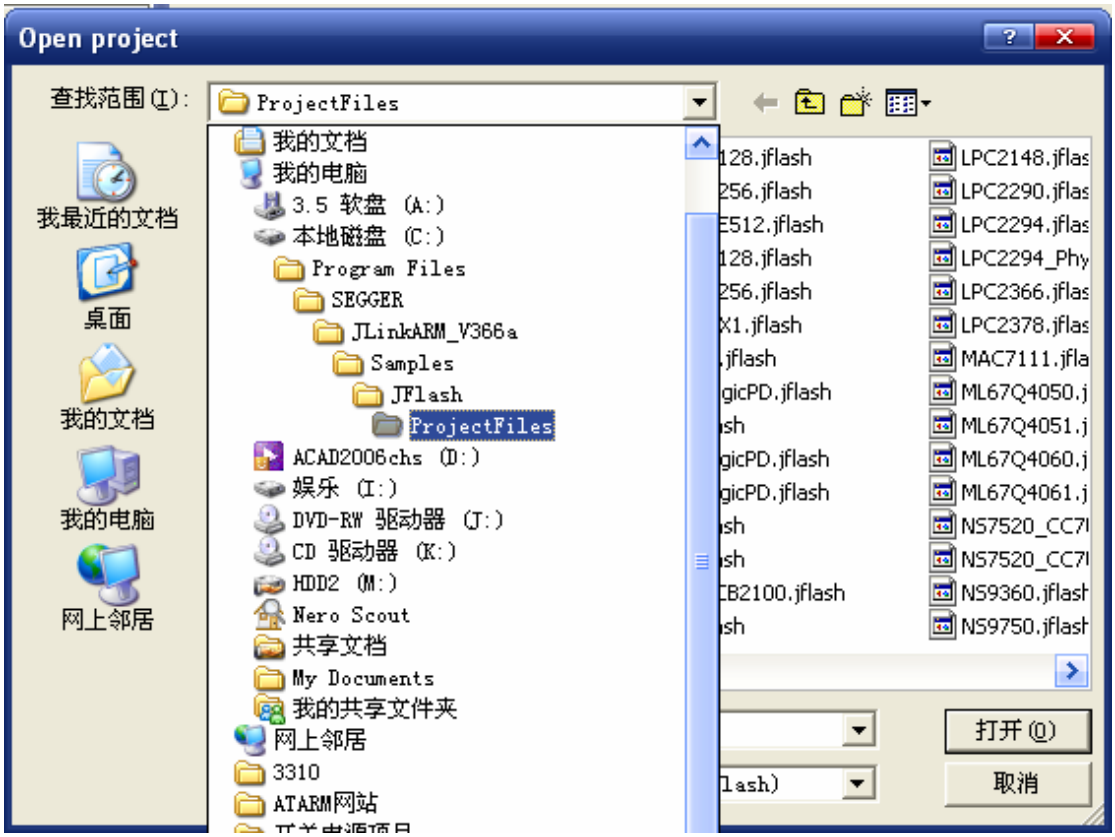

图 1，Open Project

请选择“AT91RM9200_EK.jflash”，然后点击打开。
这里我们都可以使用这个 project 的默认配置：

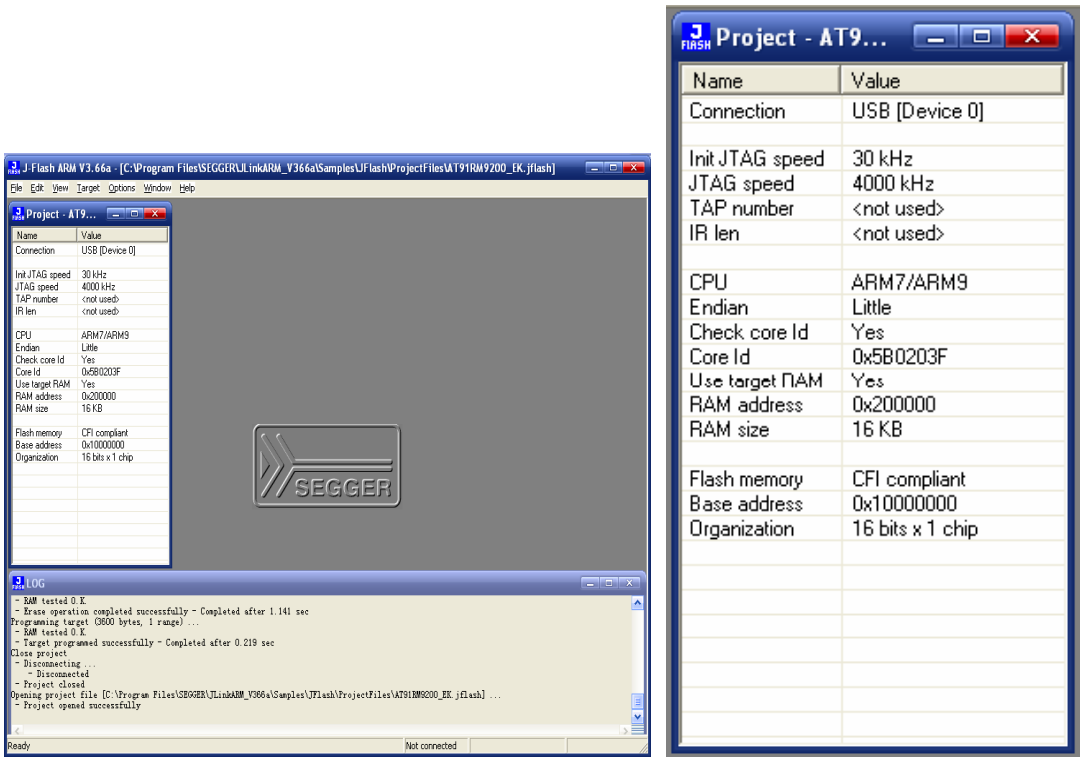

图 2，打开 9200 EK 的 Jflash 工程

默认的 JTAG 初始化速度是 30KHz，JTAG 操作速度是 4000KHz，CPU ID 是 0x5B0203F 即 AT91RM9200，RAM 地址是 0x200000，大小是 16K，这个是 9200 片内的高速 RAM。FLASH 存储器我们就直接选择 CFI，基地址是 0x10000000，16 位宽。

如果要对这些参数进行修改，可以到“Options”下的“Project settings”内进行设置：

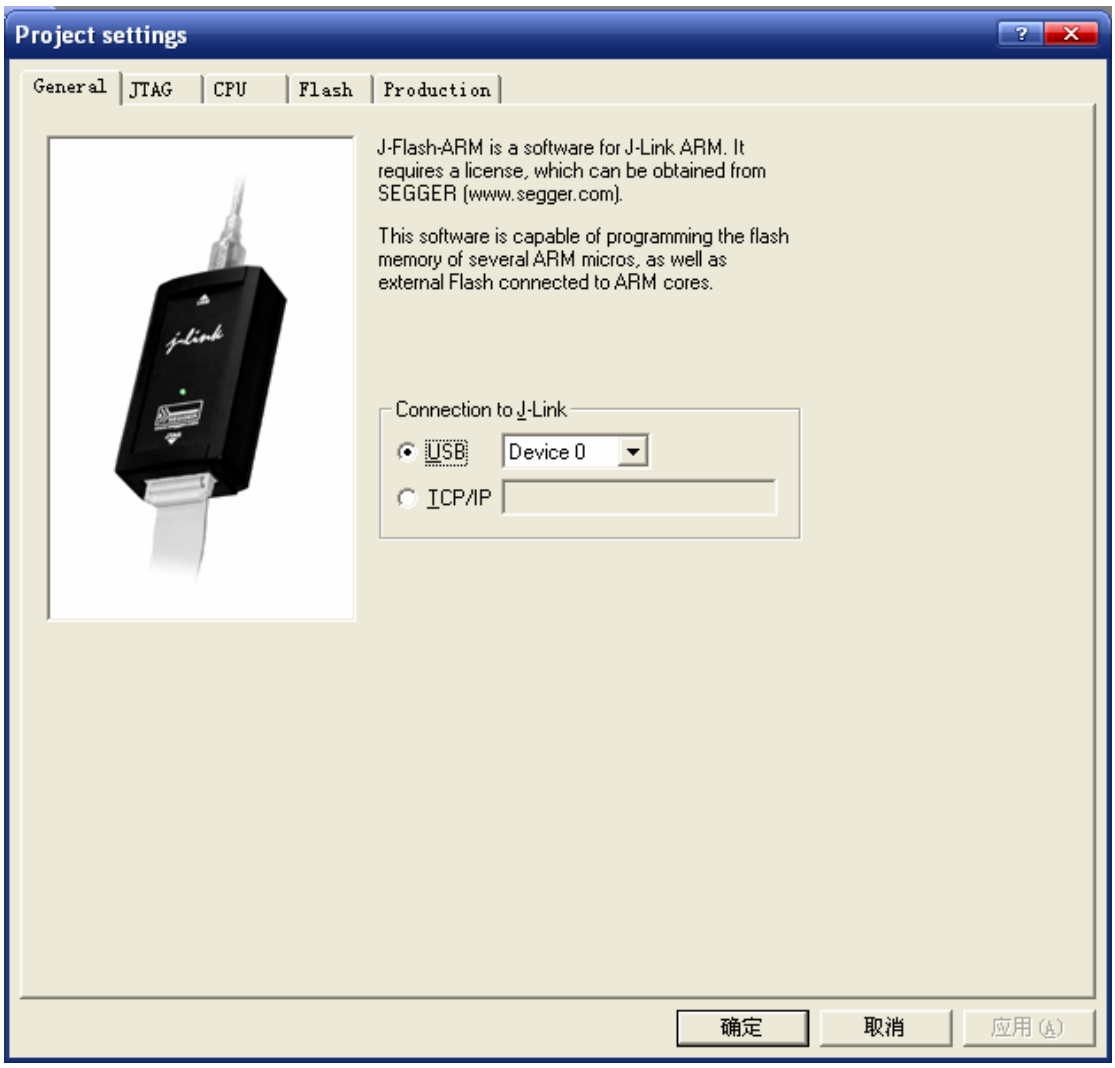

图 3，Project settings－General

这个菜单可以设置 J-LINK 的使用方式，不光可以本地使用，也可以网络使用。

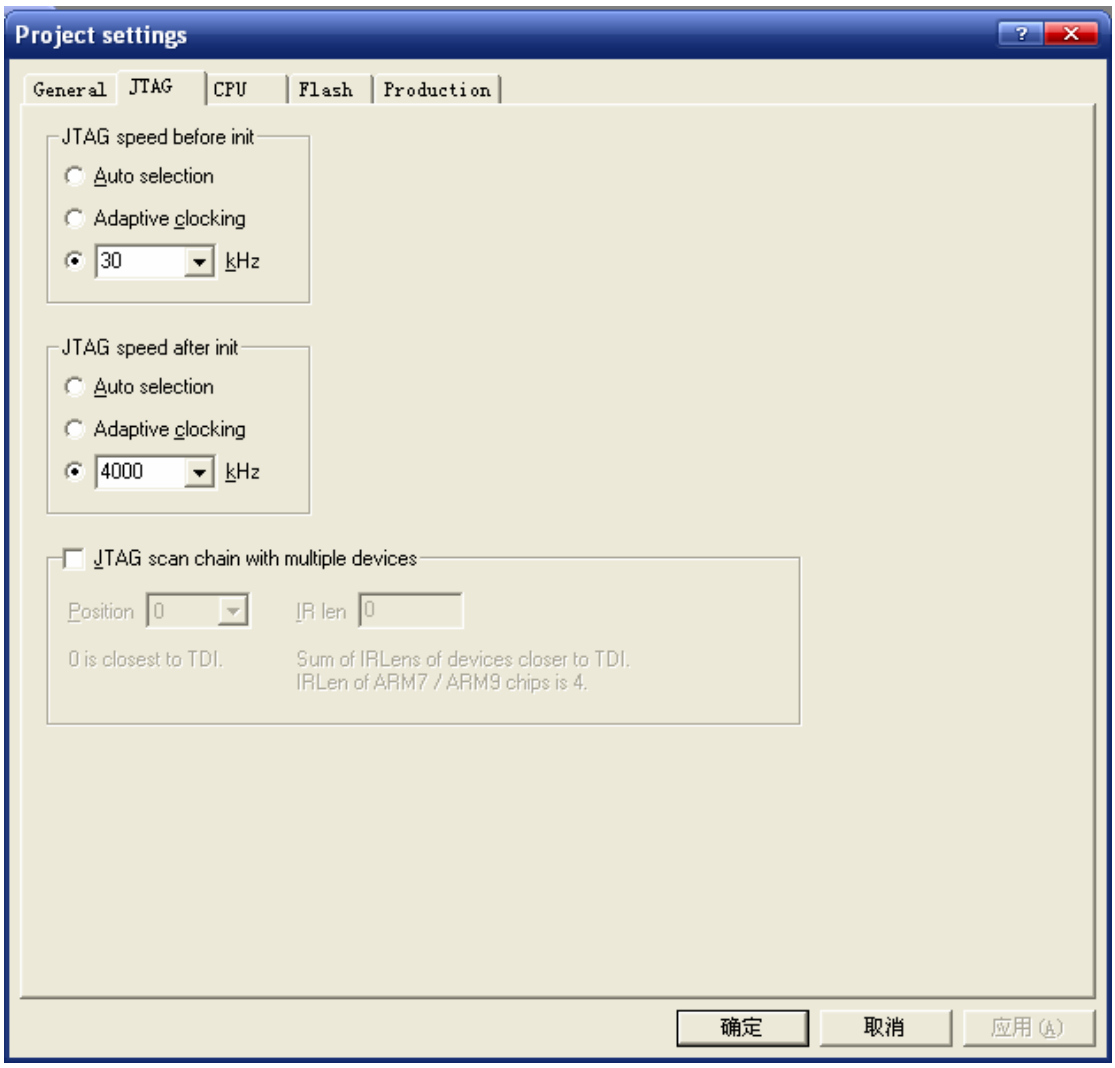

图 4，Project settings－JTAG

这个菜单用来对 JTAG 进行设置，JTAG 速度最高可以设置到 12M，建议两个 JTAG 时钟均使用 Auto。

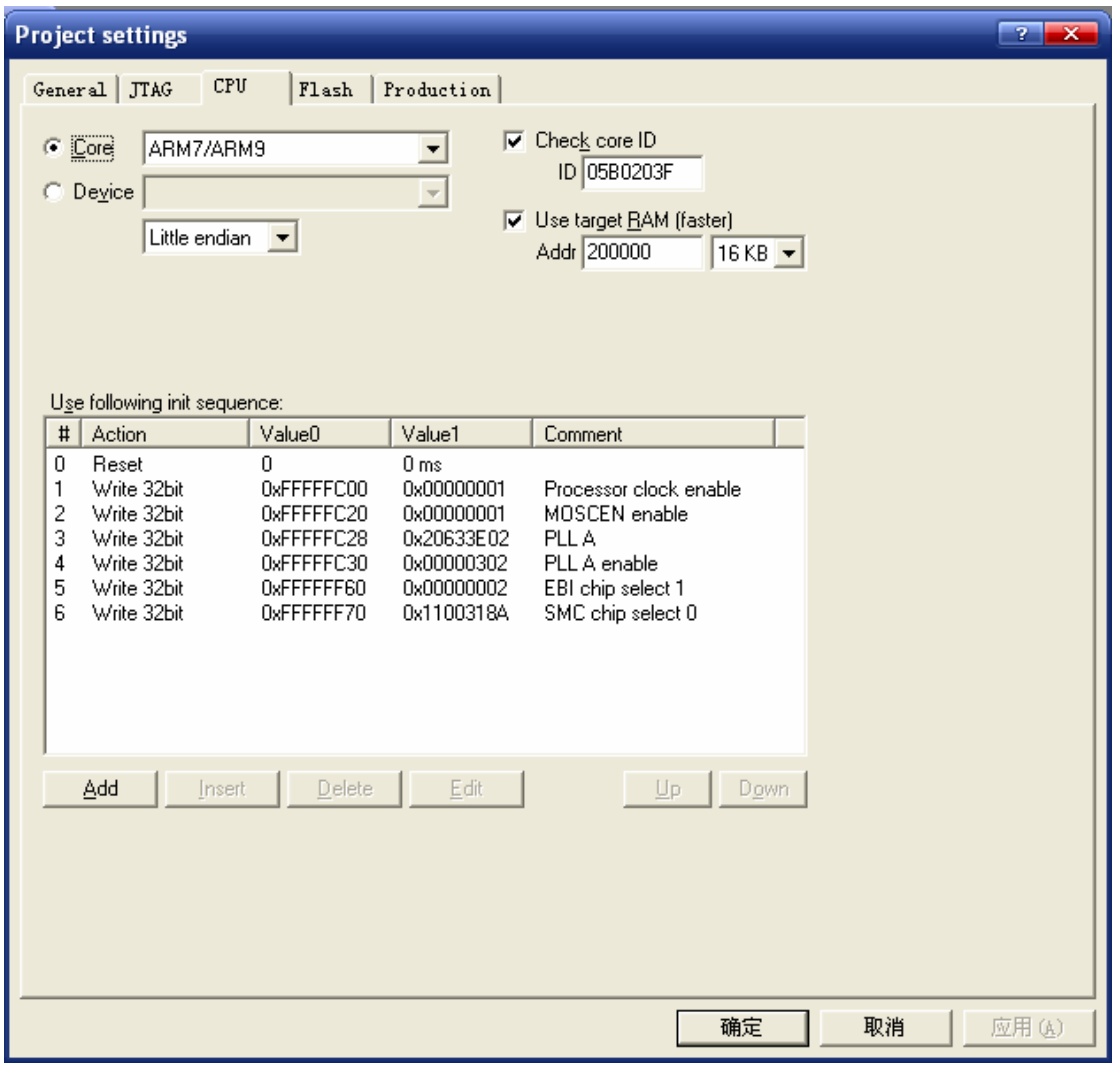

图 5，Project settings－CPU

CPU 菜单有重要的 CPU 初始化信息，这是我们直接使用现成的 JFLASH 工程的最大原因。如果你需要设置的工程在 ProjectFiles 文件夹下找不到，那就得自己新建工程了，新建的工程需要对 CPU 进行完善的初始化才能很稳定的使用，不然就会不稳定，出错，甚至根本不能烧写。

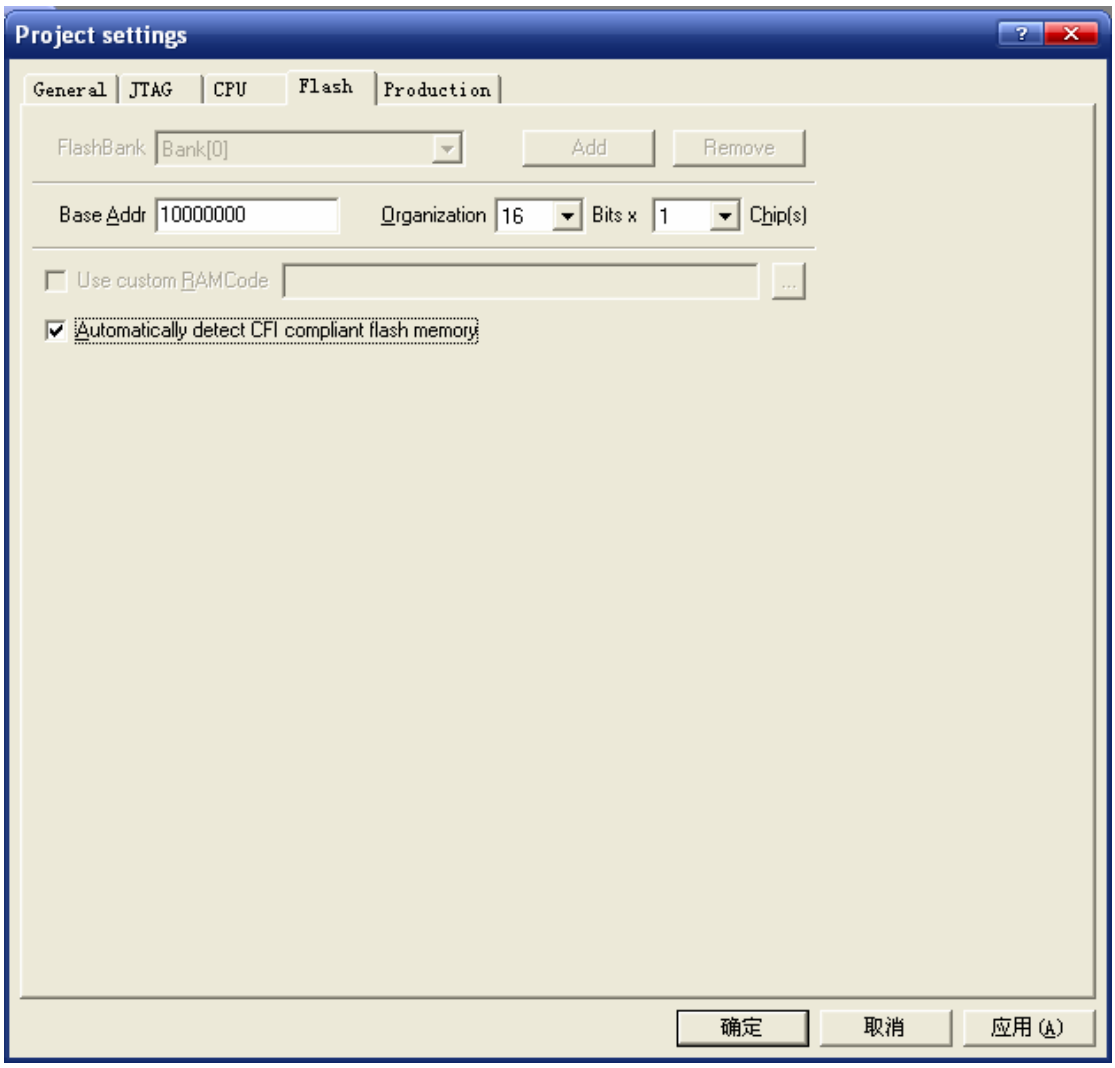

图 6，Project settings－FLASH

FLASH 菜单里面也有不少的设置，由于我们使用的是符合 CFI 要求的 28F128 和 AT49BV163D，所以我们可以直接使用 Automatically detect CFI…。

如果需要自行设置，可以把 AUTO 前的勾去掉即可。会出现一个“Select flash device”的按钮，点击按钮会出来一个可供选择的 FLASH 列表，这个列表包含了大部分市场上可以找到的 NOR FLASH。

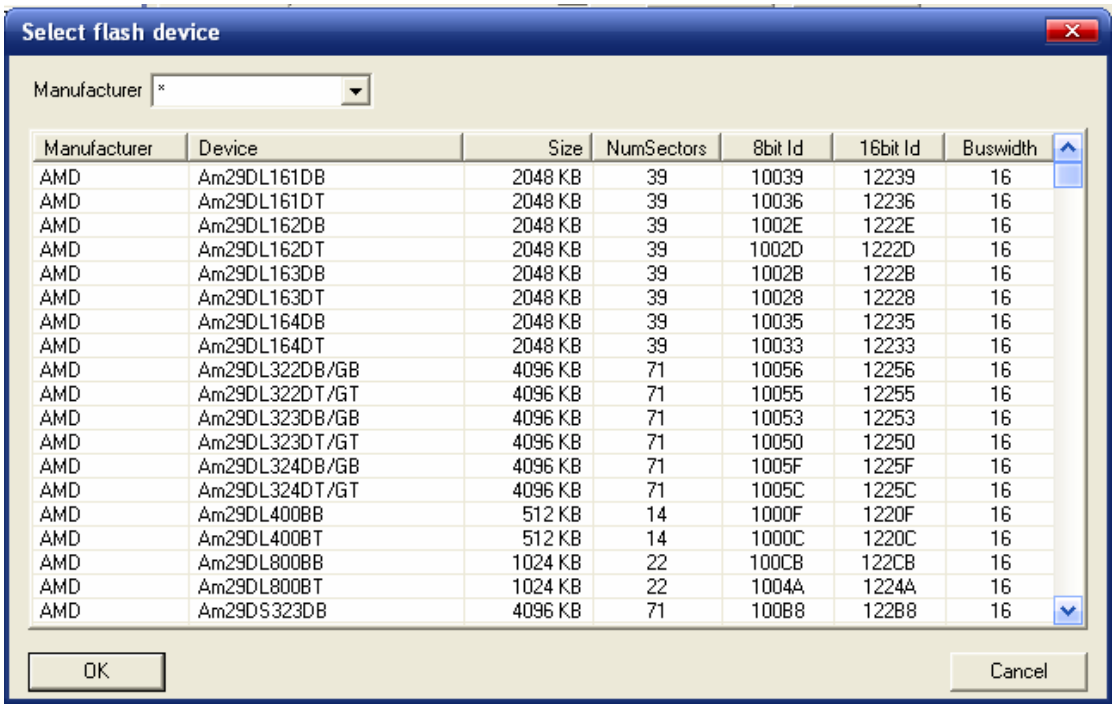

图 7，Project settings－Select FLASH Device

可以按照客户实际使用的 FLASH 型号进行选择。同时可以看到这些 FLASH 的生产厂家和 FLASH 的不少信息。

完成以上设置后，我们开始进行各种操作。

首先，选择“Target”菜单下的“Connect”，如果硬件连接可靠，在 log 框内将会出现以下提示内容：

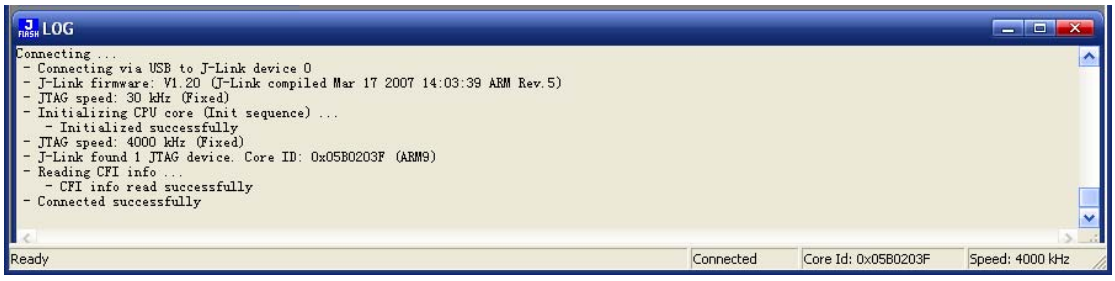

图 8，Connect

可以看到 CPU 和 FLASH 都已经读取成功。FLASH 的容量和型号也被正确识别：

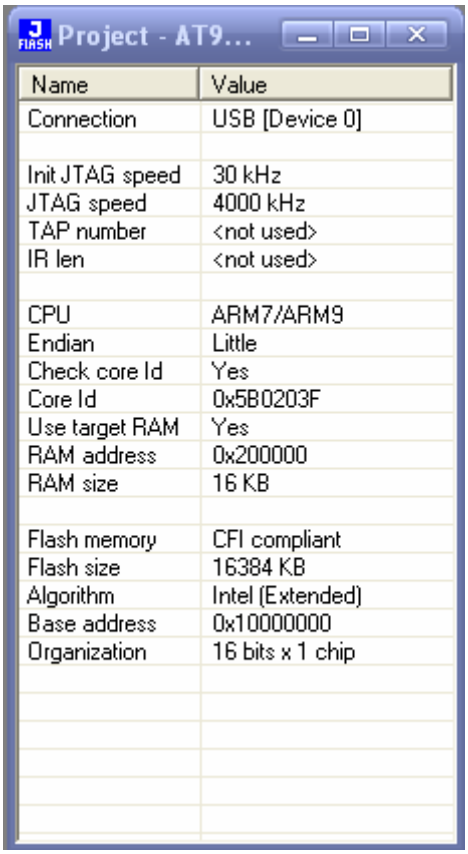

图 9，FLASH 信息

下面我们来对 28F128 进行一下擦除，查空，编程，校验等操作。

首先是擦除，请选择 Target 菜单下的 Erase chip，

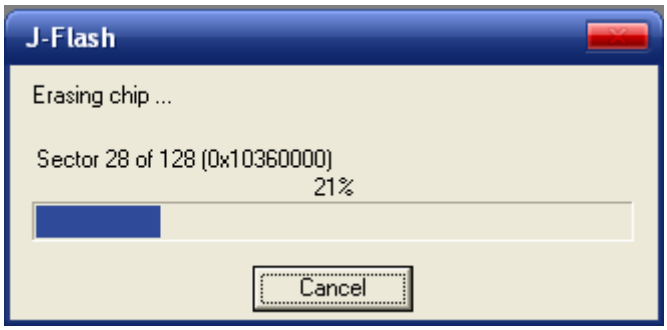

图 10，Erasing chip

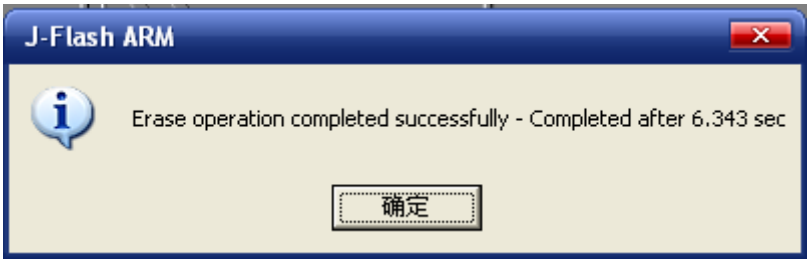

图 11，Erase 完成

很快，芯片擦除成功，并会提示擦除所用的时间。这些信息也会被记录在 log 框内。

擦除后再检查一下是否是空，选择 Target 菜单下的 Check blank：

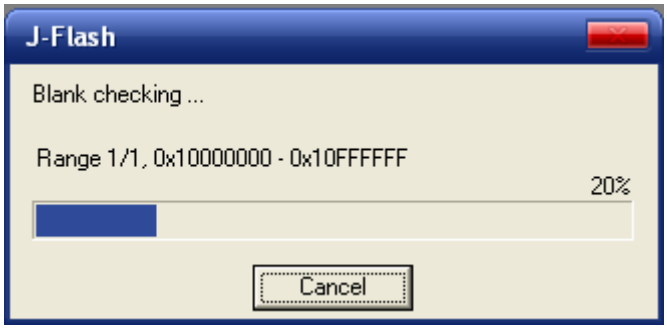

图 12，查空

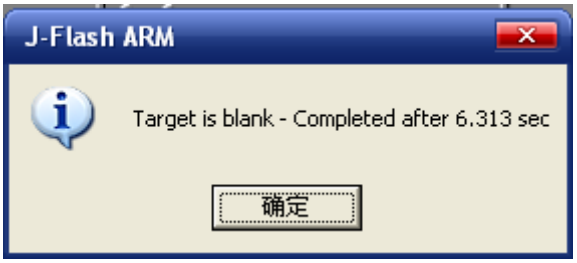

图 13，查空结束

查空操作证明之前的擦除操作的正确完成的。

接下来我们进行编程操作。把预先准备好的一个程序烧入到 0x10000000 地址，JFLASH 支持四种文件，分别是 hex，bin，mot，srec。一般我们用 hex 和 bin 文件。通过 File 菜单下的 open 子菜单来选择需要写入的文件：

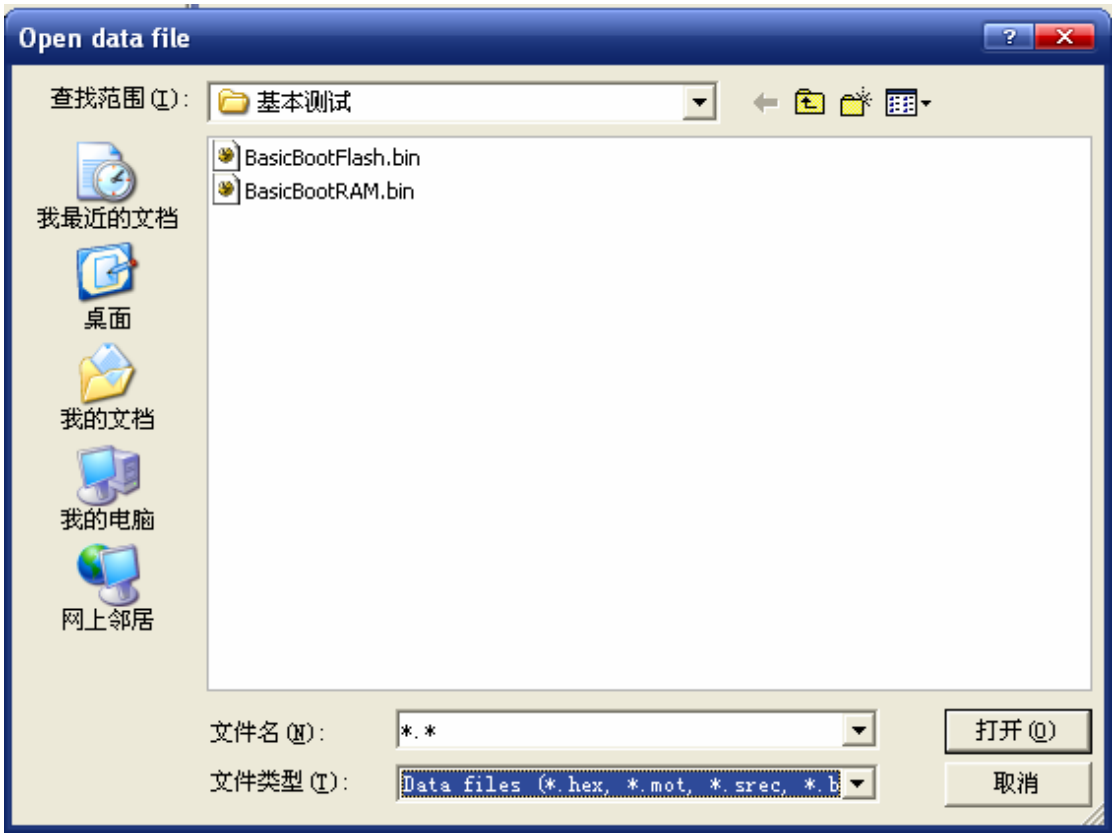

图 14，打开 Bin 文件

选择好 bin 文件后点击打开按钮，JFLASH 会提供输入其实地址，这里我们输入 10000000，默认是 16 进制，10000000 表示 NOR FLASH 的首地址。点击 OK 进行确定。

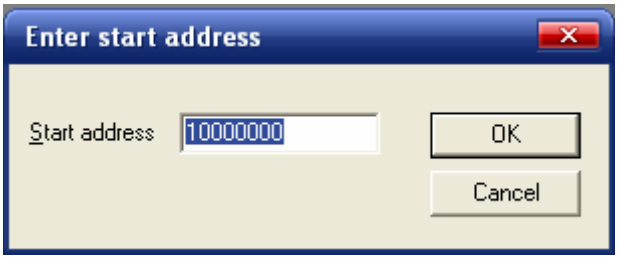

图 15，输入地址

在 Target 菜单下选择 Program，由于 bin 文件很小，很快就烧写完成。

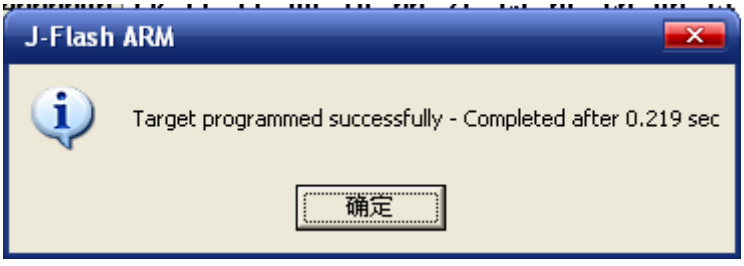

图 16，烧写完成

烧写完成后可以用 verify 进行校验。

图 17，Verify

为了方便起见，建议直接使用“Program&Verify”。

烧写完成后，请点击 Target 菜单下的 Disconnect，然后把串口模块连接到 9200 核心板上，复位后可以看到输出：

图 18，输出结果

输出结果验证了整个 FLASH 编程过程的正确性。同样，如果编程 AT49BV163D 也是一样的过程，只需要把 AT49-BOOT-28F 上的跳线切换到 AT49BV 端即可。

如果使用 CFI 设置，AT49BV163D 将被识别成 AMD 的芯片，不过这不影响 FLASH 编程的正确性。其他擦除，编程，校验都是同样的操作。编程完成后，复位，可以看到和图 18 一样的输出结果。如果担心这个运行结果是 28F128 内的程序而不是 AT49BV163D 内的内容，可以先把 28F128 整片擦除，然后再跳线到 AT49BV163D，然后编程，看看能不能输出同样的内容。

到此为止，使用 JLINK 配合 JFLASH 编程 NOR FLASH 的过程已经非常清楚了，可以看到使用 JFLASH 这个软件可以非常方便而且快速的编程 9200 核心板上的 NOR FLASH。

注意：烧写完成后请注意点击 Target 菜单下的 Disconnect，不然即便按了 RESET 按钮，9200 也不会执行 NOR FLASH 内的代码。