SAM9260 地铁用非接触 IC 卡大读卡机编程手册

版本 1.10

2014 年 4 月 11 日

苏州市永兴电子有限公司

目录

简介

为了方便 SAM9260 读卡机的编程与使用，我们为用户提供了包括源代码在内的许多程序，主要内容见下表.

文件夹 BSP		
子文件夹 AuxSerial	SAM9260 芯片放弃控制 UART0，将辅助处理器串口和上位机连接起来的 demo 程序	源代码
子文件夹 AuxUSB	通过串口 UART2 控制辅助处理器，使用辅助处理器的 USB 的 demo 程序	源代码
子文件夹 BootRecovery	一个用于恢复处理器 SAMBA 引导的程序	源代码
子文件夹 BSP	SAM9260 板的板级支持包源代码，用于操作主处理器的各种外设，如 USB,SD 卡,SPI,UART 等.	源代码
子文件夹 GPIO	使用 SAM9260 芯片进行 GPIO 操作的 demo 程序. 运行在片上 4KRAM 中.	源代码
子文件夹 I2C	使用 SAM9260 芯片读写 EEROM 的 demo 程序	源代码
子文件夹 MemoryMapping	通过串口 UART2 控制辅助处理器，使用辅助处理器的内存映射进行读卡操作的 demo 程序	源代码
子文件夹 Overclocking	SAM9260 芯片超频的 demo 程序	源代码
子文件夹 RTC	使用 SAM9260 芯片实时时钟的 demo 程序	源代码
子文件夹 SPI	使用 SAM9260 芯片读写串行 Flash 的 demo 程序	源代码
子文件夹 UART	使用 SAM9260 芯片 UART 外设读写串口 demo 程序 运行在片上 4KRAM 中.	源代码
文件夹 Linux		
子文件夹 environment		
arm-2013.11-33-arm-none-linux-gnueabi.bin	Sourcery CodeBench Lite Edition，ARM GCC 编译器 4.8.1 版	安装包
eclipse-cpp-kepler-SR1-linux-gtk-x86_64.tar.gz	Eclipse C++ Linux 4.3.1 版	安装包
Tftpd64-4.50-setup.exe	TFTP 服务器安装程序	安装包
子文件夹 u-boot	u-boot 1.3.4 版 及补丁	源代码
子文件夹 linux		
linux-2.6.30.tar.bz2	Linux 内核源代码 2.6.30 版	源代码
2.6.30-at91.patch.gz	Linux 内核源代码 AT91SAM9260 芯片移植补丁	源代码
2.6.30-at91-exp.3.tar.gz	Linux 内核源代码补丁	源代码
ncurses-5.9.tar.gz	Ncurses 控制台对话框库，5.9 版	安装包
at91sam9260ek_defconfig	ATMEL 默认 Linux 内核编译配置文件	配置文件

SAM9260_RFIDReader.config	SAM9260 读卡机 Linux 内核编译配置文件	配置文件
子文件夹 openembedded		
openembedded.tar.gz	OpenEmbedded 编译环境	安装包
bitbake-1.8.18.tar.gz	Bitbake 交叉编译器环境 1.8.18 版	安装包
oe_at91sam.tgz	AT91SAM9260 JFFS2 文件系统移植包	安装包
zlib-1.2.3.tar.bz2	图像处理库	安装包
binutils-2.18.50.0.7.tar.bz2	Binutils 汇编器和连接器	安装包
expat-2.1.0.tar.gz	XML 解析库	安装包
XML-Parser-2.34.tar.gz	XML 解析库	安装包
gnutls-2.4.2.tar.bz2	SSL 通信库	安装包
子文件夹 driver		
gpio_sam9260.ko	SAM9260 板 GPIO Linux 驱动	内核模块
gpio_sam9260_debug.ko	SAM9260 板 GPIO Linux 驱动(包含调试信息)	内核模块
spi_sam9260.ko	SAM9260 板 SPI Flash Linux 驱动	内核模块
spi_sam9260_debug.ko & spi_sam9260_debuginfo.ko	SAM9260 板 SPI Flash Linux 驱动(包含调试信息和寄存器读写信息)	内核模块
子文件夹 buildroot		
buildroot-2013.11.tar.gz	Buildroot 编译环境	安装包
dropbear-2013.60.tar.bz2	嵌入式 SSH 库	安装包
libhid-0.2.16.tar.gz	HID 设备库	安装包
子文件夹 utility		
子文件夹 autologin_arm_br	Linux 启动后自动登录程序	源代码
子文件夹 stdprocProgrammer_arm	主处理器通过串口 UART2 将固件程序下载到辅助处理器中	可执行程序
子文件夹 stdprocProgrammer_ia64	上位机通过串口将固件程序下载到辅助处理器中, 主处理器必须放弃控制 UART0,UART2	可执行程序
子文件夹 update_script		
SAM9260_Firmware_Version.txt	固件 FTP 服务器配置文件	配置文件
firmware_update.sh	固件自动更新脚本	BASH 脚本
子文件夹 demo		
子文件夹 gpio_arm_br	Linux 系统 ARM 平台下, GPIO 驱动 demo 程序	源代码
子文件夹 hello_arm_br	Linux 系统 ARM 平台下, HelloWorld 程序	源代码
子文件夹 hello_ia64	Linux 系统 IA64 平台下, HelloWorld 程序	源代码
子文件夹 loader_arm_br	Linux 系统 ARM 平台下,标准加载程序	源代码
子文件夹 protocol_arm_br	Linux 系统 ARM 平台下, 辅助处理器通信协议库	动态库
子文件夹 sharedA_arm_br	Linux 系统 ARM 平台下, 标准加载程序使用的动态库 A	源代码

子文件夹 sharedB_arm_br	Linux 系统 ARM 平台下，标准加载程序使用的动态库 B	源代码
子文件夹 socket_arm_br	Linux 系统 ARM 平台下,套接字操作 demo 程序	源代码
子文件夹 spi_arm_br	Linux 系统 ARM 平台下，SPI 驱动 demo 程序	源代码
子文件夹 test_arm_br	Linux 系统 ARM 平台下,通过向辅助处理器发送命令，进行读写卡操作的 demo 程序	源代码
文件夹 PC		
StdProtocolProgrammer.exe	辅助处理器固件升级程序，该程序通过串口进行 ISP，将辅助处理器的固件下载进去．该程序基于 MFC10.0，可以运行在 Windows XP SP2 及以上操作系统．该程序仅用于本读卡机测试使用，版权为永兴电子所有。未经授权，用户不得发布该程序。	可执行程序
StdProtocolTester.exe	辅助处理器测试程序，该程序通过串口或 USB 与辅助处理器通信，测试其读卡等功能．该程序基于 MFC10.0 及 Python2.7，可以运行在 Windows XP SP2 及以上操作系统．该程序仅用于本读卡机测试使用，版权为永兴电子所有。未经授权，用户不得发布该程序。	可执行程序
IntervalTest.exe	串口或 USB 数据通信时间测量程序，该程序能够监控串口或 USB 的数据包，测量数据包之间的时间差．该程序基于 MFC10.0，可以运行在 Windows XP SP2 及以上操作系统．该程序仅用于本读卡机测试使用，版权为永兴电子所有。未经授权，用户不得发布该程序。	可执行程序
SocketServer.exe	套接字测试程序，该程序可以作为套接字的服务器或是客户端，测试读卡机的套接字通信功能．该程序基于 MFC10.0，可以运行在 Windows XP SP2 及以上操作系统．该程序仅用于本读卡机测试使用，版权为永兴电子所有。未经授权，用户不得发布该程序。	可执行程序
Patcher.jar	补丁程序，可以处理 epatch 文件，并给目标文件夹打上补丁．该程序基于 JDK1.07_09，用户需要 Java SE Runtime Environment 7u9 及以上版本支持．该程序仅用于本读卡机测试使用，版权为永兴电子所有。未经授权，用户不得发布该程序。	可执行程序

sam-ba_2.12.exe	由 ATMEL 公司提供,用于主处理器的 ISP.该程序通过 USB 将代码烧写入主处理器 NAND Flash 中.	安装包
jre-7u51-windows-x64.exe	Java 运行时环境, 7u51 版本. 如果用户的 PC 上没有 JRE, 请安装该程序.	安装包
Firmware1228.dat	辅助处理器固件程序文件, 用户可以使用 StdProtocolProgrammer.exe 将本文件下载进辅助处理器.	二进制文件
samba.epatch	SAM9260 板的 SAMBA 补丁文件,用于给基于 Windows 系统的 SAMBA 2.12 打上补丁.	补丁文件
linux.epatch	Linux 源代码的移植补丁文件, 用于大存储器的 SAM9260 板.	补丁文件
linux_tiny.epatch	Linux 源代码的移植补丁文件, 用于小存储器的 SAM9260 板.	补丁文件
uboot.epatch	UBoot 源代码的移植补丁文件, 用于大存储器的 SAM9260 板.	补丁文件
uboot_tiny.epatch	UBoot 源代码的移植补丁文件, 用于小存储器的 SAM9260 板.	补丁文件
rootfs.epatch	根文件系统的移植补丁文件, 用于给 Buildroot 打上移植补丁.	补丁文件
文件夹 Document		
SAM9260 编程手册.pdf	SAM9260 读卡机编程手册	pdf 文档
SAM9260 使用手册.pdf	SAM9260 读卡机说明书	pdf 文档
SAM-BA.pdf	SAMBA 程序说明文档	pdf 文档
板级支持包帮助.chm	BSP 源代码说明文档	帮助文档
Sourcery CodeBench Lite.pdf	Sourcery ARM GCC 说明文档	pdf 文档
AT91SAM9260 ARM9.pdf	AT91SAM9260 芯片数据手册	pdf 文档
M25P128 Serial Flash.pdf	M25P128 芯片数据手册	pdf 文档
NAND512xxA2S NAND Flash.pdf	NAND512xxA2S 芯片数据手册	pdf 文档
文件夹 Binary		
uboot_env.bin	UBoot 环境文件,用于大存储器的 SAM9260 板.	二进制文件
uboot_env_tiny.bin	UBoot 环境文件,用于小存储器的 SAM9260 板.	二进制文件
rootfs.jffs2.bin	Linux 的根文件系统, 附带测试程序, 用于大存储器的 SAM9260 板.	二进制文件
rootfs_tiny.jffs2.bin	Linux 的根文件系统, 附带测试程序, 用于小存储器的 SAM9260 板.	二进制文件
rootfs_bare.jffs2.bin	Linux 的根文件系统, 不附带测试程序,只有根用户	二进制文件
uboot.bin	UBoot 引导程序, 用于大存储器的 SAM9260 板.	二进制文件

uboot_tiny.bin	UBoot 引导程序, 用于小存储器的 SAM9260 板.	二进制文件
RFIDBootstrapEx.bin	Bootstrap 程序, 用于引导系统至 UBoot	二进制文件
uImage.bin	Linux 操作系统, 用于大存储器的 SAM9260 板.	二进制文件
uImage_tiny.bin	Linux 操作系统, 用于小存储器的 SAM9260 板.	二进制文件
SAM9260_AuxSerial.bin	板级支持包辅助处理器串口程序,请参考 5.1 节	二进制文件
SAM9260_AuxUSB.bin	操作辅助处理器 USB 程序, 请参考 5.5 节	二进制文件
SAM9260_GPIO.bin	板级支持包 GPIO demo 程序,请参考 3.1 节	二进制文件
SAM9260_I2C.bin	板级支持包 I2C demo 程序,请参考 3.4 节	二进制文件
SAM9260_MemoryMapping.bin	辅助处理器内存映射 demo 程序,请参考 5.4 节	二进制文件
SAM9260_Overclocking.bin	板级支持包超频 demo 程序,请参考 3.5 节	二进制文件
SAM9260_RTC.bin	板级支持包 RTC demo 程序	二进制文件
SAM9260_SPI.bin	板级支持包串行 Flash demo 程序,请参考 3.3 节	二进制文件
SAM9260_UART.bin	板级支持包 UART demo 程序,请参考 3.2 节	二进制文件
Firmware1228.dat	辅助处理器固件程序文件	二进制文件

1. 快速使用指南

本章旨在为用户提供一个快速使用本读卡机的指南，在操作本读卡机前用户需要了解操作本读卡机的基本规范.

1.1 硬件操作规范

-  ESD 防护

本产品对高静电势敏感，尽管大部分集成电路带有 ESD 防护，用户还是需要避免电路板接触高静电势，否则会损坏本读卡机. 当接触电路板之前，应采取接地带或类似防护措施.

- 加电操作

在电路板处在带电状态时，除了即插即用设备，禁止任何改变电路结构的操作. 因为这样的操作很可能造成电路板的损坏. 这类可能造成损坏的操作包括但不局限于:

1) **在加电的电路板上插入 JTAG 调试器**(正确的做法为将电路板断电，等待一段时间直到电容完全放电为止. 在 JTAG 调试插座上插入调试器，给电路板重新加电)
2) **在加电的电路板上进行焊接操作.**
3) **在加电的电路板上插入串口.** 严格的来说，串口是不允许热插拔的，对串口的热插拔有一定可能损坏电路.

- 绝对最大电源电压

本读卡机的绝对最大电源电压为 15V，任何超出该电源轨的输入电压将损坏读卡机.
工作电源电压为 12V，用户需连接一个 12V 直流稳压电源到读卡机的电源插座. 该电源需要至少能够提供 0.5A 电流. 考虑到浪涌等因素,输入电源范围允许在 12V±10%以内.

1.2 电路板加电

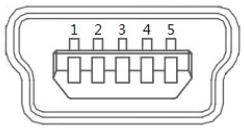
☑ **连接一个 12V 直流稳压电源到读卡机的电源插座. 如果电路板正常工作，电源指示 LED(D6)此时应该点亮.**

1.3 与辅助处理器通信

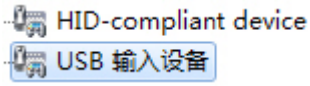
本读卡机有两个处理器，辅助处理器用于控制射频调制芯片和 SAM 卡。 我们首先试图使用辅助处理器，通过上位机软件直接给辅助处理器发送命令，看读卡机的读卡功能是否正常.

☑ **用 USB 电缆连接电路板的 MiniB 型 USB 插座和上位机 PC.**

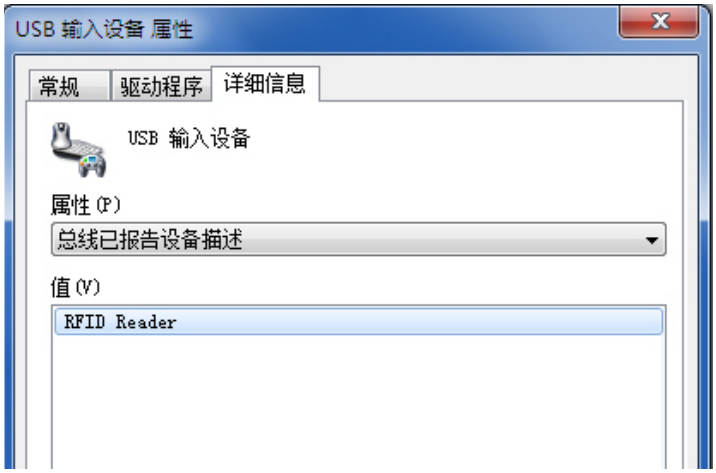
MiniB 型 USB 插座外形如下图，在电路板的右上角，插座编号为 J14.



由于辅助处理器的 USB 为一个人机接口设备 HID，无需为 PC 安装任何驱动程序即可直接使用. 当用 USB 连接到读卡机时，操作系统会自动安装驱动，如果遇到需要到网上查找驱动时，请选择否. 在驱动自动安装完成后，在设备管理器中会新增加两个设备：USB 输入设备和 HID-compliant device. 在 PC 的控制面板中打开设备管理器，可以看到新增加的设备.

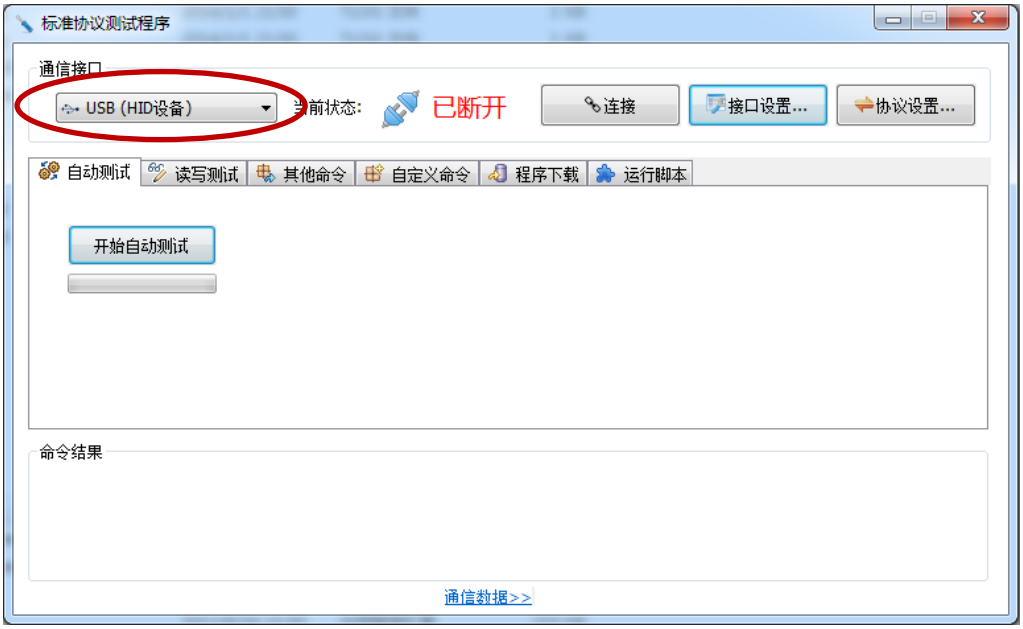


其中,在 USB 输入设备的总线报告描述中，值为 RFID Reader.

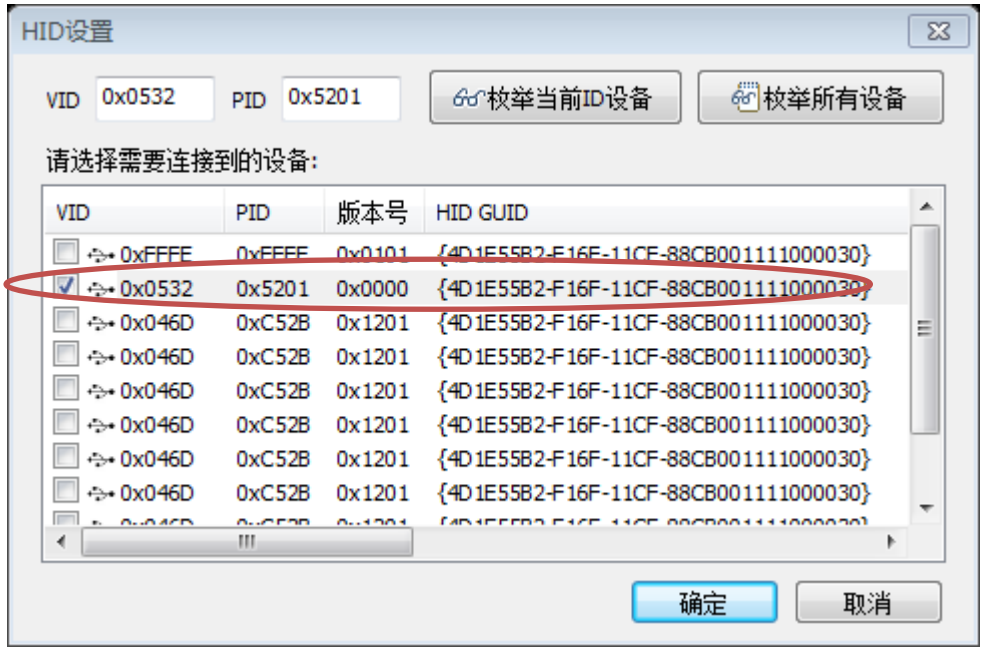


☑ 打开标准协议测试程序，连接 USB 设备.

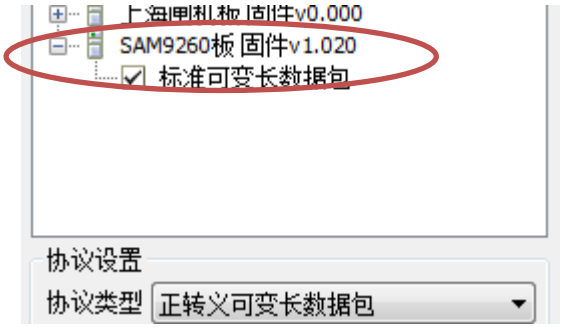
标准协议测试程序(*PC 文件夹下 StdProtocolTester.exe*)的主界面如下，在通信接口栏选择 USB(HID 设备).



单击接口设置按钮，会弹出如下对话框．单击枚举所有设备按钮，程序会自动寻找当前连接到 PC 的所有 HID 设备. 请选择 **VID 为 0x0532，PID 为 0x5201** 的那个设备．辅助处理器的 HID 参数被设置为 VID=0x0532，PID=0x5201.



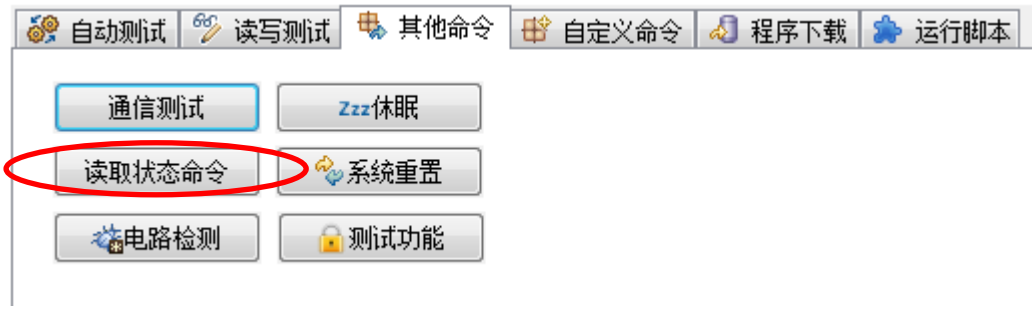
单击协议设置按钮，会弹出当前支持的协议．选择 SAM9260 板.



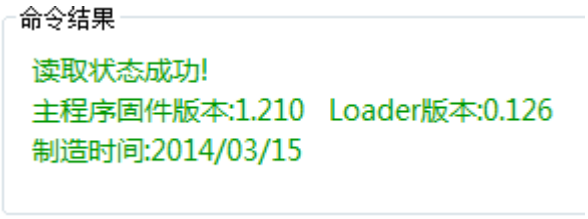
单击连接按钮，如果 USB 正确连接到 PC，此时状态应该为 已连接.

☑ 执行读取状态命令

选择下方标签中的其他命令，单击读取状态命令按钮.



此时命令结果应该会显示本读卡机的固件版本，制造时间等信息，如下图所示.

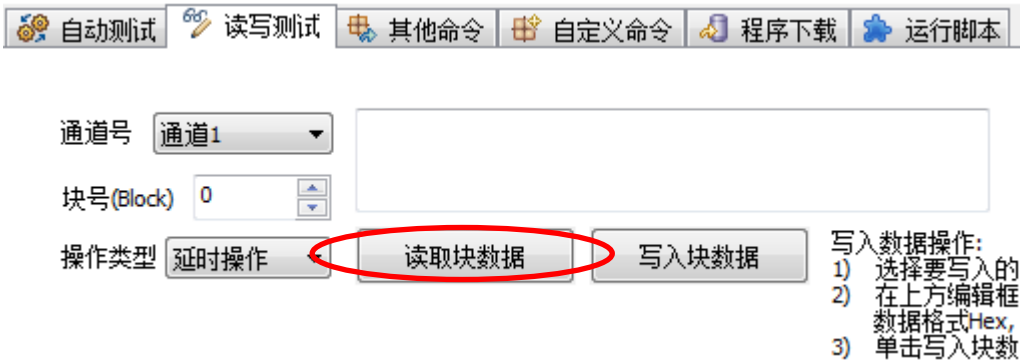


1.4 操作辅助处理器读卡

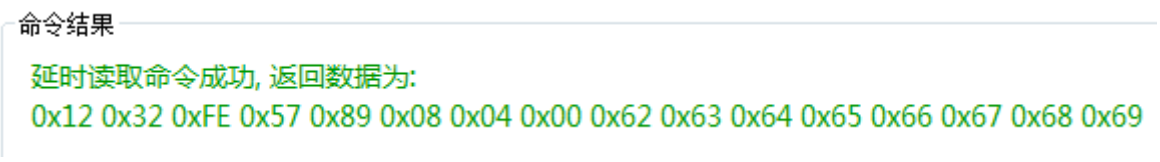
给读卡机断电，在SMA插座J21上插入天线插头，通过同轴电缆连接到天线. 在天线上放置一个Mifare M1卡. SMA插座外形如下图，在电路板的右下角. J21对应射频通道1，J19对应射频通道2，J20对应射频通道3. 由于我们插入的是J21，在下面操作中，选择的通道也应为通道1.

☑ 执行读卡命令

重复1.3节中的过程，通过USB连接到读卡机. 注意到每次读卡机在断电之后，用户都需要单击断开按钮，并重新进行连接. 选择下方标签中的读写测试，通道选择为通道1，块号选择为0，将一张密钥为0xFFFFFFFFFFFF的Mifare卡放在天线读写范围内，单击 读取块数据 按钮.



读取的数据将在下面的命令结果中显示，由于Mifare卡的第0个Block为卡片的UID，返回的数据应该为16字节ID号.

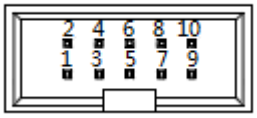


1.5　SRAM 中调试主处理器

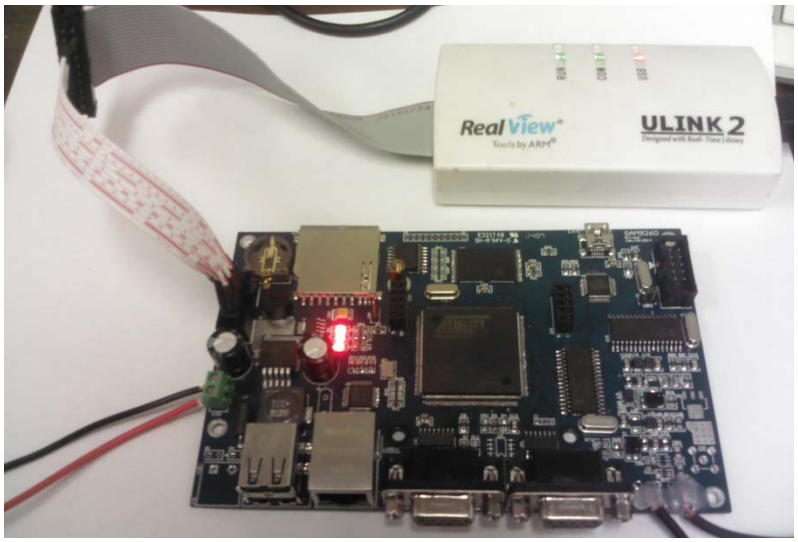
在下面的例子中，我们将使用 ULink2 作为 JTAG 调试器，用 Keil uVision4 软件作为集成开发环境．如果您使用其他的调试器或开发软件，请参考其说明手册．如果您没有 JTAG 调试器，请跳过此节．

☑ 连接 JTAG 调试器

JTAG 插座编号为 J7，是 DC3 2.54mm 间距，10pin 双排直插插针插座．其引脚定义如下：

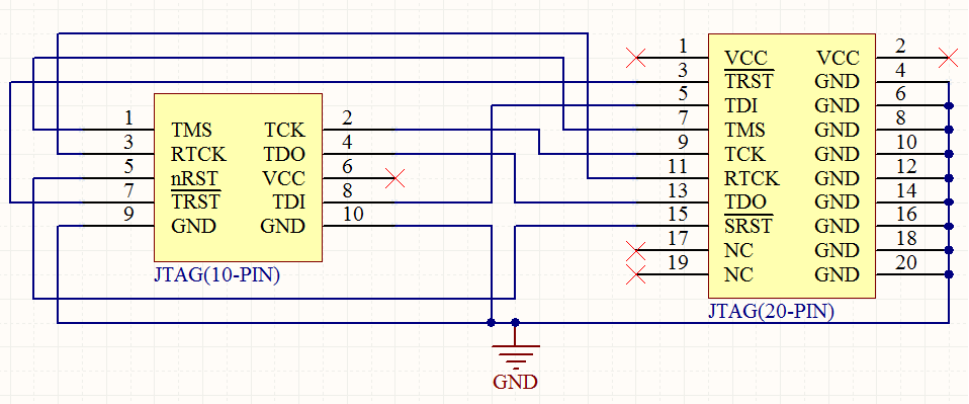


JTAG 插座的连接方式如下图所示：

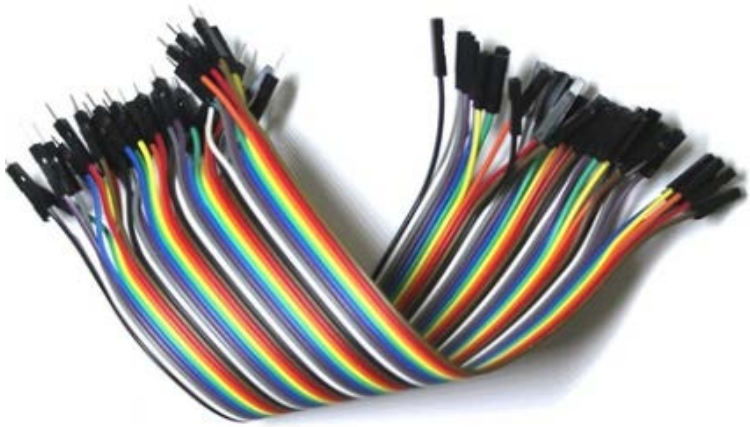


JTAG 插座引脚	定义	说明
1	TMS	JTAG 的 TMS 引脚
2	TCK	JTAG 的 TCK 引脚
3	RTCK	JTAG 的 RTCK 引脚
4	TDO	JTAG 的 TDO 引脚
5	nRST	读卡机系统复位引脚
6	VDD3.3	3.3V 电源,无过流保护
7	nTRST	JTAG 的复位引脚
8	TDI	JTAG 的 TDI 引脚
9	GND	数字地
10	GND	数字地

注意到 JTAG 插座并不是使用标准的 20pin 插座，用户需要自行制作一个 20pin 到 10pin 的针对孔转接插头，以满足上表所示的 JTAG 信号定义．一个典型的转接原理图如下所示，注意到 Vcc 是否连接取决于 uLink2 内部的跳线,此处该跳线连接到 3.3V ．



如果用户无法制作转接插头，可以考虑购买2.54mm间距的公对母IO杜邦跳线. 自行连接JTAG.

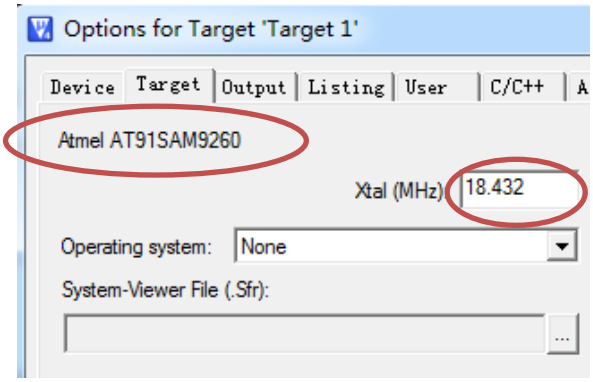


☑ 设置IDE

打开uVision，载入一个项目，此处我们载入GPIO项目(*GPIO文件夹下SAM9260_GPIO.uvproj文件*).

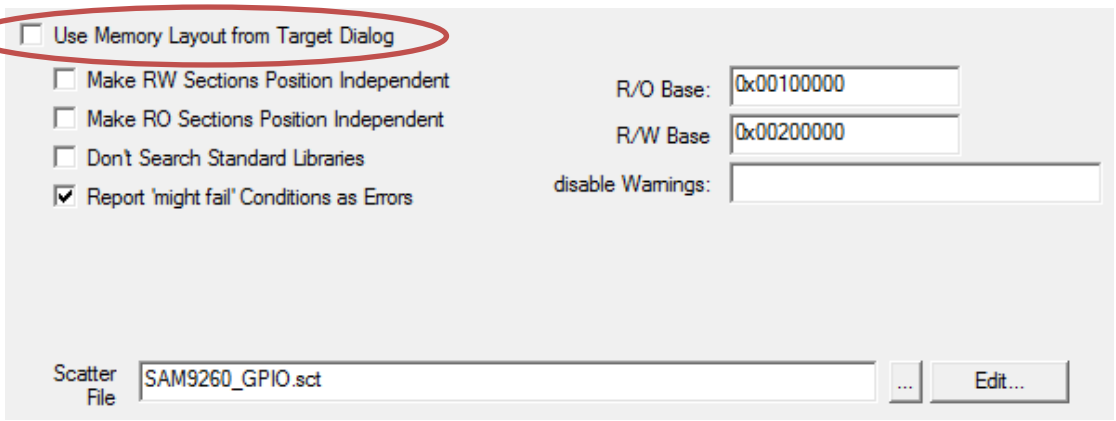
打开项目设置，用户需要以下几条信息的正确性:

- 处理器名为 AT91SAM9260
- 晶振频率为18.432



- 设置内存映射

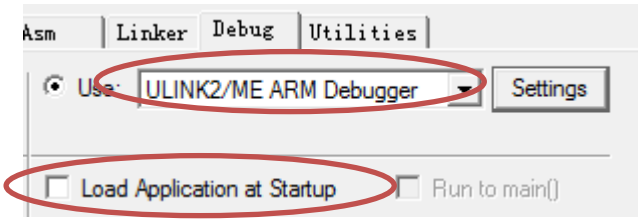
由于程序较小，我们在 SRAM 中直接运行该程序，为此我们需要设置 scatter 文件．注意，Use Memory Layout from Target Dialog 选项必须不被选中．



对于只在 SRAM 中调试的程序，其 Scatter 文件通常如下

```
Load_region 0x00200000 0x00001000 {
    Fixed_region 0x00200000 {
        *.o (RESET, +First)
        *(InRoot$$Sections)
        .ANY (+RO)
    }
    Relocate_region 0x00300000 0x00001000 {
        .ANY (+RW +ZI)
    }
}
```

- 调试器使用 ULINK2，注意这里不能选择启动时载入程序，稍后我们将手动载入程序．



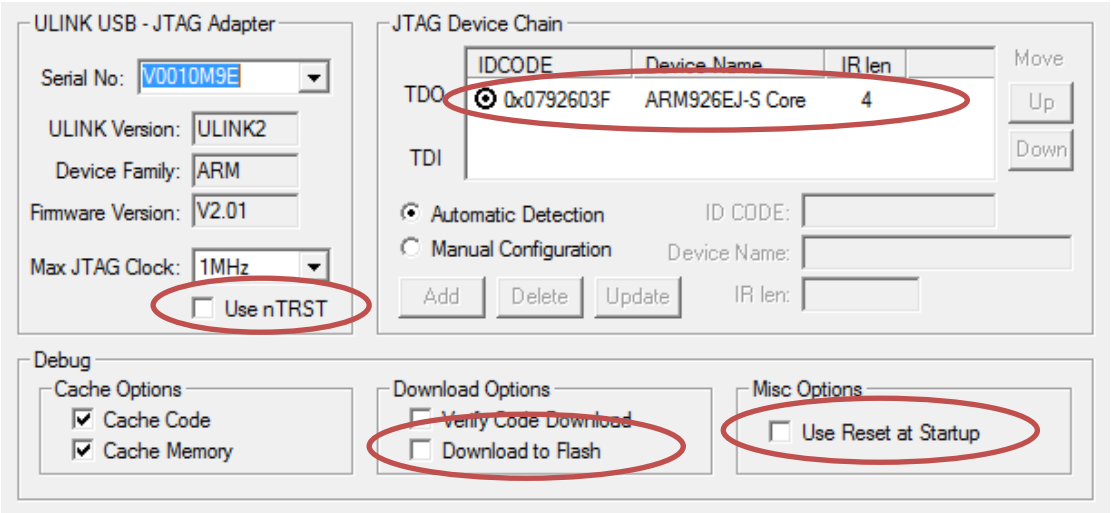
- ULINK2 的设置需要注意的是：

1） 不能使用 nTRST

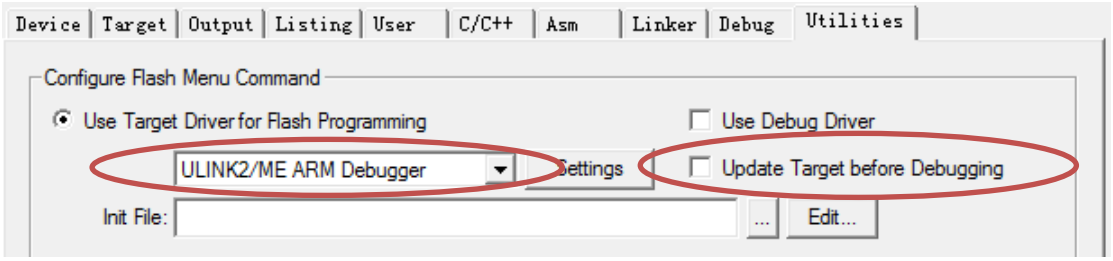
2） 不需要将程序下载至 Flash(AT91SAM9260 内部没有 Flash)

3） 启动时不进行复位

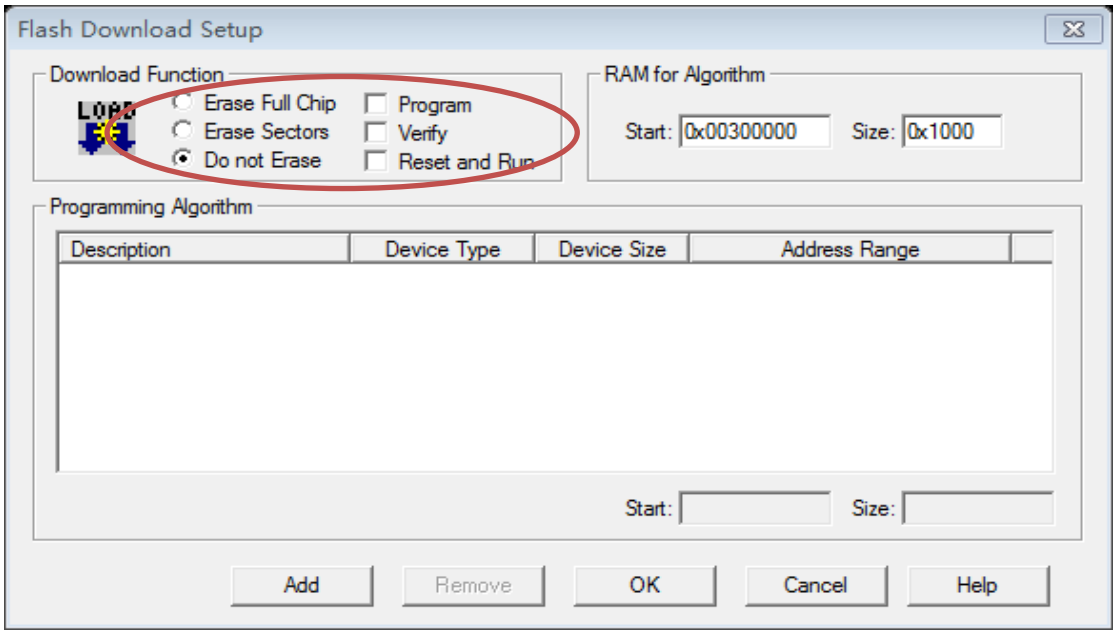
如果 ULINK 与芯片正确连接，就能够找到 ARM 内核．AT91SAM9260 芯片的 JTAG ID 为 0x0792603F，如下图所示．



- 烧写调试器也使用 ULINK2，注意不能选择 Update Target Before Debugging

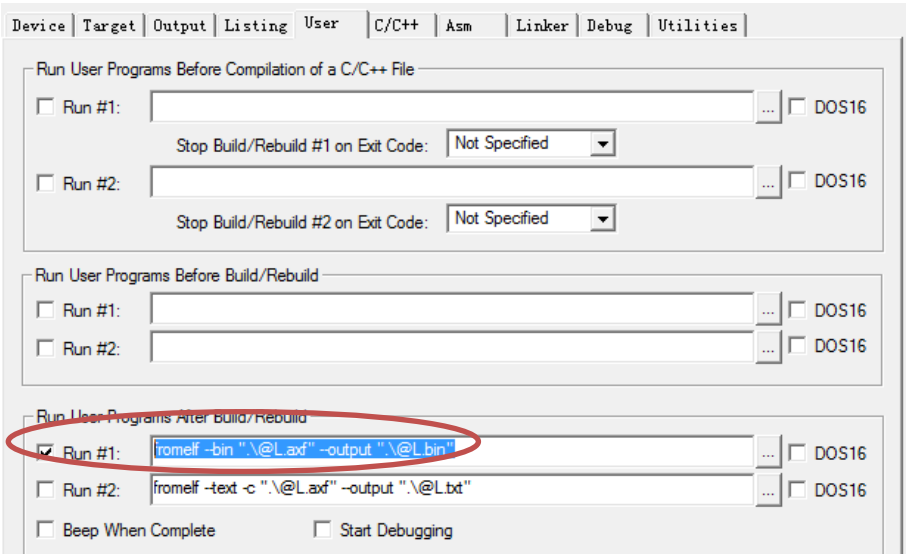


单击Flash编程设置，弹出如下对话框．由于我们将在RAM上运行程序，并不将程序写入 Flash 中．因此，此处选择不擦除，不进行编程以及验证．同时将下方的编程算法删除.



- 生成 bin 文件

在 Build 完成后，我们需要生成一个 bin 文件，在 Run#1 处添加 fromelf --bin ".\@L.axf" --output ".\@L.bin"



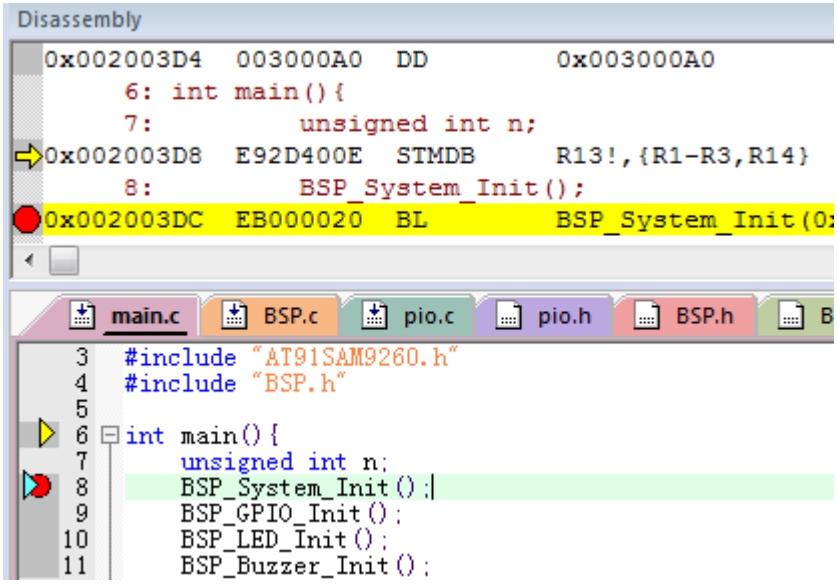
☑ 项目编译

编译该项目，如果成功，连接器会生成一个 axf 文件和一个 bin 文件．其中，axf 文件包含了调试信息，需要用 LOAD 命令载入的文件．而 bin 和 hex 则不包含调试信息，可以直接用于编程.

SAM9260_GPIO.axf	2014/2/25 17:32	AXF 文件	135 KB
SAM9260_GPIO.bin	2014/2/25 17:32	BIN 文件	3 KB
SAM9260_GPIO.hex	2014/2/25 17:32	HEX 文件	7 KB

☑ 项目调试

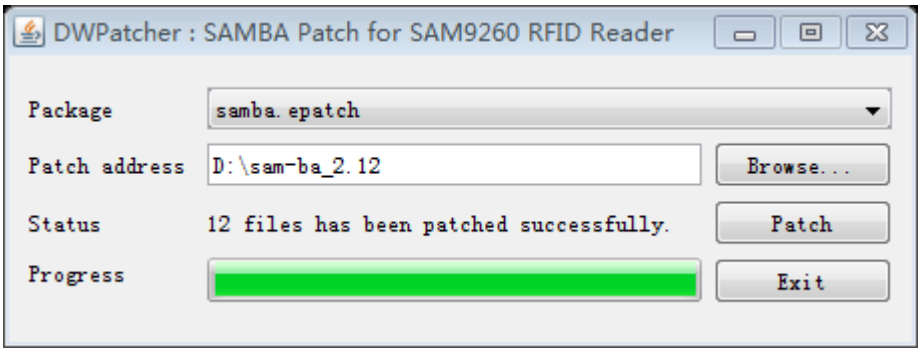
单击调试按钮  即可开始调试该程序，初始化脚本将自动加载 SAM9260_GPIO.axf 并运行到 main 函数以开始调试，用户单击全速运行  即可．GPIO 程序非常简单,在全速运行时用户可以看到两个 LED 在闪烁.



1.6　安装 SAM-BootAssist

SAMBA 为 Atmel 公司提供的一个烧写 NAND Flash 等存储器的功能程序。 本节将介绍 SAMBA 的安装，在下一节中，我们将使用 SAMBA 下载程序到 NAND Flash 中。首先运行 sam-ba_2.12.exe 安装程序(*文件夹 PC 中*)， 如下图。

在 SAMBA 安装完成后， 我们需要在 SAMBA 中添加读卡机板的设置。 运行 Patcher.jar(*文件夹 PC 中*)，设置 samba 的安装路径， 单击 Patch 按钮即可。



有关 SAMBA 程序的使用，请参考 SAM Boot Assistant User Guide(*Document 文件夹下 SAM-BA.pdf*)． SAMBA 支持通过 USB 下载，USB 的驱动在安装目录下．有关驱动的安装，请参考 SAMBA USB Notice (*Document 文件夹下 SAM-BA USB Notice.pdf*)
如果状态一直显示 Patching,不能安装补丁．请用管理员权限运行本程序．或用管理员权限打开 cmd，执行 java -jar Patcher.jar

1.7　将程序烧写入 Flash

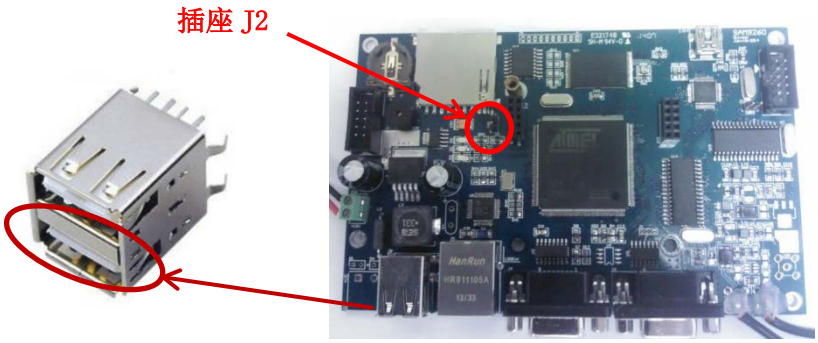
在上面的例子中，由于程序在 SRAM 中，在掉电之后，程序便不再执行．通常，我们需要将程序烧写入 Flash 中，使得再次加电时，程序会被 Bootstrap 从 Flash 中加载到 SDRAM 上并执行．载入 RTC 项目(*RTC 文件夹下 SAM9260_RTC.uvproj 文件*)，项目设置和 1.6 节相同，除了 Scatter 文件的内容．对于在 SDRAM 中运行的程序，其 Scatter 文件通常定义如下．

```
Load_region 0x21F00000 0x000C0000 {
    Fixed_region 0x21F00000 {
        *.o (RESET, +First)
        *(InRoot$$Sections)
        .ANY (+RO)
    }
    Relocate_region 0x21FC0000 0x00040000 {
        .ANY (+RW +ZI)
    }
}
```

即由 Bootstrap 将程序从 Flash 中写入 SDRAM 位于 0x21F00000 起始的 1M 字节中．并在 SDRAM 中执行程序．

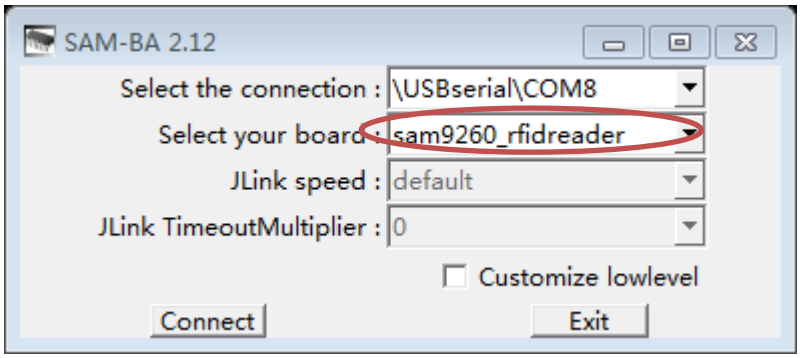
☑ 进入 ISP 状态

将电路板断电，用 USB 电缆将 PC 机和电路板连接起来．注意，USB 口为 USB 设备，即**插座编号为 J8 的双层 USB A 型插座的下面那一个**．将插座 J2 短路，插座 J2 的位置如下图所示．如果 J2 引脚为高点平（短路），RFIDBootstrapEx 将擦除 NAND Flash 的第一个 Block，使系统进入 SAMBA Monitor 状态。请注意，**插座 J2 短路必须是在系统加电前完成**.关于系统的引导顺序，请参考 2.1 节.

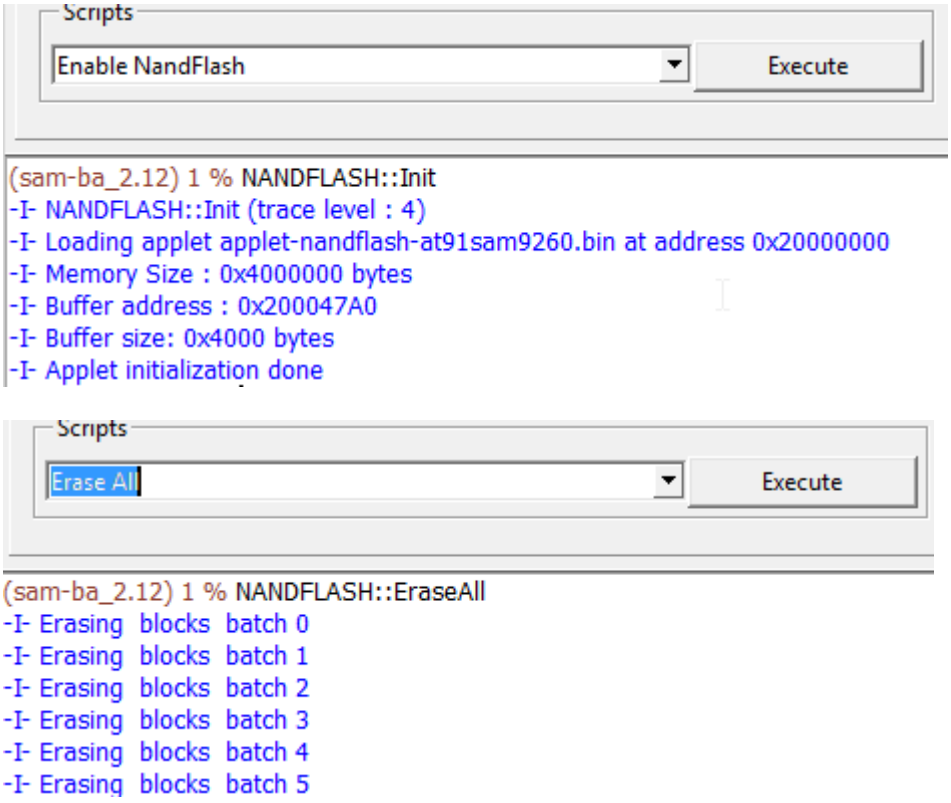


☑ 使用 SAM-BA 烧写程序

在加电后电源 LED D6 点亮约 5 秒后，用户可以将板子断电，将插座 J2 置为开路，等待 5 秒后再加电。打开 SAMBA，此时能够看到 USB 连接设备。在板子类型一栏中选择 sam9260_rfidreader，单击连接。



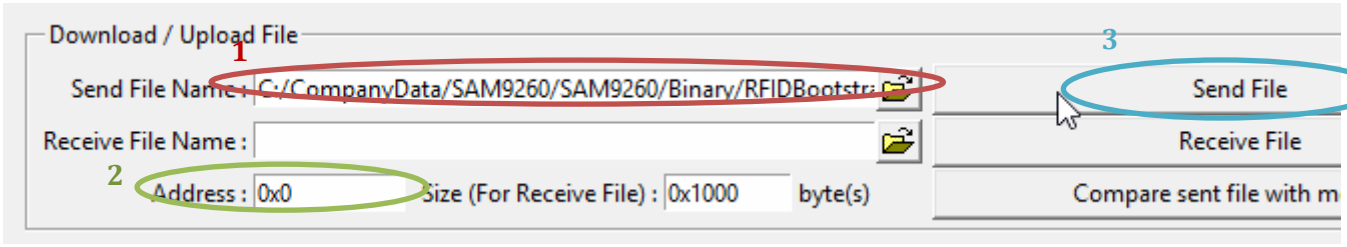
选择 NAND Flash 选项， 执行 Enable NAND Flash， 完成初始化。 再执行 Erase all，擦除整片 Flash。最后向 Flash 中烧入程序。



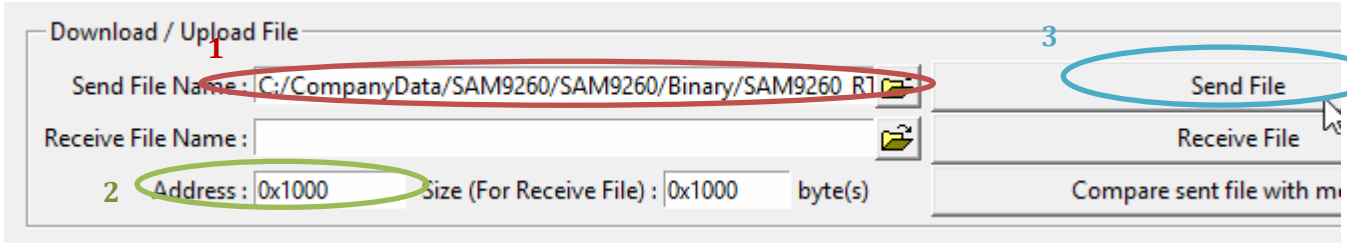
通过 SAMBA 向 Flash 的起始地址， 即偏移地址 0x00000000,烧写入 RFIDBootstrapEx.bin （*Binary 文件夹下 RFIDBootstrapEx.bin*）. 向 Flash 的偏移 0x00001000 处， 烧写入 SAM9260_RTC.bin 文件(*Binary 文件夹下 SAM9260_RTC.bin*). 这样系统在启动时会寻找到 Bootstrap， 而由 Bootstrap 负责将 SAM9260_RTC.bin 文件复制到 SDRAM 中运行.

起始地址	中止地址	最大尺寸	内容	典型尺寸
0x00000000	0x00000FFF	4Kbytes	RFIDBootstrapEx.bin	4Kbytes
0x00001000	0x0002BFFF	172Kbytes	SAM9260_RTC.bin	3.69Kbytes

烧入 RFIDBootstrapEx.bin



烧入 SAM9260_RTC.bin



警告:必须确保 Flash 的起始地址烧写的程序是 RFIDBootstrapEx.bin ，否则 ISP 引脚将失效，即使短路 J2 也不能再进入 SAMBA. 如果您不慎将 Flash 起始地址写入一个非引导类型程序，请参考 2.4 节引导恢复.

☑ 观察程序执行结果

将电路板断电，**将插座 J2 置为开路**，过一段时间后再加电，可以观察到 LED D2 每隔 1 秒钟闪烁一次. **警告:如果插座 J2 此时仍然短路，刚刚烧写的程序将被擦除，用户需要再次进入 SAMBA 烧写程序.**

1.8 烧写 Linux 系统

本节主要介绍如何将一个Linux操作系统完整的运行起来，而具体关于编译Linux系统，设置 U-boot 的过程将在第 4 章中介绍.

☑ 进入 ISP 状态

将电路板断电，用 USB 电缆将 PC 机和电路板连接起来. 注意，USB 口为 USB 设备，即**插座编号为 J8 的双层 USB A 型插座的下面那一个**. 进入 ISP 的过程与前一节一致，即短路 J2→加电 5 秒→ 断电 5 秒→J2 开路→加电→运行 SAMBA

☑ 使用 SAM-BA 烧写程序

通过 SAMBA 向 NAND Flash 烧写程序(*bin 文件在 Binary 文件夹下*)，具体的程序位置如下表所示.

大存储器(512Mb NAND Flash)

起始地址	内容	典型尺寸
0x00000000	RFIDBootstrapEx.bin	4Kbytes
0x00020000	uboot.bin	147Kbytes
0x00080000	uboot_env.bin	128Kbytes
0x00200000	uImage.bin	1.59Mbytes
0x00400000	rootfs.jffs2.bin	60Mbytes

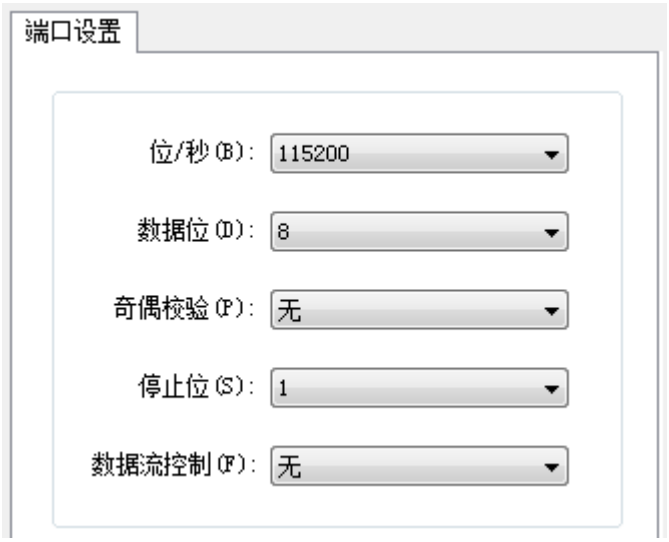
小存储器(256Mb NAND Flash)

起始地址	内容	典型尺寸
0x00000000	RFIDBootstrapEx.bin	4Kbytes
0x00001000	uboot_tiny.bin	147Kbytes
0x0002C000	uboot_env_tiny.bin	16Kbytes
0x00030000	uImage_tiny.bin	1.59Mbytes
0x00200000	rootfs_tiny.jffs2.bin	30Mbytes

警告:必须确保 Flash 的起始地址烧写的程序是 RFIDBootstrapEx.bin ，否则 ISP 引脚将失效，即使短路 J2 也不能再进入 SAMBA. 如果您不慎将 Flash 起始地址写入一个非引导类型程序，请参考 2.4 节引导恢复.

☑ 连接终端

在 Windows 操作系统下，我们使用超级终端作为终端程序. 将 PC 的串口与 SAM9260 板的串口 0(插座 J4)相连，波特率为 115200，8 位数据位，1 位停止位，无奇偶校验. 如下图所示.



在 Linux 系统下，我们使用 cu 作为终端，关于 cu 的安装和使用，请参考 4.3.2 节和 4.7.4 节.

☑ 启动 Linux 系统

在将程序烧入 NAND Flash 后，将系统断电 5 秒后再加电，此时终端会显示系统启动信息. 如果没有检测到用户输入，U-Boot 会在 3 秒后自动执行 boot 命令，启动 Linux 操作系统. 典型的 U-Boot 输出如下:

```
U-Boot 1.3.4 (Mar 15 2014 - 22:40:24)
SAM9260 RFID Reader Migration

Processor info:
    Core type: ARM926EJS
    Processor version: 0x02
    Processor type: AT91SAM9xx Series
SAM9260 RFID Reader board info:
    Firmware version: 1.210
    Loader version: 0.126
    Manufacture time: 2014/03/15
    Hardware signature:0x73483EEE29493EEEC286FF04C286B515
DRAM:  64 MB
NAND:  64 MiB
In:    serial
Out:   serial
Err:   serial
Net:   macb0
macb0: Starting autonegotiation...
macb0: Autonegotiation complete
macb0: link up, 100Mbps full-duplex (lpa: 0x45e1)
Hit any key to stop autoboot:  0
```

典型的 Linux 输出如下:

```
Starting kernel ...
Linux version 2.6.30 (eluneyun@ubuntu) (gcc version 4.8.1 (Sourcery
CodeBench
Lite 2013.11-33) ) #19 Fri Mar 14 19:16:48 PDT 2014
CPU: ARM926EJ-S [41069265] revision 5 (ARMv5TEJ), cr=00053177
CPU: VIVT data cache, VIVT instruction cache
Machine: Atmel AT91SAM9260-EK
Memory policy: ECC disabled, Data cache writeback
Clocks: CPU 198 MHz, master 99 MHz, main 18.432 MHz
Built 1 zonelists in Zone order, mobility grouping on.  Total pages: 16256
Kernel command line: mem=64M console=ttyS1,115200 root=/dev/mtdblock1 rw
mtdparts=atmel_nand:4M(bootstrap/uboot/kernel)ro,60M(rootfs)
rootfstype=jffs2
NR_IRQS:192
AT91: 96 gpio irqs in 3 banks
PID hash table entries: 256 (order: 8, 1024 bytes)
Console: colour dummy device 80x30
Dentry cache hash table entries: 8192 (order: 3, 32768 bytes)
Inode-cache hash table entries: 4096 (order: 2, 16384 bytes)
Memory: 64MB = 64MB total
Memory: 61620KB available (2832K code, 229K data, 100K init, 0K highmem)
Calibrating delay loop... 99.12 BogoMIPS (lpj=495616)
Mount-cache hash table entries: 512
CPU: Testing write buffer coherency: ok
net_namespace: 296 bytes
NET: Registered protocol family 16
AT91: Power Management
AT91: Starting after general reset
bio: create slab <bio-0> at 0
SCSI subsystem initialized
usbcore: registered new interface driver usbfs
usbcore: registered new interface driver hub
usbcore: registered new device driver usb
NET: Registered protocol family 2
IP route cache hash table entries: 1024 (order: 0, 4096 bytes)
TCP established hash table entries: 2048 (order: 2, 16384 bytes)
TCP bind hash table entries: 2048 (order: 1, 8192 bytes)
TCP: Hash tables configured (established 2048 bind 2048)
TCP reno registered
NET: Registered protocol family 1
NetWinder Floating Point Emulator V0.97 (double precision)
NTFS driver 2.1.29 [Flags: R/W DEBUG].
JFFS2 version 2.2. (NAND) (SUMMARY)  Â© 2001-2006 Red Hat, Inc.
msgmni has been set to 120
```

```
alg: No test for stdrng (krng)
io scheduler noop registered
io scheduler anticipatory registered (default)
atmel_usart.0: ttyS0 at MMIO 0xfefff200 (irq = 1) is a ATMEL_SERIAL
atmel_usart.1: ttyS1 at MMIO 0xfffb0000 (irq = 6) is a ATMEL_SERIAL
console [ttyS1] enabled
atmel_usart.2: ttyS2 at MMIO 0xfffb4000 (irq = 7) is a ATMEL_SERIAL
atmel_usart.3: ttyS3 at MMIO 0xfffb8000 (irq = 8) is a ATMEL_SERIAL
brd: module loaded
loop: module loaded
ssc ssc.0: Atmel SSC device at 0xc4900000 (irq 14)
Driver 'sd' needs updating - please use bus_type methods
macb macb: invalid hw address, using random
MACB_mii_bus: probed
eth0: Atmel MACB at 0xfffc4000 irq 21 (82:09:90:40:bc:17)
eth0: attached PHY driver[Davicom DM9161A](mii_bus:phy_addr=ffffffff:00,irq=-1)
NAND device:Manufacturer ID: 0x20,Chip ID: 0x76(ST Micro NAND 64MiB 3,3V
8-bit)
AT91 NAND: 8-bit, Software ECC
Scanning device for bad blocks
2 cmdlinepart partitions found on MTD device atmel_nand
Creating 2 MTD partitions on "atmel_nand":
0x000000000000-0x000000400000 : "bootstrap/uboot/kernel"
0x000000400000-0x000004000000 : "rootfs"
ohci_hcd: USB 1.1 'Open' Host Controller (OHCI) Driver
at91_ohci at91_ohci: AT91 OHCI
at91_ohci at91_ohci: new USB bus registered, assigned bus number 1
at91_ohci at91_ohci: irq 20, io mem 0x00500000
usb usb1: New USB device found, idVendor=1d6b, idProduct=0001
usb usb1: New USB device strings: Mfr=3, Product=2, SerialNumber=1
usb usb1: Product: AT91 OHCI
usb usb1: Manufacturer: Linux 2.6.30 ohci_hcd
usb usb1: SerialNumber: at91
usb usb1: configuration #1 chosen from 1 choice
hub 1-0:1.0: USB hub found
hub 1-0:1.0: 2 ports detected
Initializing USB Mass Storage driver...
usbcore: registered new interface driver usb-storage
USB Mass Storage support registered.
usbcore: registered new interface driver usbserial
USB Serial support registered for generic
usbcore: registered new interface driver usbserial_generic
usbserial: USB Serial Driver core
USB Serial support registered for ch341-uart
usbcore: registered new interface driver ch341
udc: at91_udc version 3 May 2006
```

```
mice: PS/2 mouse device common for all mice
rtc-at91sam9 at91_rtt.0: rtc core: registered at91_rtt as rtc0
IRQ 1/rtc0: IRQF_DISABLED is not guaranteed on shared IRQs
rtc-at91sam9 at91_rtt.0: rtc0: SET TIME!
i2c /dev entries driver
TCP cubic registered
NET: Registered protocol family 17
RPC: Registered udp transport module.
RPC: Registered tcp transport module.
rtc-at91sam9 at91_rtt.0: hctosys: unable to read the hardware clock
VFS: Mounted root (jffs2 filesystem) on device 31:1.
Freeing init memory: 100K
Starting logging: OK
Initializing random number generator... done.
Starting network...
Starting dropbear sshd: generating rsa key...
generating dsa key... OK
eth0: link up (100/Full)
Welcome to SAM9260_RFIDReader
SAM9260_RFIDReader login:
```

在 Linux 正常启动并加载根文件系统后，我们可以登录并使用系统. 在我们提供的根文件系统中，用户 root 的密钥为 szyxelec；用户 SAM9260_RFIDReader 的密钥为 szyxelec. 可以用这两个账户登录系统. 但是由于我们已经设置了自动登录，用户无需输入任何密钥即可进入系统.

执行 ls 命令可以看到有几个 demo 程序. 在本手册的后面将详细介绍每个 demo 的编程和使用. 现在先输入 ./gpio_arm_br 以执行程序 gpio_arm_br，这是一个 GPIO 驱动的 demo 程序，该程序将鸣响蜂鸣器，并使两个 LED 交替点亮 20 次.

```
$ ls
Firmware1228.dat                loader_arm_br
SAM9260_Firmware_Version.txt    sharedA.so.encrypt
autologin_arm_br                sharedB.so.encrypt
firmware_update.sh              socket_arm_br
gpio_arm_br                     spi_arm_br
gpio_sam9260.ko                 spi_sam9260.ko
hello_arm_br                    stdprocProgrammer_arm_br
libprotocol_arm_br.so           test_arm_br
loader.sh

$ ./gpio_arm_br
Pin SD_DETECT is high.
Pin IO3 is low.
```

在正确执行 gpio_arm_br 之后，执行程序 test_arm_br，该程序通过给辅助处理器发送命令，测试 Mifare 卡,CPU 卡和 PSAM 卡．这个程序首先尝试与辅助处理器通信，获取其固件版本，制造日期等信息.

```
$ ./test_arm_br
test: Serial port /dev/ttyS3 has been opened successfully.
test: APROM version:1.228
test: LDROM version:0.126
test: Manufacture time:FFFF/FF/FF
test: Place Mifare card on antenna 1, continue?(y/n)
```

然后程序会提示用户将一张密钥为 0xFFFFFFFFFFFF 的 Mifare 卡放在天线 1 上．天线 1 即射频通道 1，对应 SMA 插座 J21,靠板子外侧的那一个．卡放置完成后，用户需要输入 y 以继续测试．大致的测试过程为向扇区 1,块 0 写入数据 0xA0-0xAF，并通过不同的命令读出这个数据.

```
test: Step1. Read(Sector 0 Block 0)
test: The data in manufacture ID is 0x1232FE57890804006263646566676869
test: Step2. Write(Sector 1 Block 0)=0xA0-AF
test: Step3. Read(Sector 1 Block 0)
test: The data in block 4 is 0xA0A1A2A3A4A5A6A7A8A9AAABACADAEAF
test: Step4. Write(Sector 1 Block 3) Change password
test: Step5. Read(Sector 1 Block 0)
test: The data in block 4 is 0xA0A1A2A3A4A5A6A7A8A9AAABACADAEAF
test: Step6. Write password to EEROM
test: Step7. Read(Sector 1 Block 0)
test: The data in block 4 is 0xA0A1A2A3A4A5A6A7A8A9AAABACADAEAF
test: Step8. Request
test: ATQA: 0x4 SAK: 0x8
test: Step9. Read(Sector 1 Block 0)
test: The data in block 4 is 0xA0A1A2A3A4A5A6A7A8A9AAABACADAEAF
test: Step10. Init purse(Sector 1 Block 1)=200
test: Step11. Increment(Sector 1 Block 1)=12
test: Step12. Read purse(Sector 1 Block 1)
test: Purse block 5: value=212 address=0xc8
test: Step13. Write(Sector 1 Block 3) Change password back to FF
test: Step14. Halt
test: Place CPU card on antenna 1, continue?(y/n)
```

接下来,程序会提示将 CPU 卡放置在天线 1 上．由于不知道具体结构和密钥信息，CPU 卡的测试主要是执行 获取随机数和选择文件两个 APDU 命令.

```
test: Step15. Request
test: ATQA: 0x4 SAK: 0x28
test: Step16. Init CPU Tag
test: CPUTagDataRate: 0x0
test: The data in ATS is 0x107880A00220900000000000D68020DD
test: Step17. Get challenge
test: The data in APDU response is 0xFD3D50599000
test: Step18. Select file
test: The data in APDU response is 0x6F15840E315041592E53595321
test: Step19. Halt CPU tag
```

测试程序的最后将测试 SAM 卡，过程与 CPU 卡类似.

```
test: Insert PSAM card to slot 1, continue?(y/n)y
test: Step20. Reset SAM card
test: The data in SAM ATR is 0x3B9D180001130307FAED5713E6D
test: Step21. Get challenge
test: The data in SAM get challenge is 0x84B01464AD9000
test: Step22. Select file
test: The data in SAM select file is 0x6117
test: Step23. Get response
test: The data in SAM get response is 0xC06F15840E3150415S
test: Complete!
```

注意：在系统使用一段时间之后，有时会弹出如下警告

```
# JFFS2 notice: (314) check_node_data: wrong data CRC in data node
at 0x01abae7c: read 0x6b7053b2, calculated 0xea168339.
```

这通常是由于不正常关机时(例如突然断电)，未向 Flash 中写入剩余数据造成的. 通常这些警告可以被忽略. 为了减少类似的系统错误，必须使用 reboot 或 shutdown 等命令关闭系统.