



XR871 Quick Start Guide

Revision 1.0

May 9, 2017

Declaration

THIS DOCUMENTATION IS THE ORIGINAL WORK AND COPYRIGHTED PROPERTY OF XRADIO TECHNOLOGY ("XRADIO"). REPRODUCTION IN WHOLE OR IN PART MUST OBTAIN THE WRITTEN APPROVAL OF XRADIO AND GIVE CLEAR ACKNOWLEDGEMENT TO THE COPYRIGHT OWNER.

THE INFORMATION FURNISHED BY XRADIO IS BELIEVED TO BE ACCURATE AND RELIABLE. XRADIO RESERVES THE RIGHT TO MAKE CHANGES IN CIRCUIT DESIGN AND/OR SPECIFICATIONS AT ANY TIME WITHOUT NOTICE. XRADIO DOES NOT ASSUME ANY RESPONSIBILITY AND LIABILITY FOR ITS USE. NOR FOR ANY INFRINGEMENTS OF PATENTS OR OTHER RIGHTS OF THE THIRD PARTIES WHICH MAY RESULT FROM ITS USE. NO LICENSE IS GRANTED BY IMPLICATION OR OTHERWISE UNDER ANY PATENT OR PATENT RIGHTS OF XRADIO. THIS DATASHEET NEITHER STATES NOR IMPLIES WARRANTY OF ANY KIND, INCLUDING FITNESS FOR ANY PARTICULAR APPLICATION.

THIRD PARTY LICENCES MAY BE REQUIRED TO IMPLEMENT THE SOLUTION/PRODUCT. CUSTOMERS SHALL BE SOLELY RESPONSIBLE TO OBTAIN ALL APPROPRIATELY REQUIRED THIRD PARTY LICENCES. XRADIO SHALL NOT BE LIABLE FOR ANY LICENCE FEE OR ROYALTY DUE IN RESPECT OF ANY REQUIRED THIRD PARTY LICENCE. XRADIO SHALL HAVE NO WARRANTY, INDEMNITY OR OTHER OBLIGATIONS WITH RESPECT TO MATTERS COVERED UNDER ANY REQUIRED THIRD PARTY LICENCE.

Revision History

Version	Data	Summary of Changes
1.0	2017-5-9	Initial Version
	2017-8-7	修改配置文件说明

Table 1-1 Revision History

Contents

Declaration.....	2
Revision History	3
Contents.....	4
Tables.....	5
Figures	6
1 开发环境搭建	7
1.1 开发环境选择	7
1.2 选择 Toolchain.....	7
1.3 Linux Toolchain 安装与配置.....	7
1.4 Windows Toolchain 安装与配置.....	8
1.4.1 安装 Cygwin 开发环境.....	8
1.4.2 Windows Toolchain 安装与配置	9
2 SDK 说明.....	11
2.1 目录结构	11
2.2 SDK 相关配置文件.....	12
2.3 编译示例工程	14
3 烧录工具使用	15
3.1 烧录工具界面	15
3.2 固件下载.....	16
3.2.1 进入升级模式	16
3.2.2 升级固件	16

Tables

Table 1-1 Revision History 3

Figures

图 1-1 Cygwin 安装包内容.....	8
图 1-2 Cygwin 安装窗口	8
图 1-3 Cygwin 安装过程	9
图 1-4 Cygwin 安装项目检查.....	9
图 2-1 image.cfg 文件内容.....	13
图 3-1 烧录工具界面	15

1 开发环境搭建

1.1 开发环境选择

Windows 环境: Cygwin + GCC

Linux 环境: Ubuntu14.2 以上 + GCC

1.2 选择 Toolchain

Toolchain: gcc-arm-none-eabi-4_9-2015q2

Windows 版本

https://launchpad.net/gcc-arm-embedded/4.9/4.9-2015-q2-update/+download/gcc-arm-none-eabi-4_9-2015q2-20150609-win32.zip

Linux 版本

https://launchpad.net/gcc-arm-embedded/4.9/4.9-2015-q2-update/+download/gcc-arm-none-eabi-4_9-2015q2-20150609-linux.tar.bz2

1.3 Linux Toolchain 安装与配置

- 使用相对路径

1. 下载 Linux 版本的工具链压缩包 gcc-arm-none-eabi-4_9-2015q2-20150609-linux.tar.bz2。
2. 进入控制台程序, 在 HOME 目录下创建 tools 目录, 并把压缩包拷贝到此路径。

```
#cd ~  
#mkdir tools  
#cd tools
```

然后将下载的压缩包放入 tools 目录, 并进入到 tools 目录解压。

```
#tar -xvf gcc-arm-none-eabi-4_9-2015q2-20150609-linux.tar.bz2 ./
```

3. 将解压的目录名更改为 gcc-arm-none-eabi-4_9-2015q2, 使之与 sdk-code/gcc.mk 里的 CC_DIR 变量保持一致。

- 使用环境变量

1. 下载工具链压缩包并解压到个人指定的安装目录
2. 修改系统启动配置文件 ~/.profile (没有的话自己添加), 将 GCC 执行文件路径添加到 PATH 环境变量中,

如下所示：

```
export PATH=$PATH:/usr/gcc-arm-none-eabi-4_9-2015q2/bin
```

3. 在 console 执行 source ~/.profile 脚本，更新环境变量配置（下次重新登录时不需要）。
4. 修改 sdk-code/gcc.mk，设置 CC_PREFIX 变量为 arm-none-eabi-（去除路径使用环境变量配置）。

1.4 Windows Toolchain 安装与配置

1.4.1 安装 Cygwin 开发环境

1. Cygwin 开发环境下载 Cygwin.7z

百度网盘下载: <http://pan.baidu.com/s/1jH4lAw2>

Mega 下载: https://mega.nz/#!YfggBKzA!n6ThMpUb_5w0rrc39m9Jj5180tDvilrx8Y_BOrN8NoM

注：以上环境安装包直接安装即可，也可下载官方 Cygwin 再独立安装工具链工具。

2. 解压 Cygwin 压缩包，得到以下文件

名称	大小
lib	286 679 1...
setup-x86.exe	751 104
setup.bat	1 259
setup.log	68 263
setup.log.full	2 414

图 1-1 Cygwin 安装包内容

3. 进入 Cygwin 目录并双击运行 setup.bat，运行后将出现以下窗口




图 1-2 Cygwin 安装窗口

4. 根据提示输入安装路径，例如安装至 C 盘根目录: C:\

(注意：安装包存放路径及安装路径不可存在中文以及空格，安装路径必须是已存在的路径，否则无法安装)

5. 输入路径后得到询问是否安装，输入“Y”开始安装，输入“N”退出安装。输入 Y 后在弹出的对话框



中选择“是”则开始安装，并显示安装进度，安装完毕后桌面会产生图标 ，表示 Cygwin 组件已安装完成。安装确认及进度如下图所示：

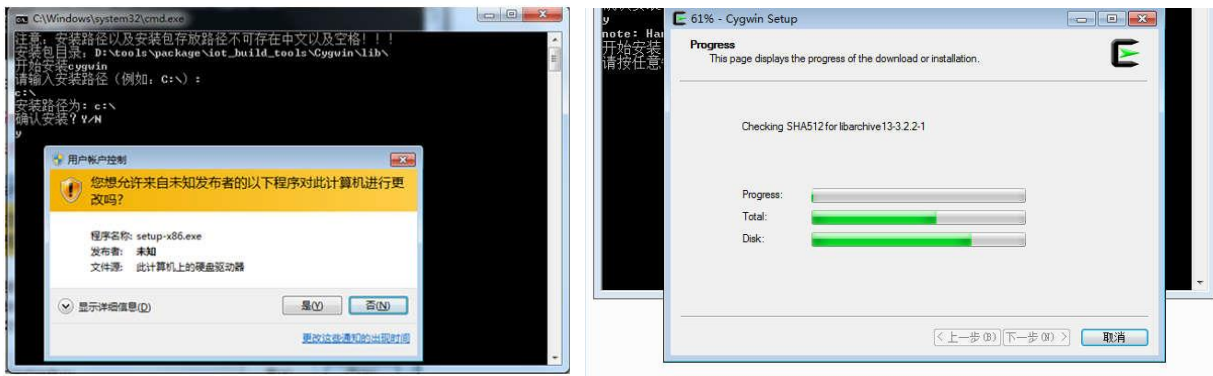


图 1-3 Cygwin 安装过程

6. 验证 Cygwin 安装完整性，双击桌面 Cygwin 图标进入 Cygwin 终端，输入 `cygcheck -c` 命令，系统将列出已安装组件，使用此工具包安装的 Cygwin 自动安装了常用工具，满足开发的基本需求。

```

Tiuyitong@PCLiuYitong ~
$ cygcheck -c
Cygwin Package Information
Package      Version      Status
_automake    000727-1    OK
_update-info-dir 01317-1    OK
alternatives 1.3.30c-10  OK
base-cygwin  3.3-1       OK
base-files   4.2-3       OK
bash         4.1.17-9    OK
binutils     2.24.51-6   OK
bzip2        1.0.6-2     OK
ca-certificates 2.2-1       OK
cmake        2.8.9-2     OK
coreutils    8.23-4      OK
crypt        1.2-1       OK
csih         0.9.8-2     OK
cygrunsrv    1.61-1      OK
cygutils     1.4.14-1    OK
cygwin       1.7.33-1    OK
cygwin-devel 1.7.33-1    OK
dash         0.5.8-3     OK
diffutils    3.3-2       OK
editrights   1.03-1      OK
file         5.20-1      OK
findutils    4.5.12-1    OK
gawk         4.1.1-1     OK
gcc-core     4.8.3-4     OK
gcc-java     4.8.3-4     OK
getent       2.18.90-4   OK
git          2.1.1-1     OK
grep        2.21-1      OK
groff        1.22.2-2    OK
  
```

图 1-4 Cygwin 安装项目检查

1.4.2 Windows Toolchain 安装与配置

- 使用相对路径配置

1. 下载 Windows 版本的工具链压缩包 `gcc-arm-none-eabi-4_9-2015q2-20150609-win32.zip`。
2. 双击桌面 Cygwin 图标进入 Cygwin 终端，进入控制台程序，在 HOME 目录下创建 `tools` 目录，并把压缩包拷贝到此路径。

```

#cd ~
#mkdir tools
#cd tools
  
```

然后将下载的 zip 压缩包解压到 tools 目录。

3. 将解压的目录名更改为 gcc-arm-none-eabi-4_9-2015q2, 使之与 sdk-code/gcc.mk 里的 CC_DIR 变量保持一致。

- 使用环境变量

1. 下载工具链压缩包并解压到个人指定的安装目录
2. 修改系统启动配置文件 ~/.profile (没有的话自己添加), 将 GCC 执行文件路径添加到 PATH 环境变量中, 如下所示:

```
export PATH=$PATH:/usr/x86_64_linux_gcc_4_8_4_Cygwin/bin
```

3. 在 console 执行 source ~/.profile 脚本, 更新环境变量配置 (下次重新登录时不需要)。
4. 修改 sdk-code/gcc.mk, 设置 CC_PREFIX 变量为 arm-none-eabi- (去除路径使用环境变量配置)。

2 SDK 说明

2.1 目录结构

```

|-- README
|-- bin
|-- config.mk
|-- gcc.mk
|-- include
|-- lib
|   |-- libairkiss_aes.a
|   |-- libamr.a
|   |-- libamren.a
|   |-- libcedarx.a
|   `-- libmp3.a
|-- project
|   |-- audio_player
|   |-- common
|   |-- evb_main_board_sensor_v1_0
|   |-- image_cfg
|   |-- linker_script
|   `-- wlan_demo
|-- src
|   |-- Makefile
|   |-- audio
|   |-- console
|   |-- driver
|   |   |-- chip
|   |   `-- component
|   |-- efpg
|   |-- fs
|   |-- image
|   |-- kernel
|   |   |-- FreeRTOS
|   |   `-- os
|   |-- libc
|   |-- net
|   |-- ota
|   |-- pm
|   `-- sys
`-- tools

```

airkiss 库，音频编解码库

工程目录

功能源码目录

BSP 和外设驱动

文件系统

系统内核

打包及烧录工具

2.2 SDK 相关配置文件

1. 配置文件列表

sdk-code/config.mk	- 全局配置文件，配置一些全局参数，一般不需要修改
sdk-code/gcc.mk	- 全局的编译规则，一般只需要修改CC_DIR的定义，将其指向对应路径
sdk-code/src/lib.mk	- 模块（library）通用编译规则，一般不需要修改
sdk-code/project/prjconfig.mk	- 工程默认配置选项，一般不需要修改
sdk-code/project/project.mk	- 工程通用编译规则，一般不需要修改
sdk-code/project/xxx/gcc/localconfig.mk	- 工程配置选项（最终定义，覆盖其他定义）
sdk-code/project/xxx/gcc/Makefile	- 工程Makefile
sdk-code/project/xxx/prj_config.h	- 工程功能属性配置文件，由其他c文件所用

2. 工具链配置

相关文件

- `sdk-code/gcc.mk`

如果 Toolchain 安装在\$HOME/tools 下，则按照下方示例配置工具链路径，如果不是，则需要修改 gcc.mk，将 CC_DIR 指定到安装目录的对应中，如下所示：

```
4 CC_DIR = ~/tools/gcc-arm-none-eabi-4_9-2015q2/bin
5 CC_PREFIX = $(CC_DIR)/arm-none-eabi-
```

3. 工程配置

相关文件

- 公共配置文件 `sdk-code/config.mk`
- 工程配置文件 `sdk-code/project/prjconfig.mk`
- 工程私有配置文件 `sdk-code/project/xxx/gcc/localconfig.mk`

其中，只需要修改 localconfig.mk 文件即可，相关配置如下：

```
1 #
2 # project local config options, override the common config options
3 #
4
5 # -----
6 # board definition
7 # -----
8 __PRJ_CONFIG_BOARD := xr871_evb_main
9
10 # -----
11 # override global config options
12 # -----
13 # set y to enable bootloader and disable some features, for bootloader only
```

```

14 # export __CONFIG_BOOTLOADER := y
15
16 # set n to disable dual core features, for bootloader only
17 # export __CONFIG_ARCH_DUAL_CORE := n
18
19 # -----
20 # override project common config options
21 # -----
22 # support both sta and ap, default to n
23 __PRJ_CONFIG_WLAN_STA_AP := y
24
25 # support xplayer, default to n
26 # __PRJ_CONFIG_XPLAYER := y
27
28 # enable XIP, default to n
29 # __PRJ_CONFIG_XIP := y
30
31 # enable OTA, default to n
32 # __PRJ_CONFIG_OTA := y

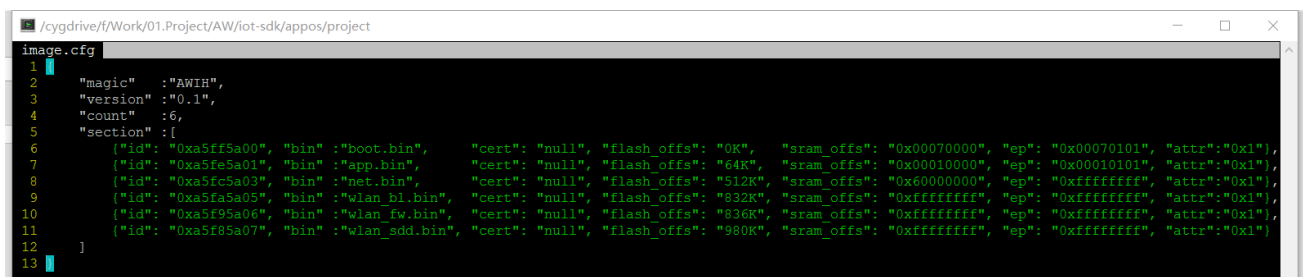
```

4. 打包配置

相关文件

- 通用配置文件 `sdk-code/project/image_cfg/xr871/image.cfg`
- 专用配置文件 `sdk-code/project/xxx/gcc/Makefile` 中指定

Image.cfg 是工具编译完成之后打包固件使用的配置文件，内容如下：



```

1
2 "magic" : "AWIH",
3 "version" : "0.1",
4 "count" : 6,
5 "section" : [
6   {"id": "0xa5ff5a00", "bin": "boot.bin", "cert": "null", "flash_offs": "0K", "sram_offs": "0x00070000", "ep": "0x00070101", "attr": "0x1"},
7   {"id": "0xa5fe5a01", "bin": "app.bin", "cert": "null", "flash_offs": "64K", "sram_offs": "0x00010000", "ep": "0x00010101", "attr": "0x1"},
8   {"id": "0xa5fc5a03", "bin": "net.bin", "cert": "null", "flash_offs": "512K", "sram_offs": "0x60000000", "ep": "0xffffffff", "attr": "0x1"},
9   {"id": "0xa5fa5a05", "bin": "wlan_bl.bin", "cert": "null", "flash_offs": "832K", "sram_offs": "0xffffffff", "ep": "0xffffffff", "attr": "0x1"},
10  {"id": "0xa5f95a06", "bin": "wlan_fw.bin", "cert": "null", "flash_offs": "836K", "sram_offs": "0xffffffff", "ep": "0xffffffff", "attr": "0x1"},
11  {"id": "0xa5f85a07", "bin": "wlan_sdd.bin", "cert": "null", "flash_offs": "980K", "sram_offs": "0xffffffff", "ep": "0xffffffff", "attr": "0x1"}
12 ]
13

```

图 2-1 image.cfg 文件内容

Image.cfg 字段内容含义如下：

- magic* - 固件魔数，表头标识，固定为AWIH
- version* - 版本号
- count* - 打包文件列表即 *section* 字段中文件的数目
- section* - 文件列表

- `id` - 该段固件的 ID，ID 定义参考在 `sdk-code/include/sys/image.h` 中定义
- `bin` - 该段固件 `bin` 文件名称，表示当前 `bin` 文件所代表的固件，名字不能超过 16 字符
- `cert` - 该固件 OEM 签名证书，如果没有证书填 `null`，名字不能超过 16 字符
- `flash_offs` - 该段固件存放在 FLASH 中的位置偏移，以 KB 为单位
- `sram_offs` - 该段固件加载到 SRAM 中的地址偏移，以 B 为单位，`0xFFFFFFFF` 为无效值
- `ep` - 该段固件的 ENTRY POINT，`0xFFFFFFFF` 为无效值，使用时忽略
- `attr` - 表示该段固件属性，暂未使用，模式配 1

5. 链接配置脚本

相关文件

- 通用配置文件 `sdk-code/project/linker_script/gcc/xr871/appos.ld`
- 专用配置文件 `sdk-code/project/xxx/gcc/Makefile` 中指定

链接脚本文件的主要目的是描述如何将输入文件中的 `section` 映射到输出文件中，并控制输出文件的存储布局。目前应用系统 `code` 和 `data` 如果不超过 448KB 的 SRAM 空间可参考同路径的 `appos.ld` 文件，如果超过则需要开启 FLASH XIP 功能，此时 `ld` 文件请参考同路径的 `appos_xip.ld` 文件，对 MEMORY 定义和 SECTION 中的 `.xip` 字段进行内容修改即可。

2.3 编译示例工程

注：此章节内容后续可能会变化。

在 `sdk-code/project` 预置一些示例工程，典型的如基于开发板的音频应用工程 `audio_player` 和控制应用工程 `evb_main_board_sensor_v1_0`。编译工程时进入到工程目录（以 `audio_player` 为例）`sdk-code/project/audio_player/gcc`，然后执行以下命令即可：

```
# Clean 工程
make clean          - 清理工程目录下的临时文件
make lib_clean     - 清理 src 下的临时文件

# 编译工程
make lib           - 编译 src 下的各模块，输出为 lib 文件
make              - 编译工程应用，输出 axf 或 elf 文件
make image        - 打包成 xr-system.img 固件
```

具体细节内容请参见工程目录下的 `Makefile`。

3 烧录工具使用

3.1 烧录工具界面



图 3-1 烧录工具界面

烧录工具为 PhoenixMC，其功能为通过串口将指定的合法固件文件烧录到进入烧录模式的目标主机中，其功能点包括串口设置，固件选择，平台选择，升级确认，状态显示，固件信息等，其中：

- 串口设置 - 刷新可更新已连接串口设备列表，勾选对应的 COM 口进行固件升级，当串口设备改变时，点击“刷新”按钮进行列表刷新。串口的波特率目前有三种选择：9600，115200，921600，速度越快，烧录时间越少，具体情况根据具体的 FLASH 表现而定。
- 固件选择 - 点击“选择固件”按钮会弹出对话框，选择指定的 img 文件，选定后在固件信息栏里面会显示固件的详细信息，另外通过拖拽的方式将固件直接拖入工具界面也可以达到同样的效果。
- 选择平台 - 目前此工具支持 XR871 和 XR871-OTA 两种固件，由于 OTA 固件稍有差异，因此在此有所区分（未来可能会改变）。
- 烧写状态 - 此进度条指示当前选定的串口的设备烧写进度，在状态提示栏里会提示当前的烧写步骤，当烧写成功时，进度条会达成 100% 的进度并显示为绿色，当烧写失败时，进度条显示为红色并报告错误。

PhoenixMC 工具具备可同时烧录多个目标主机的功能，将 PC 通过多个串口与多个设备相连，并刷新串口列

表，同时勾选多个对应的 COM 口，点击升级时将同时对多个设备进行烧录，在烧录的过程中，点击不同的 COM 口时，在进度条上将显示各自 COM 口的烧录进度，最终实现多个设备的烧写。

3.2 固件下载

3.2.1 进入升级模式

下载固件之前首先要保证目标设备进入升级模式，设备进入升级模式有以下几种方法：

1. 未烧录过的设备（FLASH 上无有效内容）在上电后自动进入升级模式；
2. 烧录过的设备且能够正常启动并进入控制台的，通过在控制台输入 `upgrade` 命令使设备进入升级模式；
3. 通过将 STRAP IO PB02，PB03 同时置为低电平并 RESET 设备，使设备进入升级模式，进入后需释放 PB02，PB03 IO 使其处于上拉状态；
4. 在上电过程中短接 FLASH 的 MISO 信号到 GND，使系统启动失败后自动进入到升级模式；

3.2.2 升级固件

当设备进入到升级模式后，即可通过 PhoenixMC 工具对其烧录，步骤如下：

1. 使用串口连接 PC 与设备，在 PhoenixMC 中的串口列表中勾选对应的 COM 口，并设置波特率为 921600；
2. 点击“选择固件”按钮选择想要烧录的固件文件；
3. 选择对应的 IC 平台如 XR871；
4. 点击“升级固件”按钮进行设备升级，直至进度信息显示 Upgrade OK 烧录成功；