**ZHUHAI JIELI TECHNOLOGY CO.,LTD**

**Introduction of AC692X\_SDK**

**Introduction of AC692X\_SDK User's Manual**

**Rev 1.1 - Sunday, April 08, 2018**

**This translated version is for reference only, and the English version shall prevail in case ofany discrepancy between the translated and English versions.**

**CONTENTS**

Chapter 1 Quick Using Instructions of SDK Development Package............................ 5

1.1 Purpose of writing ...........................................................................................5

1.2 Installation of IDE Development Tools... ........................................................6

1.3 Opening of AC692x\_SDK Project ......... ....................................................... 7

1.4 Introduction to the Project Contents................................................................ 8

1.5 Some Common Setup Instructions........................................ ......................... 9

1.6 Program Download Instructions.....................................................................11

Chapter2 Upgrade Instructions........................ ............................................................12

2.1 Download Upgrade Instructions.....................................................................12

Chapter3 VM Instructions for Using............................................ ...............................17

3.1 VM Overview............... ................................... .............................................17

3.2 Basic using of VM......................................................................................... 18

Chapter4 AUX mode............................ .......................................................................22

4.1 Overall Design............................................................................................... 22

Chapter5 Bluetooth Certification Instructions.............................................................23

5.1 FCC Certification Instructions ................ .................................................... 23

5.2 BQB.......... ................................................................................................... .26

Chapter6 Instructions for Use Bluetooth Development.............................................. 27

6.1 Terms and Abbreviations................................................................................27

6.2 Development Instructions..............................................................................28

Chapter7 Instructions for Using Music Development .................................................31

7.1 Overall Design................................................................................................31

7.2 Overall Architecture Design................ ..........................................................32

7.3 Decoding Channel Instructions......................................................................34

7.4 Partial API Function Instructions............... ...................................................35

Chapter8 Instructions for Using Radio Development..................................................36

8.1 Overall Design ..............................................................................................36

8.2 Radio Design Instructions..................... ........................................................37

8.3 Built-in Radio Searching Parameters Instructions.........................................38

Chapter9 Instructions for Using Clock Development .................................................40

9.1 Introduction to Chapters................ ................................................................40

9.2 Overall Design ..............................................................................................40

9.3 Design Instructions of System Entry Module............................................... 43

9.4 Design Instructions of Setting the time module.... ....................................... 44

9.5 Design Instructions of Alarm Module..... ......................................................45

9.6 Instructions of RTC Module Special Function ..................... ........................46

Chapter11 Instructions for Using PC Slave Development.......................................... 48

11.1 Overall Design .................... ........................................................................48

13.2 Design Instructions of System Entry Module............................................. 50

13.3 Design Instructions of Card Reader Module................................................51

13.4 Design Instructions of HID Operation Module............................................52

13.4 Design Instructions of USB\_SPK Module...... ........................................... 53

13.5 PC Detection Function .................... ...........................................................54

Chapter14 F1A Warning Tone file.............. .................................................................55

14.1 Overview of F1A Warning Tone .................................................................55

Modify log

|  |  |  |
| --- | --- | --- |
| Version | Date | Description |
| 1.1 | 2018 / 4 /08 | Add notes for using the VM |
| 1.0 | 2018 / 3 / 30 | AC692x User’s Manual |
| Update: | * Establish the initial version * Defining the document grid |  |

Chapter 1 Quick Using Instructions of SDK Development Package

1.1 Purpose of writing

This document mainly describes how to use the AC692x\_SDK development package and some problems that should be paid attention to in the development. It provides a reference for the user to carry out the secondary development, including:

Development environment building, installation steps of IDE, certification methods and other pre-development preparation;

Contents structure, structure description of the project, files that the user is allowed to modify, and files that the user is not allowed to modify;

1.2 Installation of IDE Development Tools

* Compile environment installation
*  Toolchain installation

To develop the AC692x you must reinstall the toolchain:

1, codeblocks-16.01mingw-setup, the same as AC690x

2, jl\_toolchain\_pi32v2\_lto\_2.1.6.exe

*  Installation tutorial:

Jl\_toolchain\_pi32v2\_lto\_2.1.6

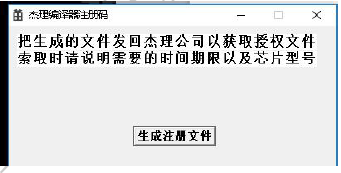
The tool needs to be registered, otherwise it will go wrong in the link step and display "No Such File"

*  Registration tool:
* Generate a key file

In order to register, it first needs to generate a key file to feed back to Jieli Technology.

Start --> Program --> JL toolchain --> Generate License Key File --> "Generate Registration File"

Then save the key file and send it to Jieli Technology, and explain the items you need to use (AC54, AC691X, AC692X, etc.) or backend (PI32, PI32V2, Q32S)



* Import lic file

Start --> Program --> JL toolchain --> Import License File --> Open License File

Then select the lic file you have obtained.



1.3 Opening of AC692x\_SDK Project

1. Before you open the project, make sure you have installed the newly released IDE and tool chain of Jieli Technology.

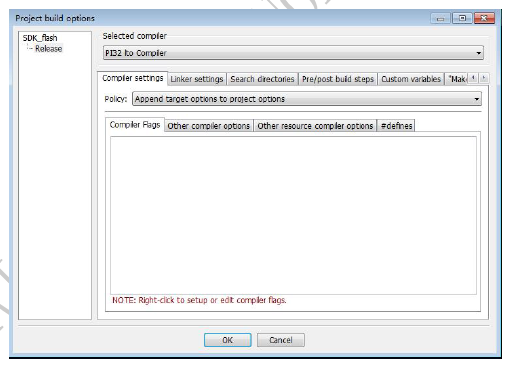
2. Double-click the ac69\_app.cbp project file in the apps/ directory, or drag the ac69\_app.cbp file to the CodeBlocks shortcut to open the AC692x SDK project.

Note: CodeBlocks is installed for the first time. If the compilation fails, you may need to specify the compiler (if the compilation is normal, you do not need to specify). Method: Right click on the project name - 》build options - 》Compiler settings, as shown below:

1. Before opening the project, make sure that the latest IDE and toolchain released by Jieli Technology have been installed, because the AC692x has some new compilation features.So be sure to install the compiler version (jl\_toolchain\_pi32v2\_lto\_2.1.6.exe) before it can be used normally, otherwise the SDK can compile normally but will execute incorrectly.

2. Double-click the sdk.cbp project file in the apps/ directory, or drag the sdk.cbp file to the CodeBlocks shortcut to open it, and then the SDK project will be opened.

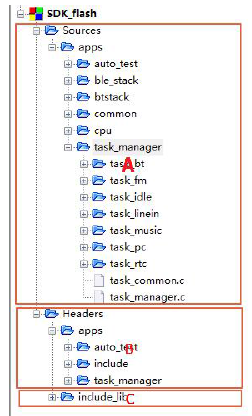
Note: CodeBlocks is installed for the first time. If the compilation fails, you may need to specify the compiler (if the compilation is normal, you do not need to specify). Method: Right click on the project name - 》build options - 》Compiler settings, as shown below:



After you have selected it, you can recompile it.

1.4 Introduction to the Project Contents

After the project opens, the project directory looks like this:



Screenshot area A: the user can see and modify some. C files, the user can also add their own c file

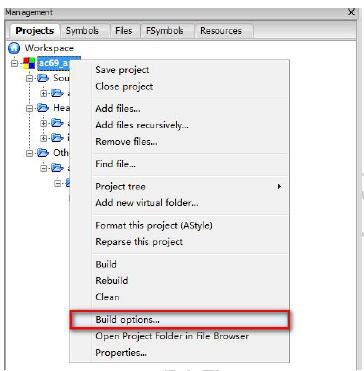
Screenshot area B: apps directory provides external application header files, users can easily modify

Screenshot C area: include\_lib provides the header files used by the library, closely related to the compilation of the library, customers try not to modify the contents

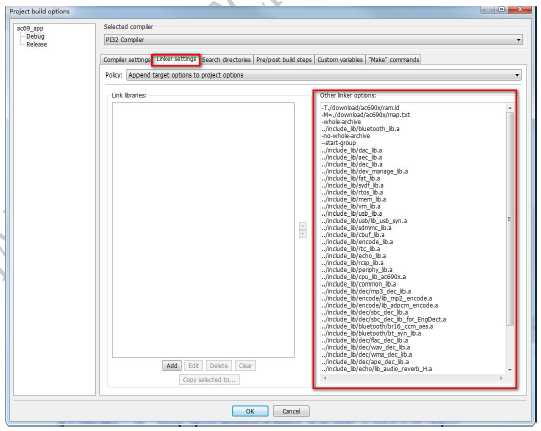
The Map.txt file in the apps\download\post\_build\flash directory is updated every time it is compiled. The content is mainly the mapping relationship between engineering functions and some variables. sdk.ld mainly indicates the usage of ram.

1.5 Some Common Setup Instructions

Select the project, select the right button to set, as shown below:



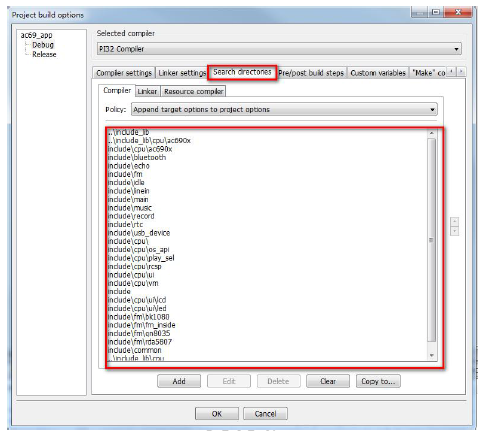
After clicking the red border, the following settings interface will appear, select the Linker settings to set the library file:



Note: When adding a library file (changing the library file), you need to re-rebuild, just modify the .c or .h file to execute the build.

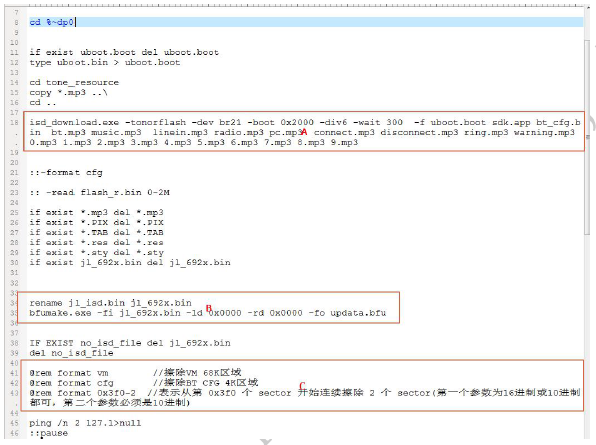
The red box on the right is the added library file

Select Search directories to set the path:

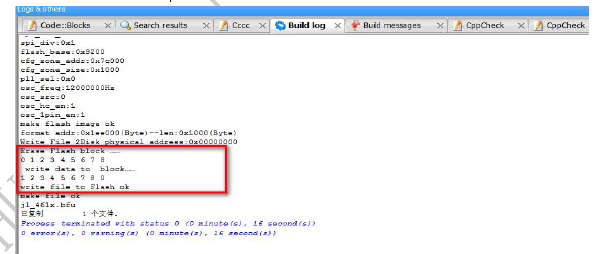


1.6 Program Download Instructions

Download.bat file is a download batch file, the corresponding configuration, (A section is responsible for downloading the program to flash,B is responsible for generating and writing files and upgrading files (bfu):



Finally, write file to Flash ok is displayed, which means the download is successful.



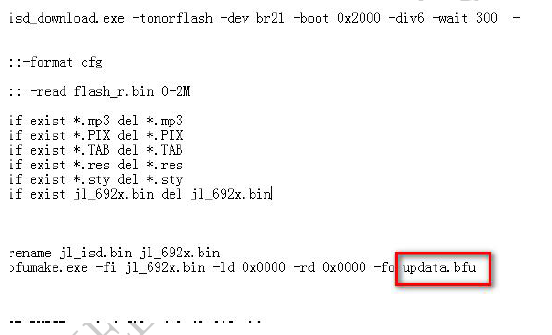
Note: After the compiler is compiled, it will be downloaded directly. Before downloading, you need to reset the flash to power on.(In the development board, press the flash reset button to power on, or press the flash reset button and then press the Reset button to reset). After the hard disk symbol is displayed on the pc, you can download it. After the download is complete, please power off and restart.

Chapter2 Upgrade Instructions

2.1 Download Upgrade Instructions

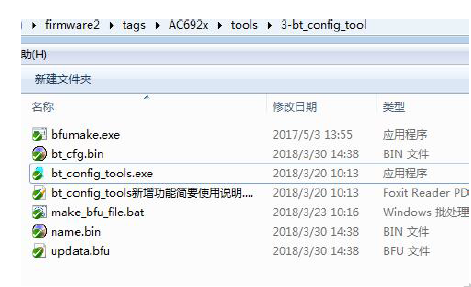
Download and download the 692x series, including: PC online upgrade, U disk / TF card upgrade, PC forced upgrade, test box Bluetooth wireless upgrade. [Note] flash partition: code area, configuration area, vm area.

* Generate upgrade file updata.bfu for the upgraded code. The following figure AC692x\flash\download.bat generates the updata.bfu file to be upgraded. This upgrade file can be updated to update the relevant code area.

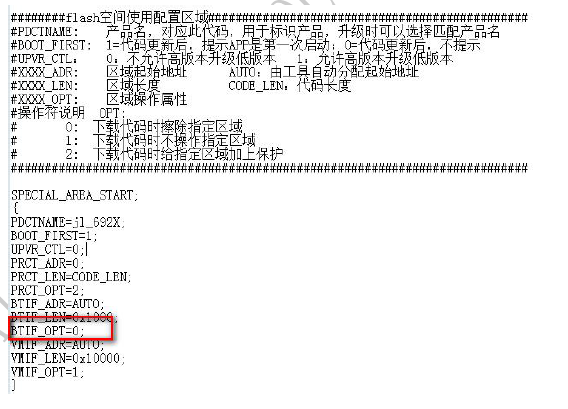


Writing the bin file and upgrading the updata.bfu file must be generated if the prototype or development board is not connected to the computer, ie offline generation!

* Generate updata.bfu for separately upgrading the Bluetooth name . The following figure modifies the Bluetooth name that needs to be upgraded via the tool bt \_ config \_ tool . exe , save . Then double - click the batch make \_ bfu \_ file . bat , and finally generate the upgrade file updata.bfu of the upgrade Bluetooth name . ( This upgrade file is upgraded by erasing the flash configuration area , and then writing the relevant information to the configuration area )



* The upgrade requires erasing the flash configuration area and VM. For example, to erase the flash configuration area, you need to update the configuration information and modify AC692x\ flash\ isd\_tools.cfg as shown in figure 1 below. The resulting updata.bfu will be erased with flash configuration information, erased when upgraded.



# Operator Description OPT:

# 0: Erase the specified area when downloading the code

# 1: Do not operate the specified area when downloading code

# 2: Protect the specified area when downloading the code

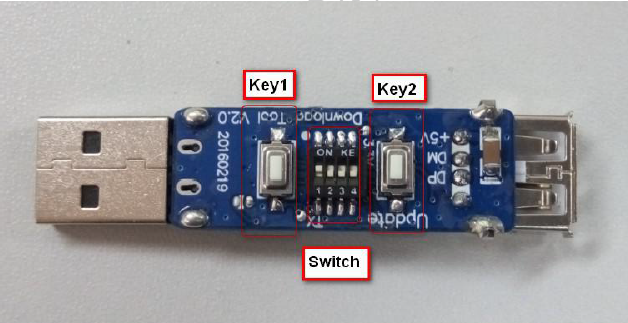
* PC forced upgrade

When using our custom USB tool, press the "update" button and the blue light will be on. When the prototype is powered off, insert the USB upgrade tool and the blue light is off. You can enter the upgrade mode, click batch AC692x\flash\download.bat to download. This upgrade mode is mainly used when the chip can't run or burn out the program!

 The upgrade tool uses the usb interface. When it is used normally, the female port is connected to the computer, and the male port is connected to the usb port of the prototype. The V2.0 version adds a DIP switch for subsequent upgrades and compatibility with older versions.

|  |  |  |
| --- | --- | --- |
| DIP switch | Operating mode | Description |
| 0-0-0-0 | AC460X | Used to upgrade the AC460X series chip |
| 1-0-0-0 | AC69XX\_USB slave upgrade | Used to upgrade AC9XX series chips |
| Other | Reserved | Reserved |

As shown in the following figure:



* After the USB cable is connected to this tool, the default mode is normal and no operation is performed, which is equivalent to a USB extension cable.
* Description of the upgrade steps

1. Power off the prototype completely.

2. Connect the female interface of this tool to the computer with a USB cable and wait for led1 to light up.

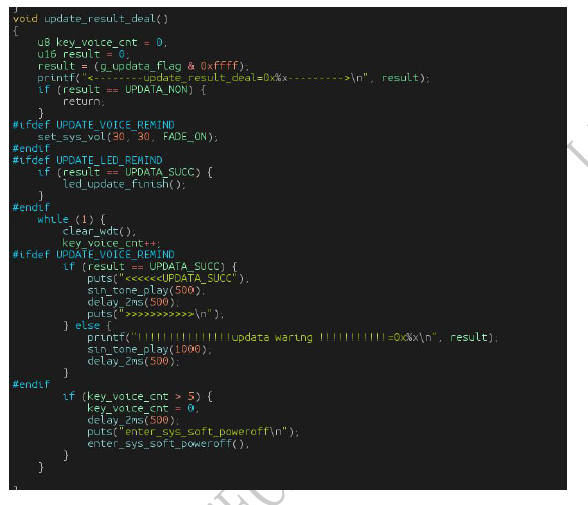
3. Press key2, the upgrade indicator led2 is always on (if you press key2, led2 does not light, please check the hardware or retry step 2 after power off)

4. Insert the upgrade tool male port into the usb of the prototype, turn the prototype on, and the prototype has entered the upgrade mode. (If the prototype is turned on normally and cannot enter the upgrade mode, please try step 1)

5. After successfully entering the upgrade mode, the code can be automatically upgraded and downloaded to the prototype after recompiling the code.

* PC online upgrade
* First, the prototype program needs to have the PC slave function, and then when the boot, connect PC, into PC upgrade mode.Click batch to download AC692x\ flasht\ download.bat. The upgrade requires erasing the flash configuration area and the VM reference described above for configuration.
* U disk / TF Card upgrade
* First of all, the prototype program needs to have U-disk function or TF card function. Then boot, enter music mode, detected U disk or TF card with updata.bfu files into upgrade mode. The upgrade requires erasing the flash configuration area and the VM reference described above for configuration.
* Test box Bluetooth wireless upgrade
* First of all, the prototype program needs Bluetooth mode to connect the state, then in the boot with the upgrade file updata.bfu TF card inserted into the Jieli one-drag two test box, and then use the test box to connect the prototype. Upgrade transfer into upgrade mode (the test box needs to be updated to AC690x\_1T2 Test Box V1.0.9 to support opening updata.bfu for upgrade). For more details, please refer to (Description for the AC690x\_1T2 test box. pdf )

For the U disk/TF card/Bluetooth wireless upgrade, the user can modify the app/cpu/updata.c of sdk in the result of the upgrade completion. As shown in the figure below, after the upgrade is completed, the light will be on and there will be a button sound. If the upgrade is wrong, there will be an alarm sound. Users should remember to test the relevant upgrade function during development to prevent the program from being mistaken during mass production and cannot be saved.



Chapter3 VM Instructions for Using

3.1 VM Overview

Overview:

VM (virtual memory) is a virtual memory system used to store information to flash devices. Due to the limited number of memory devices, the system can balance the wear and tear of memory devices and extend the life of the device.

The basic use of VM is mainly divided into the following four steps:

* Initialize the VM: For any VM operation, you must ensure that the VM has been initialized.
* Apply VM: Get a VM handle after applying for a VM. This handle provides VM read and write usage.
* Read VM: Operate through the VM handle.
* Write VM: Operate through the VM handle.

The VM has a minimum total capacity of 8K and a maximum total capacity of 128K. Modify the configuration of the isd\_tools.cfg file.

3.2 Basic using of VM

* Apply VM:

|  |  |
| --- | --- |
| Function prototype | Void vm\_open\_all(void) |
| Functional description | Apply for data\_len length vm storage space |
| Parameter description | Index: the serial number of the applied VM area  Data\_len: the size of the applied VM area |
| Return | Operate on the handle in the applied VM area |

void vm\_open\_all(void);

Before using VM storage, it must first perform open operations, and use vm\_opoen or vm\_open\_all. Which needs to be noted that the new version of the VM does not need to pass the length parameter when it is opened.

enum {

VM\_REMOTE\_DB = 1,

VM\_REMOTE\_DB\_END = (VM\_REMOTE\_DB + 20),

//vm\_start index

VM\_SYS\_VOL,

VM\_SYS\_EQ,

VM\_DEV0\_BREAKPOINT,

VM\_DEV1\_BREAKPOINT,

VM\_DEV2\_BREAKPOINT,

VM\_DEV3\_BREAKPOINT,

VM\_MUSIC\_DEVICE,

VM\_PC\_VOL,

VM\_FM\_INFO,

VM\_PHONE\_VOL,

VM\_BT\_STEREO\_INFO,

VM\_BT\_OSC\_INT\_R,

VM\_BT\_OSC\_INT\_L,

//------Please add a new VM item under the secondary line ------//

VM\_TEST,

VM\_MAX\_INDEX,

};

* VM Reading：

|  |  |
| --- | --- |
| Function prototype | vm\_err vm\_read(u8 index, void \* data\_buf，u16 len) |
| Functional description | read VM |
| Parameter description | index: the corresponding index enumeration variable table when applying vm  \*data\_buf: read buff pointer |
| Return | len: successful  ret < 0: error |

Usage example:

err = vm\_read(vm\_hdl\_vol,&sys\_info\_var.vol, 1);

* VM Writing：

|  |  |
| --- | --- |
| Function prototype | vm\_err vm\_write(u8 index ,const void \*data\_buf, u16 len) |
| Functional description | write VM |
| Parameter description | index: the corresponding index enumeration variable table when applying vm  \*data\_buf: the data pointer to be written |
| Return | len: successful  ret < 0: error |

Usage example:

err = vm\_write(vm\_hdl\_vol,&sys\_info\_var.vol);

About the VM read and write operations of the AC692X Note:

This version of VM read and write requires a length parameter. Only one last valid data is stored inside each VM.

The read and write length is greater than the internal data length, and only the effective length is returned. The read and write fetch length is less than the internal access length, and the read and write length is returned.

VM and DAC:

The DAC can continue to work while the VM is being read/defragmented

In the vm\_init\_api function, the parameter is 0:vm. The DAC is not allowed to work during the operation. The parameter is 1:vm. The DAC is allowed to work during the work.

It should be noted that if the DAC is allowed to work, all functions or constants called in dac\_isr\_cb need to be placed in the audio\_text section (using the AT\_AUDIO definition)

Chapter4 AUX mode

4.1 Overall Design

* System:

This note is primarily based on the SDK development kit to implement AUX functionality.

The main features of the AUX application include:

 aux input channel selection

 aux\_adc conversion

* Overall architectural design:

The AUX task is divided into two functional modules:

1. Input channel selection : aux input with amux0 , amux1 , amux2 and dac \_ amux . Each channel can be input only one way through one of the channels and is suitable for IC packages with less IO . The dac \_ amux is the two output channels of the dac , one of which serves as an input and the other as an output function .
2. Aux\_adc conversion: this function is designed to achieve digital conversion of aux analog input. Energy sampling can be realized, and aux input can be sampled by ad, and then digital sound effect can be processed.

* Application startup

The aux mode enters in two ways:

1 Insert the aux data line, jump into aux mode, if you turn on the aux insert detection function

2 Switch through the mode button, directly into aux mode, the premise is to turn off the aux insert detection function

* Application exit:

When the Mode key is pressed, the aux data line is pulled out, or other tasks need to be activated, the main thread will force to exit the aux mode, releasing the Aux task resource and turning off the warning tone.

Chapter5 Bluetooth Certification Instructions

5.1 FCC Certification Instructions

* Fcc authentication configuration:
* The macro that opens Bluetooth test mode is described in the file Bluetooth\_api.h as shown in figure 1 below：

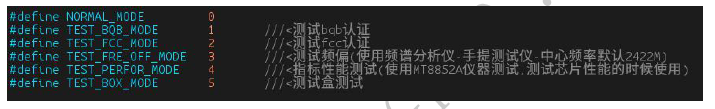


Figure 1

* To enable FCC authentication, you need to configure the BT\_MODE macro to TEST\_FCC\_MODE in sdk\_cfg.h, as shown in the following figure. (Note that after the function is enabled, the PC online upgrade and upgrade should be upgraded by using the USB port to boot. The download.bat should use the -erase configuration to erase all flashes.)



Figure 2

* Sdk's fcc authentication uses the USB port (DP is TX\ DM RX) by default to do uart serial communication with the computer (configured in uart.c as shown in figure 3 below) to ensure that the serial port communicates properly with the computer. When sdk\_cfg.h needs to shield all programs using the usb port (as shown in figure 4 below), pay attention to the packaging of each chip. If the usb port is tied to the other feet, it should be set to high resistance, otherwise it may affect the serial port communication with the computer. If you want to use another hardware serial port, you can modify the uar yourself The corresponding hardware settings in the void fcc\_uart\_init () function of the t.c file.



Figure 3

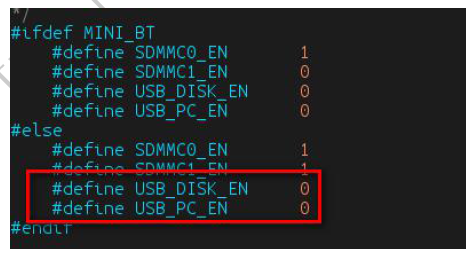


Figure 4

* FCC certification PC tool FCCAssist\_1.5.exe:
* Depending on the type of system that uses the computer, first open the REG\_WINDOWS\_UART folder and click register\_uart\_WIN7.bat or register\_uart\_WINXP.bat batch to follow the library file;
* Open the software FCCAssist\_1.5.exe, click serialport to select the drop-down box, select the correct serial port, if the normal next to the red flag will become green, as shown in figure 5;
* During the test process, set the appropriate configuration items as needed, then press the sendconfiguration button, if the send is successful as shown in the bottom left corner of the diagram, otherwise the transmission will not be successful (figure 5).

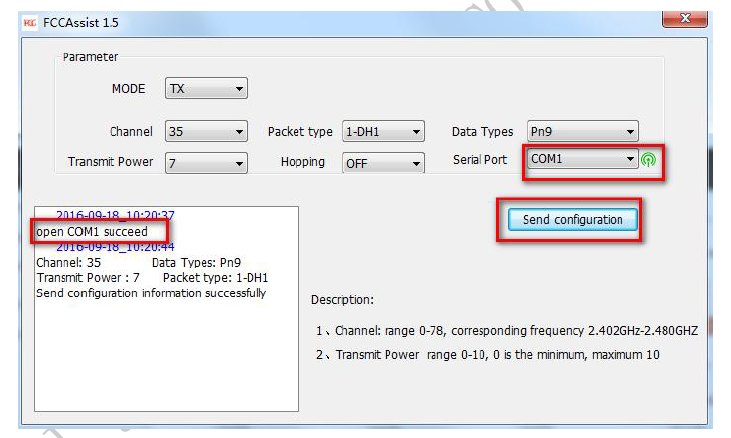


Figure 5

Note: Sdk uses the usb port by default to communicate with the computer. If the port links to the serial board dead, (the system detects the usb online condition on startup), it can cause the computer to fail to boot normally. Therefore, the prototype should be turned on after the usb port and serial port board link up with the computer communication.

5.2 BQB

* Introduction to BQB configuration
* To enable BQB authentication, you need to configure the BT\_MODE macro to TEST\_BQB\_MODE in sdk\_cfg.h, as shown in Figure 6.



Figure 6

* RF Test of BQB
* Open the TEST\_BQB\_MODE macro, the prototype can enter the dut test, the phone can search the configured Bluetooth name after booting, the phone is not connected, only for the BQB test connection.
* The antenna end of the prototype to weld shielding wire, and the antenna end of the material can not be changed first.
* Go to the lab to link the prototype and the test instrument to screen wire, power on the link test can be.
* There is one setting item with a central frequency of 2M.
* When testing a RF, there may be items that require special processing, try to pass all the items once, and finally take
* Look at those items and see what to do with them.
* Profile Test of BQB
* Open NORMAL\_MODE mode, prototype antenna does not need to weld shielding wire;
* According to the customer's prototype function to select a good test items, the test is mainly test functions such as up and down, pause, answer, hang up, back chain, and so on.

Chapter6 Instructions for Use Bluetooth Development

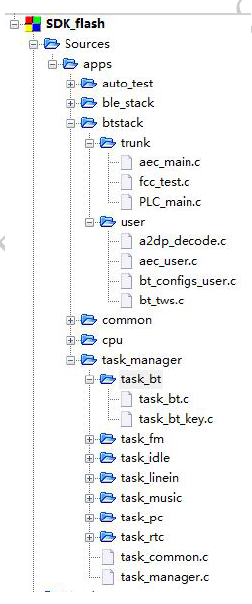
6.1 Terms and Abbreviations

Note: List the definitions of the terminology used in this document and the original phrase of the foreign language initials.

|  |  |
| --- | --- |
| Abbreviations and terms | Explanation |
| AC692x | Jieli Technology AC692x series chip |
| 1 to1 | Bluetooth only supports one phone connection |
|  |  |
|  |  |
|  |  |
|  |  |

6.2 Development Instructions

* The AC692X project opens:
* Before you open the project, make sure you have installed Jerry's latest release of jl\_toolchain and CodeBlocks.
* Double-click the sdk.cbp file in the apps directory, or drag the sdk.cbp file to open under the CodeBlocks shortcut, the project will be opened.
* Engineering catalogue description:
* When the project opens, the project catalog looks like this



Sources : Some of the . c files that users can modify can also add their own c files at their own discretion.

Headers: used to put the header file, the header file under apps can be modified, but the header file under include\_lib is the header file corresponding to the library. Users should be careful not to modify it at will, if you want to modify the library and update the header file.

* Main C file for the Bluetooth section

1, the file aec\_main.c

Some process code for call echo cancellation, including the addition of the initialization and arithmetic module

2, the file fcc\_test.c

Parameter setting of FCC authentication mode

3, the file PLC\_main.c

The packet loss repair module code of the call does not need to be changed in general.

4, the file a2dp\_decode.c

Some decoding processes for Bluetooth music are mainly music modules outside the open and close libraries. No need for general conditions

Want to change

5, the file aec\_user.c

Mainly used to set echo cancellation parameters, echo cancellation can refer to this file

6, the file bt\_configs\_user.c

Mainly used for some parameter settings and callback function registration of Bluetooth, this will be modified frequently. File protocol selection, the following macros can choose the Bluetooth-supported protocol when configuring bt\_configs\_user.c.

///---sdp service record profile- User selection support agreement--///

#define USER\_SUPPORT\_PROFILE\_SPP 1

#define USER\_SUPPORT\_PROFILE\_HFP 1

#define USER\_SUPPORT\_PROFILE\_A2DP 1

#define USER\_SUPPORT\_PROFILE\_AVCTP 1

#define USER\_SUPPORT\_PROFILE\_HID 0

Description of the main functions in bt\_configs\_user.c:

◎ Function static void bt\_setup\_init(u8 \*adr, char \*name, u8 idx, char \*pin\_code);

The flow of a classic Bluetooth profile.

◎ Function static void bt\_function\_select\_init (); is a parameter configuration function, this function concentrates most of the functions for secondary development settings. Sometimes the patch adds some new configuration and should be placed in this function.

◎ Function static void ble\_config\_select\_init(void); is the process of reading the BLE Bluetooth configuration file.

◎ Function static void bredr\_handle\_register () set up a callback function to register some Bluetooth libraries. These registered libraries will be used in the corresponding process.

◎ Function void bt\_mode\_init () is the initialization of Bluetooth hardware and Bluetooth protocol stack. Some will have a sequence requirement, do not arbitrarily adjust the initialization position.

7, the file bt\_tws.c

A process is used to process files in the box, and some SDK versions do not support the box.

8, the file task\_bt.c

Process processing, message processing, which will often modify the file.

The main function description:

◎ function void bt\_work\_state\_control (u8 enble); control Bluetooth can be found connectable.

◎ function void hook\_hfp\_incoming\_phone\_number (char \*number, u16 length); If a call comes in, the library will call this callback function to the upper feedback phone number.

◎ function int btstack\_status\_update\_deal (u8 \* info, u16 len), this function belongs to a callback function in the library, the flow of the Bluetooth protocol stack is directly called, be careful not to add too much delay or wait for operation in this function, it will affect this Bluetooth command. The resolution of the receipt, it is recommended to judge the state of the process, the complex process pushes the message to the void task\_bt\_deal (void \*hdl) main loop function processing. Other callback functions should also pay attention to.

◎The function void bt\_discon\_complete\_handle(u8 \*addr, int reason) is the same as before processing some messages on the Bluetooth connection and disconnected.

◎ function static void \*task\_bt\_init (void \* priv) mode entry will be called.

◎ function static void task\_bt\_exit (void \*\*hdl) mode exit will be called.

◎ function void task\_bt\_deal (void \*hdl) main loop function, processing button messages or other situations generated messages.

Note that some of the files are background management functions, and will be used later. AC692x does not support background.

9, the file task\_bt\_key.c

Button message table

10, the file bt\_ui.c

Display processing with screen version

The most important header file for Bluetooth----avctp\_user.h

This header file defines the status of many Bluetooth libraries and Bluetooth supported commands. When you need to develop personalized Bluetooth function, you can query the command interface of the file. In many cases, update the library to pay attention to whether there is any file in the patch to update.

Chapter7 Instructions for Using Music Development

7.1 Overall Design

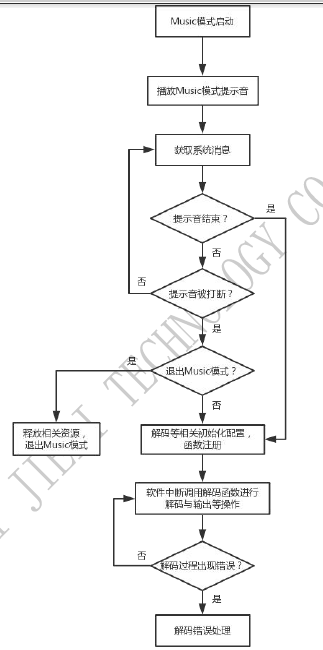
* Functional overview:

The Music task implements the following functionality:

* Support for playback of MP3,WMA,WAV,FLAC,APE files.
* Support to play SD card U disk flash.
* Support for single channel decoding.
* Support song play time, total time, song name, etc.
* Support song breakpoint playback.
* Support fast forward, fast back, reread, pause, and continue to play.
* Support variable sampling, channel control EQ / spectrum, volume / automatic mute and other sound effects.
* Supports multiple sound effects to be turned on at the same time.

7.2 Overall Architecture Design

* The overall software flow chart of the music pattern is shown in figure 1.2.1:
* Music playback initialization includes file playback mode, device selection mode, supporting file suffix settings, decoding error handling and other functions of registration.
* Before decoding, the equipment to be operated will be checked online, and the file format of the file to be played will be checked. If the device is not online or there is no file in the device, the file is damaged, the file format is not supported, etc. Will do the corresponding error handling.
* The function call related to decoding is carried out in the interrupt service function of the soft interrupt. By registering the soft interrupt, with the data output from the DAC side, the interrupt flag of the soft interrupt is set up at any time. The decoding and output operations are carried out in the soft interrupt service function, and the errors generated in the decoding process are handled.
* Because decoding and other operations are in the soft interrupt service function, so in the playback process support up-and-down, pause\ play, plug devices and other responses.
* The system supports multiple sound effects open at the same time, the decoded data through sound processing superimposed output.



Get system messages

Whether the tone ends

Start the Music mode

Play the tone of the Music mode

NO

NO

NO

NO

YES

YES

YES

YES

Decoding error handling process

When the software is interrupted, the decoding function is called to perform decoding and output operations.

Perform initialization and other related initialization configurations, function registration

Release related resources and exit Music mode

Is there an error in the decoding process?

Whether to exit the Music mode

Whether the tone is interrupted

Figure 7.2.1 Flowchart of the music mode

7.3 Decoding Channel Instructions

* The decoding process is as follows:
* Initialize the decoding system parameters to enable decoding libraries in different formats.
* File format check, call the corresponding decoding function.
* Decoding and decoding output.
* Error handling when decoding error.
* In the process of playing, receive message processing, according to different messages, there are song control, song switching, equipment switching, sound control and other operations.
* After a song is played, look for the next song to play automatically.
* If a new device is inserted, go to the latest inserted device and look for songs to play.
* If the playing device is unplugged, send a SYS\_EVENT\_DEC\_DEVICE\_ERR message, look for the next device to play, and wait to exit music mode if there is no device left.

7.4 Partial API Function Instructions

Function prototype:

MUSIC\_PLAYER \*music\_player\_creat(void)

Function description: creation and setting of Music play control handle

Parameter description: None

Returns: NULL: creation failed

Other: Create successfully, return the corresponding handle address

Tbool music\_player\_play(MUSIC\_PLAYER \*obj, MUSIC\_PLAYER\_BP \*bp\_info, u8 is\_auto)

Functional Description: Flow control of Music play

Parameter description: \*obj Music play control handle

\*bp\_info breakpoint information

Is\_auto auxiliary parameter

Returns: false: Played failed

True: Played successfully

Chapter8 Instructions for Using Radio Development

8.1 Overall Design

* Systems:

This function is mainly based on the SDK development kit to achieve the function of radio.

The main functions of FM application include:

* Automatic platform search mode, manual platform search mode, semi-automatic platform search mode.
* Support pause and broadcast radio.
* Support breakpoint memory, can remember the last playing frequency.
* Overall architectural design:

Radio is divided into two functional modules:

1. Radio main mode module: Initialize the FM module, play the current channel, and press the up and down buttons to select the station to play.
2. Automatic search, semi-automatic search, manual search.Application startup, application exit.

* Application launch, application exit

The radio mode can be entered in the following ways:

Press Mode to switch mode, enter FM mode, and press Mode again to jump out of FM mode.

8.2 Radio Design Instructions

* Module description

In the radio process, a unified interface is adopted to be compatible with a variety of radio modules. Different radio modules can be added to the radio process by providing the following functions:

* Get module ID function
* Startup/initialization function
* Close function
* Set the volume function
* Set the frequency function
* Process function

The radio receiving process has already added the basic radio usage function. The main function needs to be connected to the basic driver of the module to realize the following functions:

* Full frequency search
* Search on/next frequency
* Move up/down frequency
* Mute, unmute

8.3 Built-in Radio Searching Parameters Instructions

* Introduction to built-in FM search parameters
*  FMSCAN\_SEEK\_CNT\_MIN: The zero crossing count determines the minimum value.
*  FMSCAN\_SEEK\_CNT\_MAX: The zero crossing count determines the maximum value.
*  FMSCAN\_CNR: signal to noise ratio threshold.

These three parameters are set in the function fm\_inside\_io\_ctrl (SET\_FM\_INSIDE\_SCAN\_ARG1, as follows:

Fm\_inside\_io\_ctrl(SET\_FM\_INSIDE\_SCAN\_ARG1, FMSCAN\_SEEK\_CNT\_MIN,

FMSCAN\_SEEK\_CNT\_MAX, FMSCAN\_CNR);

*  True radio station judgment: When scanning a radio station, the signal zero-crossing statistics seek\_cnt is in [FMSCAN\_SEEK\_CNT\_MIN, within the range of FMSCAN\_SEEK\_CNT\_MAX], and the current station's signal noise cnr>= FMSCAN\_CNR, it is judged as true, otherwise it is judged as false.
* Clear and unclear radio features
* Characteristics of clear normal radio station: In a fixed time, the signal zero-point statistical value seek\_cnt is in a relatively small range, and the value of signal-to-noise ratio cnr is relatively large.
* No station (white noise) or unclear station characteristics: seek\_cnt is generally large, and cnr is relatively small.
* Built-in FM general debugging method:

1. Configure the search by default parameters. If the search station does not meet the requirements, first check the hardware, if there is no problem with the hardware, continue with step 2.

2. Execute a search radio (key message MSG\_FM\_SCAN\_ALL\_INIT), open the program to print, view seek\_cnt, cnr0, cnr1 of all stations. Preliminary statistics of each station's zero-crossing seek\_cnt, signal-to-noise ratio cnr0, cnr1.

Typical search radio printing parameters are as follows:

[freq: 875 seek\_cnt: 554 cnr: 22]

[freq: 876 seek\_cnt: 385 cnr:-21]

[freq: 877 seek\_cnt: 545 cnr: 0]

Adjust the signal-to-noise ratio threshold FMSCAN\_CNR as needed, and the zero-crossing value range [FMSCAN\_SEEK\_CNT\_MIN,

FMSCAN\_SEEK\_CNT\_MAX], when (cnr>=FMSCAN\_CNR))&&(FMSCAN\_SEEK\_CNT\_MIN<=seek\_cnt)&&(seek\_cnt<=FMSCAN\_SEEK\_CNT\_MA

When X), the current station judges to be a true station, otherwise it is judged to be a false station.

Note: When searching for the seek\_cnt/cnr information while searching for a station, you need to call the following function to open the relevant print in the library.

Fm\_inside\_io\_ctrl (SET\_FM\_INSIDE\_PRINTF, 1); //1 Open the print in the library. 0 Turn off printing in the library.

If you suspect that the library is open for printing when searching for radio stations, it will affect the search parameters. You can turn off the printing when searching for radio stations.

The function uniformly prints the search parameters (the first time the parameters are stored in the internal RAM).

Fm\_insice\_scan\_info\_printf(875,1080);//Print the search radio parameters from 87.5 to 108.0.

* Common search radio problem processing:

1) The number of radio stations received is not enough: first lower FMSCAN\_CNR, if not, then relax[FMSCAN\_SEEK\_CNT\_MIN,FMSCAN\_SEEK\_CNT\_MAX]

2) Too many fake stations: first increase FMSCAN\_CNR, then narrow [FMSCAN\_SEEK\_CNT\_MIN,

FMSCAN\_SEEK\_CNT\_MAX].

*  Other API functions:

1) void fm\_inside\_set\_stereo(u8 set); //Two-channel (stereo) effect setting. set value range [0,127].

When the set value is 0, it is equivalent to a complete mono.

The closer the value of set is to 127, the more obvious the two-channel effect is.

When the set value is 127, it is two-channel.

2) void fm\_inside\_set\_abw(u8 set); //audio bandwidth setting, set value range [0,128].

The larger the value of set, the wider the audio bandwidth. Bandwidth adjustable range [2k, 16k]

3) void fm\_inside\_deempasis\_set(u8 set); //To aggravate parameter settings. set can only be set to 0 or 1.

When the set value is 0, the de-emphasis time parameter is 50us.

When the set value is 1, the de-emphasis time is 75us.

4) s16 fm\_inside\_rssi\_read(void); //Receive RSSI value acquisition. The unit is dB.

Chapter9 Instructions for Using Clock Development

9.1 Introduction to Chapters

This section describes the design development documentation that can be used to develop the AC692x \_ SDK application , as well as references to testers for testing RTC applications . The overall function of RTC application , interface and data attribute of RTC are defined in the document ; the basic structure , function module and name of each program are divided into the detailed design and coding of RTC application .

9.2 Overall Design

* Overview of requirements

This module is mainly based on the AC692x\_SDK system development kit to implement RTC functions. The main features of the RTC application include:

(1) Support system time adjustment.

(2) Support alarm setting processing.

The development of this module is based on the Jieli AC692x\_SDK software development kit and the corresponding hardware platform.

* Overall architecture design

RTC is mainly divided into three functional modules:

 Display module: mainly shows the current time

Set module: set system time and alarm time

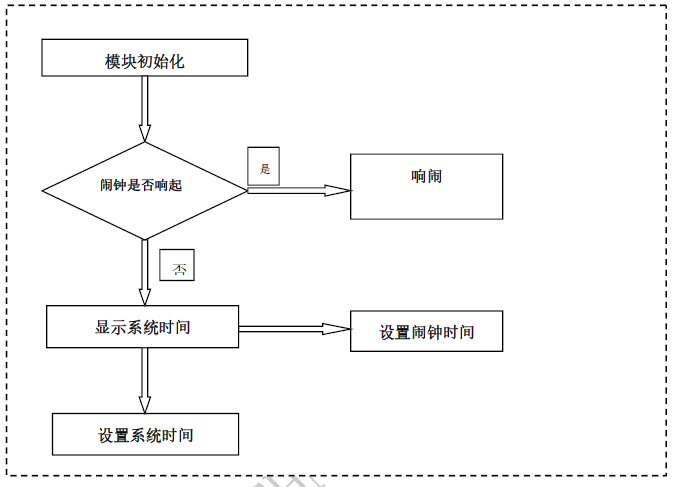
Alarm response module: mainly responds to the alarm

* Overall architecture

Figure 10.2.1 RTC flow chart

* Functional module division

|  |  |  |
| --- | --- | --- |
| Module name  System entrance module alarm response  Display module  Clock setting module | Function brief  Set system related information and respond to alarm operations  Display system time, set system time and alarm time  Set time and alarm operation flow | Corresponding file  Rtc.c  Rtc\_ui.c  Rtc\_setting.c |



NO

YES

Set the alarm time

Alarm clock

Set system time

Display system time

Does the alarm sound?

Module initialization

|  |  |  |
| --- | --- | --- |
| Module name  System entrance module alarm response  Display module  Clock setting module | Function brief  Set system related information and respond to alarm operations  Display system time, set system time and alarm time  Set time and alarm operation flow | Corresponding file  Rtc.c  Rtc\_ui.c  Rtc\_setting.c |

* Application life cycle
* Press Mode to switch mode, enter RTC mode, start running, when press Mode key again out of RTC mode, the application life cycle is over.
* Application startup

When switching to the RTC task, the void rtc\_info\_init (void) function is called to create and initialize the resources and hardware modules required for RTC.

The activation process consists of the following steps:

(1) initialize RTC resources and configure initial values

(2) initialize RTC information to see if the clock and alarm clock range is normal

* Application exit

When there is a message to switch tasks or devices to be inserted, task\_common processing will choose to switch tasks and exit RTC mode.

9.3 Design Instructions of System Entry Module

* Module description:

The corresponding file is task\_rtc.c is the entry point of the RTC application, and is responsible for the initialization of the system when entering. Responsible for closing resources and hardware modules when exiting.

* Module function

This module is the entry module of the application. It is responsible for system initialization, screen initialization, DAC initialization, etc. when entering.

* Module interface design

Functional Description: A callback function registered to the RTC interrupt. The flag distinguishes when the callback is generated. The if(RTC\_ISR\_ALARM\_ON == flag) branch to be processed when the alarm is triggered.

Function prototype: void rtc\_isr\_user\_handler(u8 flag)

Other instructions:

1) if(RTC\_ISR\_PCNT==flag) counts the interrupt generated by the overflow

2) if(RTC\_ISR\_LDO5V ==flag) is the interrupt generated by LDO5V detection

9.4 Design Instructions of Setting the time module

* Module description

This module adjusts the time and alarm clock to the customer.

* Module function

This module adjusts the time and alarm clock to the customer.

* Module interface design

Function description: According to the setting mode, set the time in RTC mode or alarm clock

Function prototype: void rtc\_setting(int msg)

For other interfaces with more RTC function settings, please consult the rtc\_api.h file.

* Exception handling

Timeout, main interface message, device insertion will exit setting mode.

9.5 Design Instructions of Alarm Module

* Module description

This module responds to the alarm.

* Module function

The module responds to the alarm and displays the alarm interface and the alarm sound.

* Module interface design

Function description: alarm clock, clock setting

Prototype: void rtc\_setting(int msg)

Function description: alarm switch

Prototype: void rtc\_sw(u8 flag)

For other interfaces with more RTC function settings, please consult the rtc\_api.h file.

* Exception handling

Timeout, main interface message, device insertion will exit setting mode.

9.6 Instructions of RTC Module Special Function

* Ordinary IO port

RTC has an independent IO port PORTR0 / 3 / PORTR1 default output 0, in which PORTR0 reuses 32.768KHz crystal pins for the RTC internal clock module .PORTR3 to reuse the 12-26MHz crystal oscillator, which can connect the external crystal oscillator as the system clock.

PORTR0~3 can wake up RTC low-power mode.

The function that sets the IO port looks like this, with only one IO port at a time:

void PORTR\_DIR(u8 port, u8 val);

void PORTR\_OUT(u8 port, u8 val);

void PORTR\_HD(u8 port, u8 val);

void PORTR\_PU(u8 port, u8 val);

void PORTR\_PD(u8 port, u8 val);

u8 PORTR\_IN(u8 port);

Port:PORTR0, PORTR1, PORTR2, PORTR3 Val: 0 or 1

NOTE:

1. PORT3 pin must be stored outside 32.768KHz crystal oscillator, clock function can work properly

2. If PORTR3 uses a crystal oscillator, it cannot be set to output, otherwise the clock will stop

3. After entering low power consumption, PORTRIO can also work normally, and other common IO ports are in high resistance state.

* PORTR1 and PORTR2 can be used for ADkey, and the following code examples are provided

/\*1: PR1 port voltage drive ADC , 0: no this function\*/

//if you want to use port1 to be ADKEY io, you can set as below

PORTR1\_ADCEN\_CTL(1);

PORTR\_PD(PORTR1 , 0);

PORTR\_PU(PORTR1 , 0);

PORTR\_DIR(PORTR1 , 1);

PORTR\_DIE(PORTR1 , 0);

* The RTC module also supports counter overflow wake-up and generates corresponding interrupts.

Function Description: Set the relevant configuration for counting wake-up.

Prototype: void pcnt\_init(u8 port, u8 edge);

Function description: Set the count value of the count register.

Prototype: void set\_pcnt\_value(u8 value);

Function description: Get the count value of the count register.

Prototype: int get\_pcnt\_value();

* PORTR port long press reset function interface

Function description: setting the configuration of button reset

Prototype: int rtc\_port\_reset(u8 mode, u8 port, u8 enable, u8 edge);

* Low power consumption

The RTC supports the independent low-power mode. When the power is changed to low power, the main control power supply will be turned off. After entering the low power consumption, only the RTC module works. Please refer to the relevant chapter of low power consumption.

Chapter10 Instructions for Using PC Slave Development

10.1 Overall Design

* Systems:

This note is mainly based on the SDK development kit to achieve the functions of PC.

The main functions of PC application include:

* Support card reader function.
* Support sound card function.
* Support for HID functionality
* Overall architectural design:

The PC task is divided into three functional modules:

1. Card reader function module: slave mode PC can operate the storage device in the device.
2. Sound card function module: slave mode, acting as PC music player speakers.
3. HID operation module: the main slave acting as sound card, PC music player playing music sound, by controlling the upper / next music from the slave, to control the corresponding operation of the PC player.

* Application startup

The PC mode enters in two ways:

1. After USB is plugged into the device from the machine line, the device manager detects that the SYS\_EVENT\_PC\_IN, main thread is sent to the main thread, which forces it to exit the current mode and activates the PC mode.
2. By switching mode keys, the main thread forces out of the current mode and activates PC mode.

* Exit of the application:

When the Mode key is pressed to unwire the PC or other tasks need to be activated, the main thread forces out of PC mode

Release PC task resources

Closing cue tone

* Application dependency library and its interface description

usb\_lib.a --USB Module device library

lib\_usb\_syn --PC Mode AUDIO synchronization library

10.2 Design Instructions of System Entry Module

* Module description

The corresponding file is task\_pc.c, which is the entry point of PC application. It is responsible for the variable application and hardware initialization of PC mode when entering, and releasing the variable resource occupied by PC mode when exiting.

* Module function

This module is the entry module of the application program, which is mainly responsible for the initialization of the system when entering, such as the DAC initialization of AUDIO function and hid initialization.

10.3 Design Instructions of Card Reader Module

* Module description

This module mainly implements reading of large-capacity storage devices. Card\_reader\_io.c provides an IO connection between the reader and the device.

* Module function

The card reader function.

* Module interface design

Card reader function implementation function description:

*  Prototype: s32 app\_usb\_slave\_card\_reader(u32 cmd)
*  Function description: PC card reader execution flow function
*  Parameter description: cmd command
*  Return value description: execution status
*  Prototype: sUSB\_DEV\_IO \*get\_card\_read\_io(DEV\_TYPE dev\_type)
*  Function description: Get the IO of the card reader operating device
*  Parameter description: dev\_type device type
*  Return value description: Operation IO of the device

10.4 Design Instructions of HID Operation Module

* Module description

This module mainly realizes the corresponding operation of the player on the PC with the device connected to the PC wire.

* Module function

This module realizes HID function.

* Module interface design

HID function description:

* Function prototype: void usb\_slave\_hid (U32 key)
* Function description: PC HID function
* Parameter description: key: PC HID button command
* Return value description: none

10.5 Design Instructions of USB\_SPK Module

* Module description

This module mainly implements this device as a PC sound card function.

* Module function

This module implements the USB\_AUDIO function.

* Module interface design

PC\_AUDIO sound card

Function description:

* Function prototype: u8 pc\_set\_speaker\_vol( u32 pc\_mute\_status)
* Function description: PC AUDIO function volume setting function
* Parameter description: pc\_mute\_status: mute status
* Return value description: current sound card volume value
* Function prototype: void pc\_dac\_mute (bool mute\_status, u8 fade\_en)
* Function description: PC AUDIO mute setting function
* Parameter description: state of statusmute: mute status fadeen: fade in and fade-out setting
* Return value description: none

10.6 PC Detection Function

In each mode, when the PC line is plugged in, it will automatically jump to PC mode, or you can switch to PC mode by using the Mode button.

*  Function prototype: void pc\_check\_api(void)
*  Function description: PC online detection function
*  Parameter description: void
*  Return value description: void

The PC detection function is added to the device detection task. If the PC line is detected, the device detection task sends out the EVENT\_PC\_IN event and switches from the task\_common control to the PC mode.

Chapter11 F1A Warning Tone file

11.1 Overview of F1A Warning Tone

F 1A audio file is the audio format that our company customizes according to the characteristic of prompt audio file. Without affecting the sound quality, increase compression ratio, as far as possible so that the hint tone occupies less code space.

Turn on the corresponding function.

The corresponding functional macro is in the sdk\_cfg.h file

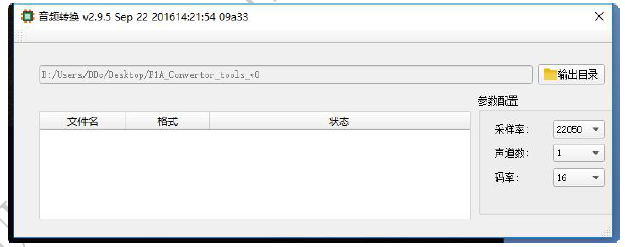
#define DEC\_TYPE\_F1A\_ENABLE ///<30K\_code\_space

Download the corresponding note file.

In the apps\download\ac692x\post\_build\download.bat file

isd\_download.exe …. power\_off.f1a bt.f1a music.f1a record.f1a linein.f1a radio.f1a pc.f1a wait.f1a connect.f1a disconnect.f1a ring.f1a ….

F1A\_Convertor\_tools\_v1.0 can convert normal audio files to F1A files.



Figure